Training Sound Event Classifiers Using Different Types of Supervision

Eduardo Fonseca

TESI DOCTORAL UPF / 2021

Thesis Directors:

Dr. Xavier Serra i Casals and Dr. Frederic Font Corbera

Music Technology Group

Dept. of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona, Spain

Dissertation submitted to the Department of Information and Communication Technologies of Universitat Pompeu Fabra in partial fulfillment of the requirements for the degree of

DOCTOR PER LA UNIVERSITAT POMPEU FABRA

Music Technology Group (http://mtg.upf.edu), Department of Information and Communication Technologies (http://www.upf.edu/dtic), Universitat Pompeu Fabra (http://www.upf.edu), Barcelona, Spain.

The doctoral defense was held on ..... at the Universitat Pompeu Fabra and scored as .....

Dr. Xavier Serra i Casals
(Thesis Supervisor)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

Dr. Frederic Font Corbera
(Thesis Supervisor)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

Dr. Emmanouil Benetos
(Thesis Committee Member)
Queen Mary University of London (QMUL) and The Alan Turing Institute, London, United Kingdom

---

Dr. Annamaria Mesaros
(Thesis Committee Member)
Tampere University (TAU), Tampere, Finland

Dr. Marius Miron
(Thesis Committee Member)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

This thesis has been carried out at the Music Technology Group (MTG) of Universitat Pompeu Fabra in Barcelona, Spain, from October 2016 to September 2021. This thesis is supervised by Dr. Xavier Serra i Casals and Dr. Frederic Font Corbera. This thesis included two research stays at Google Research Sound Understanding, from September 2019 to December 2019 in New York City, and from October 2020 to February 2021, online.

Work in Chapter 3 has been conducted in collaboration with Xavier Favory and Jordi Pons (Universitat Pompeu Fabra). Work in Section 5.6 has been conducted in collaboration with Shawn Hershey, Manoj Plakal, Daniel P.W. Ellis, Aren Jansen, and R. Channing Moore (Google Research, New York and California). Work in Section 6.3 has been conducted in collaboration with Diego Ortego (Dublin City University, Ireland). Work in Section 6.4 has been conducted in collaboration with Aren Jansen, Daniel P.W. Ellis, Scott Wisdom, and Marco Tagliasacchi (Google Research, California, New York, Massachusetts, and Switzerland). The organization of the DCASE Challenges described in Appendix A has been carried out in collaboration with Manoj Plakal and Daniel P.W. Ellis (Google Research, New York) and Frederic Font Corbera (Universitat Pompeu Fabra).

This work has been supported by the Department of Information and Communication Technologies (DTIC) PhD fellowship, Universitat Pompeu Fabra, and by the European Union's Horizon 2020 research and innovation programme (grant agreement 688382 AudioCommons), and by two Google Faculty Research Awards 2017 and 2018.

Acknowledgments

A few years back, before joining the MTG, Xavier Serra asked me to make a short proposal about something I'd like to work on. I remember coming across a couple of WASPAA papers about classification of sound events. I remember thinking: "I think I'd like to do this". A lot of the work carried out ever since is described in the next pages; work that would not have been possible without the support of many people that influenced me in the past few years, both at the technical and personal level.

First and foremost, I would like to express my enormous gratitude to my supervisors, Xavier Serra and Frederic Font, not only for giving me the opportunity to do this work and for invaluable advice, but also for giving me the freedom to choose my own path.

I would also like to thank all the researchers I had the opportunity (and pleasure) to meet and collaborate with during the past few years. I want to thank them for their help and feedback during these years and also for the enriching and valuable discussions, sometimes about work, sometimes about many other things. From the MTG, special thanks to Xavier Favory and Jordi Pons with whom I shared the first steps. Also, many thanks to Andrés Ferraro, Andrés Pérez-López, Rong Gong, Olga Slizovskaia, Dmitry Bogdanov, Sergio Oramas, Alastair Porter, Antonio Ramires, Pablo Zinemanas, Marius Miron, Pablo Alonso, Minz Won, Pritish Chandna, Rafael Caro, Sankalp Gulati, Ajay Srinivasamurthy and probably some more that I'm missing! From Google Research, I would like to express my sincere gratitude to my supervisors Daniel P.W. Ellis, Aren Jansen, and Manoj Plakal, for believing in me and teaching me a lot of things about sound recognition (and much more), and also special thanks to Shawn Hershey, R. Channing Moore, Scott Wisdom, Marco Tagliasacchi and John R. Hershey. I'm also grateful to Nicolas Turpault, Romain Serizel, Annamaria Mesaros and Toni Heittola from the DCASE community, and to Diego Ortego, Kevin McGuinness and Noel E O'Connor from Dublin City University. I would like to mark my gratitude to Luis Hernández from Universidad Politécnica de Madrid, who inspired me even before I knew I wanted to do a PhD. I also want to thank everyone who contributed annotations for Freesound audio, and very specially Mercedes Collado, Ceren Can, Rachit Gupta, Javier Arredondo, Gary Avendano and Sara Fernández, for their outstanding commitment and perseverance. Finally, I'm grateful to Lydia Garcia, Sonia Espí and Cristina Garrido from UPF, for their invaluable support on the administrative side.

I would also like to thank all my friends at Barcelona and Asturias, for their support along the way. Special thanks to Merch, Dani A., Julio, Alba, Carla,

Dani and Pitu, for supporting me in this journey in many different ways. Also many thanks to María Serrano, Elias, Xavi, Cate, Ali, Carmen, Solana, Auro, Inma, and to María Silva, Brian, Gelo, Eva, and also Javi, Jaime and Álvaro from Room 114.

Finally, I cannot thank enough my parents, and very especially my sister, for their constant and unwavering support.

Abstract

The automatic recognition of sound events has gained attention in the past few years, motivated by emerging applications in fields such as healthcare, smart homes, or urban planning. When the work for this thesis started, research on sound event classification was mainly focused on supervised learning using small datasets, often carefully annotated with vocabularies limited to specific domains (e.g., urban or domestic). However, such small datasets do not support training classifiers able to recognize hundreds of sound events occurring in our everyday environment, such as kettle whistles, bird tweets, cars passing by, or different types of alarms. At the same time, large amounts of environmental sound data are hosted in websites such as Freesound or YouTube, which can be convenient for training large-vocabulary classifiers, particularly using data-hungry deep learning approaches. To advance the state-of-the-art in sound event classification, this thesis investigates several strands of dataset creation as well as supervised and unsupervised learning to train large-vocabulary sound event classifiers, using different types of supervision in novel and alternative ways. Specifically, we focus on supervised learning using clean and noisy labels, as well as self-supervised representation learning from unlabeled data.

The first part of this thesis focuses on the creation of FSD50K, a large-vocabulary dataset with over 100h of audio manually labeled using 200 classes of sound events. We provide a detailed description of the creation process and a comprehensive characterization of the dataset. In addition, we explore architectural modifications to increase shift invariance in CNNs, improving robustness to time/frequency shifts in input spectrograms. In the second part, we focus on training sound event classifiers using noisy labels. First, we propose a dataset that supports the investigation of real label noise. Then, we explore network-agnostic approaches to mitigate the effect of label noise during training, including regularization techniques, noise-robust loss functions, and strategies to reject noisy labeled examples. Further, we develop a teacher-student framework to address the problem of missing labels in sound event datasets. In the third part, we propose algorithms to learn audio representations from unlabeled data. In particular, we develop self-supervised contrastive learning frameworks, where representations are learned by comparing pairs of examples constructed via data augmentation and automatic sound separation methods. Finally, we report on the organization of two DCASE Challenge Tasks on automatic audio tagging with noisy labels. By providing data resources as well as state-of-the-art approaches and audio representations, this thesis contributes to the advancement of open sound event research, and to the transition from traditional supervised learning using clean labels to other learning strategies less dependent on costly annotation efforts.

Resumen

El interés en el reconocimiento automático de eventos sonoros se ha incrementado en los últimos años, motivado por nuevas aplicaciones en campos como la asistencia médica, smart homes, o urbanismo. Al comienzo de esta tesis, la investigación en clasificación de eventos sonoros se centraba principalmente en aprendizaje supervisado usando datasets pequeños, a menudo anotados cuidadosamente con vocabularios limitados a dominios específicos (como el urbano o el doméstico). Sin embargo, tales datasets no permiten entrenar clasificadores capaces de reconocer los cientos de eventos sonoros que ocurren en nuestro entorno, como silbidos de kettle, sonidos de pájaros, coches pasando, o diferentes alarmas. Al mismo tiempo, websites como Freesound o YouTube albergan grandes cantidades de datos de sonido ambiental, que pueden ser útiles para entrenar clasificadores con un vocabulario más extenso, particularmente utilizando métodos de deep learning que requieren gran cantidad de datos. Para avanzar el estado del arte en la clasificación de eventos sonoros, esta tesis investiga varios aspectos de la creación de datasets, así como de aprendizaje supervisado y no supervisado para entrenar clasificadores de eventos sonoros con un vocabulario extenso, utilizando diferentes tipos de supervisión de manera novedosa y alternativa. En concreto, nos centramos en aprendizaje supervisado usando etiquetas sin ruido y con ruido, así como en aprendizaje de representaciones auto-supervisado a partir de datos no etiquetados.

La primera parte de esta tesis se centra en la creación de FSD50K, un dataset con más de 100h de audio etiquetado manualmente usando 200 clases de eventos sonoros. Presentamos una descripción detallada del proceso de creación y una caracterización exhaustiva del dataset. Además, exploramos modificaciones arquitectónicas para aumentar la invariancia frente a desplazamientos en CNNs, mejorando la robustez frente a desplazamientos de tiempo/frecuencia en los espectrogramas de entrada. En la segunda parte, nos centramos en entrenar clasificadores de eventos sonoros usando etiquetas con ruido. Primero, proponemos un dataset que permite la investigación del ruido de etiquetas real. Después, exploramos métodos agnósticos a la arquitectura de red para mitigar el efecto del ruido en las etiquetas durante el entrenamiento, incluyendo técnicas de regularización, funciones de coste robustas al ruido, y estrategias para rechazar ejemplos etiquetados con ruido. Además, desarrollamos un método teacher-student para abordar el problema de las etiquetas ausentes en datasets de eventos sonoros. En la tercera parte, proponemos algoritmos para aprender representaciones de audio a partir de datos sin etiquetar. En particular, desarrollamos métodos de aprendizaje contrastivos auto-supervisados, donde las representaciones se aprenden comparando pares de ejemplos calculados a través de métodos de aumento de datos y separación automática de sonido.

Finalmente, reportamos sobre la organización de dos DCASE Challenge Tasks para el tageado automático de audio a partir de etiquetas ruidosas. Mediante la propuesta de datasets, así como de métodos de vanguardia y representaciones de audio, esta tesis contribuye al avance de la investigación abierta sobre eventos sonoros y a la transición del aprendizaje supervisado tradicional utilizando etiquetas sin ruido a otras estrategias de aprendizaje menos dependientes de costosos esfuerzos de anotación.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

Sounds within our everyday environment carry a huge amount of information about the physical events happening around us (Krause, 2012). We, humans, perceive sounds constantly every day, either at home or in the street, at work or out in the nature. We have the ability to recognize and understand most of these sounds, and that plays a crucial role in our everyday life. For instance, it allows us to communicate with others and to enjoy listening to music. But beyond speech and music, recognizing all kinds of *sound events* like a kettle whistle, a cat purring or a fire alarm, allows us to interact with objects and animals, and, for instance, avoid hazardous situations. The latter in particular can be specially useful for survival purposes, which highlights the importance of the human auditory system. Perhaps because of that, the human auditory system has been fine-tuned over millions of years of training and we have become amazingly good at distinguishing hundreds of *everyday sounds* (Masterton et al., 1969).

However, the identification of everyday sounds by machines still lags far behind the capabilities of the human auditory system. Automatic processing and recognition of sounds by computers in order to understand the environment is a challenging endeavour. This is due to a number of reasons, for example, the large diversity of sound sources that we ideally would like to recognize, and the large variability of acoustic patterns that sound sources of the same type can produce. Furthermore, the acoustic characteristics of these sounds are highly varying, including all sorts of acoustic noise or sounds that overlap with each other. The benefits of machines successfully performing this perceptual task and listening like a person listens would be manifold. For example, these algorithms could have a great impact on healthcare through context-aware applications (Schilit et al., 1994) and intelligent wearable interfaces (Xu et al., 2008) that could improve the life quality of the hearing impaired or the elderly,

or as part of smart domestic assistants to aid everyday life (e.g., Google Home or Amazon Alexa). In addition, nowadays we have access to huge amounts of acoustic material to enjoy and learn from, sitting in websites like Freesound or YouTube. For instance, the soundtracks of YouTube videos have millions of hours of sound (Lyon, 2017). Sound understanding algorithms could be useful for automatic description of such multimedia content, which enables automatic subtitling and captioning, or could lead to better content-based organization and retrieval for these large multimedia repositories. Yet, further research is needed to develop robust systems capable of recognizing a wide range of sound events in different types of audio streams, and performing those tasks similarly as humans do. The computational analysis of environmental sounds has received growing interest from the research community in recent times. Research efforts targeting such goals are typically categorized under the field of *machine listening*.

Broadly speaking, the machine listening field is aimed at studying computational methods for the automatic analysis and understanding of sound, thus enabling machines to be aware of their environment (Wang, 2010). Within this field, Sound Event Recognition (SER) is the task of automatically identifying the sounds occurring in our daily lives, assigning a label within a target set of sound classes. SER has gained increasing attention in the past few years, becoming a useful component in applications related to healthcare (Drugman et al., 2012; Hüwel et al., 2020; Messner et al., 2020), identification of urban noise sources for urban sound planning (Bello et al., 2019), bioacoustics monitoring (Xu et al., 2017; Lostanlen et al., 2019; Cramer et al., 2020), multimedia event detection (Wang et al., 2016), large-scale event discovery (Jansen et al., 2017), acoustics surveillance for security issues (Crocco et al., 2016; Sánchez-Hevia et al., 2017), or noise monitoring for industrial applications (Morrison & Pardo, 2019). These applications have a high impact on human welfare, making this field have a great potential of social impact in multiple areas. Further, the SER research community has grown substantially over the last decade. This is evidenced by the increasing traction of the *Detection and Classification of Acoustic Scenes and Events (DCASE)* Challenge and Workshop (Mesaros et al., 2017a), which, among other things, promote research and evaluation on common publicly available datasets. This traction can be noticed not only in terms of participants and authors, but also in terms of companies sponsoring and participating in such events and therefore very interested in the field.

Within the broad context of machine listening, this thesis particularly focuses on dataset creation methods and computational methods for SER using different types of supervision, as we explain next. The SER research context at the time when the work of this thesis started (late 2016) can be broadly described in terms of the methods and the datasets used. In terms of methods, the SER problem was approached using mainly supervised learning, with

the research community gradually shifting from feature engineering methods (Mesaros et al., 2010; Cotton & Ellis, 2011; Foggia et al., 2015) to deep learning (Cakir et al., 2015; Piczak, 2015a; Parascandolo et al., 2016), which was beginning to be adopted after its success in fields like computer vision or speech recognition. In terms of datasets, most publicly available datasets were of relatively limited size (less than 9h of audio in Salamon et al. (2014); Piczak (2015b); Foster et al. (2015)) and/or using a vocabulary limited to specific domains (e.g., urban or domestic sounds with up to 10 classes (Salamon et al., 2014; Foster et al., 2015)). These two characteristics allowed for a careful annotation of the audio material in the mentioned datasets. However, such datasets present some limitations. First, deep learning approaches are data-hungry techniques that require large amounts of training data in order to show its potential—it has been shown that the more data the better (Sun et al., 2017). Yet the data resources available when this thesis started were not large. Second, SER research used to be limited to the few domain-specific datasets which were available at the time. Whether it is about domestic sounds, urban sounds, or other domains, each of those datasets alone encompasses just a small bit of the variety of sounds in our everyday environment. At the same time, the type of sounds that can be recognized by sound event recognizers is limited by the diversity of the data used to train them. Consequently, the limited size and coverage of the publicly available datasets was limiting the recognition performance and the applicability of sound event recognizers to different scenarios. Therefore, there was a need for more audio data with higher diversity, in order to allow training general-purpose classifiers able to recognize tens, or ideally hundreds, of sound classes.

Despite these needs, there existed large amounts of environmental sound data with a high diversity of everyday sounds in web repositories such as Freesound,[1] YouTube[2] or Flickr.[3] These repositories have two main aspects in common. First, they host very large amounts of audio, taken from video recordings (in the case of YouTube and Flickr), or audio recordings (in the case of Freesound). Second, the audio data generally lacks reliable homogeneous labels describing the audio content; instead, these repositories typically contain metadata such as textual descriptions or *tags* manually introduced by the contributors of such repositories. These user-provided metadata can be sparse and not necessarily informative of the audio content but potentially video-centered (in the case of YouTube and Flickr), or generally audio-centered but potentially incomplete or inaccurate (in the case of Freesound) (Font, 2015). In this context, the main question that we formulated at the beginning of this thesis was: What actions can we take to overcome the limitations of the SER field, and allow the

[1] https://freesound.org/
[2] https://www.youtube.com/
[3] https://www.flickr.com/

improvement of coverage and performance in sound event recognizers? In this thesis, we identify four possible research avenues to pursue this goal, which are introduced in the next Sections (1.2.1, 1.2.2, 1.3.1 and 1.3.2), highlighting their main challenges and opportunities.

1.2 The Commonly-used Paradigm

The commonly-used paradigm as defined in this thesis consists of adopting the classical supervised learning approach. In this paradigm, classifiers are trained using labels that are assumed to be reliable, which are typically produced manually. In this context, the most obvious way to improve an existing method is to build a labelled dataset containing more data and higher diversity than those previously existing. However, assuming the amount of data that can be gathered in this way is somewhat limited due to the costly annotation process, an alternative is to adopt approaches to improve the generalization of sound event recognizers. Sections 1.2.1 and 1.2.2 introduce these two research avenues, respectively.

1.2.1 Building a New Dataset

The most evident option to achieve the established goal is to build better datasets. For data-driven methodologies such as deep learning, it becomes clear that new, larger, and more comprehensive datasets for development and evaluation of SER models are important. This urgency resembles the evolution witnessed in the computer vision field, where the release of ImageNet allowed significant breakthroughs for image recognition and other visual tasks (Russakovsky et al., 2015).

In the SER field, the aforementioned web repositories can be utilized as sources of data for sound event dataset creation. The most prominent example of this is Google's AudioSet, released in 2017, shortly after the beginning of this thesis. AudioSet consists of ≈2.1M audio clips manually labeled using 527 classes (Gemmeke et al., 2017). Its unprecedented size, coverage and diversity represented a milestone that has transformed SER research, mitigating the aforementioned need for large-vocabulary sound event data resources. However, in our view, AudioSet has the major shortcoming of not being an open dataset. Specifically, AudioSet is composed of audio tracks taken from YouTube videos, which are not freely distributable due to YouTube Terms of Service. This is the reason why AudioSet is released as a dataset of audio features (instead of audio waveforms),[4] which limits the adoption of a number of SER methods.

[4] https://research.google.com/audioset/download.html

For this reason, some researchers opt to download the audio tracks from the original YouTube videos, despite the ambiguities related to user rights, and the fact that the constituent videos are gradually disappearing (see Section 2.4.1.2 for details).

After AudioSet, several efforts in dataset creation for SER have been made (e.g. Cartwright et al. (2019c); Turpault et al. (2019); Adavanne et al. (2019); Gharib et al. (2019); Dekkers et al. (2017); Fonseca et al. (2019b,c); Cartwright et al. (2020)). However, these recent datasets are task/domain-specific, or of a much more limited coverage (e.g., usually featuring few tens of classes), and some of them are composed of synthetic audio material. This contrasts with the computer vision field, where major efforts have been made to collect large datasets as alternatives to ImageNet (e.g. Lin et al. (2014); Li et al. (2017b); Kuznetsova et al. (2020)), allowing benchmarking on complementary recognition problems. Thus, the SER field lags far behind in terms of dataset availability.

At the beginning of this thesis, we understood that open dataset creation initiatives were needed to foster SER research. Building a fully-open, distributable and stable, large-vocabulary sound event dataset with reliable labels is useful for the research community. Not only it allows the training and stable benchmarking of general-purpose sound event classifiers, but also may have a broader impact in different strands of machine listening research. However, if the reference labels are to be accurate and reliable (for which careful manual annotation is likely to be needed) this can be a very laborious process. Building a dataset with these characteristics, which we call **F**ree**s**ound **D**ataset **50k** (*FSD50K*), has been an important focus throughout the development of this thesis, for which we provide details in Chapter 3.

1.2.2 Improving Generalization

Due to the costly process of manual annotation, it is likely that the amount of training data gathered as explained in the previous Section is less than ideal, especially for some rare or ambiguous classes. Therefore, techniques to compensate for this issue and increase the generalization of sound event recognizers are paramount. Generalization is the ability of a machine learning algorithm to perform well on previously unseen examples (Goodfellow et al., 2016). The most obvious (and probably effective) path to improve generalization to new examples is to gather and use more training examples, which leads us back to Section 1.2.1. However, when this is not feasible, alternative ways to improve generalization include improving the network architecture and using data augmentation.

The most common network architectures for sound recognition in recent years are Convolutional Neural Networks (CNNs) (Hershey et al., 2017; Kong et al., 2020a; Fonseca et al., 2020b; Gong et al., 2021b). Among the reasons why they have been widely accepted is the property of *translation* or *shift invariance*. According to this property, if the input to the network is translated (i.e., shifted) by a small amount, the output predictions should not change, or should change only minimally. However, recent studies in computer vision have put into question this commonly-assumed property, showing that small shifts in the input can affect the output predictions of CNNs substantially. Fluctuations in the output probabilities can yield changes in the top-predicted classes, leading to unstable behaviours sometimes triggered by perturbations imperceptible to humans. For example, Azulay & Weiss (2018) quantify that by shifting or resizing a random input image by one single pixel, the top class predicted can change with a probability of up to 15% and 30%, respectively. This and other related works empirically show the brittleness of CNNs against minor input perturbations, and their only-partial invariance to shifts (Engstrom et al., 2018; Zhang, 2019). As a result, increasing the robustness of CNNs against this kind of shifts is an issue of particular concern, especially when the training data (hence the generalization capability) is not abundant.

Another way to improve generalization is via data augmentation techniques, by generating additional training examples that can cover situations that may be encountered in real life, when the model is doing inference on unseen data. Several data augmentation techniques have been developed for environmental sound processing tasks, including pitch shifting, time stretching, or dynamic range compression, to name a few (Salamon & Bello, 2017). Others are based on the simple concept of mixing sounds—an audio-informed process that perhaps seems more suitable for general-purpose audio applications. This mixing strategy has been recently adopted and found particularly effective for various tasks in sound event research (Tokozume et al., 2018; Jansen et al., 2018; Kong et al., 2020a; Wang & van den Oord, 2020).

Implementing measures to improve generalization overall and, in particular, robustness to small time/frequency shifts is, therefore, a worthwhile research avenue. Tackling this problem via architectural modifications and data augmentation methods can be a way to increase recognition performance without gathering more training examples. This is one of the objectives of this thesis, covered in Chapter 4.

1.3 A New Perspective on Sound Event Recognition

The commonly-used paradigm introduced in Section 1.2 is fundamentally constrained by the reliance on careful audio annotations. Carefully annotating sound events is a costly and time-consuming process which often leads to limited training data, this being one of the main weaknesses in deep learning methodologies. However, as mentioned in Section 1.1, there exist large amounts of varied everyday sound data in web repositories such as Freesound or YouTube, sometimes accompanied by user-provided metadata, e.g., in the form of tags. A new perspective on SER consists of accepting that the external supervision of training data may not be optimal, or may be even non-existent, and finding ways to train sound event classifiers in these more adverse scenarios. For example, one alternative to gather more training data is to use less precise annotations over larger amounts of data, finding a trade-off between annotation quality and data size. Another alternative is to leverage the aforementioned *imperfect* sources of data *as is*, utilizing the user-provided tags as noisy labels without turning to costly annotation processes. These two alternatives are representative use cases of training classifiers with labels that are potentially noisy and imprecise. This research avenue typically allow training at a larger scale, overcoming the limitations imposed by annotation budgets. However, techniques to mitigate the negative effect of label noise are usually required in order to increase performance, which can be limited by the poorer supervision. This research avenue is further motivated in Section 1.3.1. Another line of work that has recently gained attention in the deep learning community consists of disregarding any external supervision whatsoever, and turning to the learning paradigm of *unsupervised learning*. Here, self-supervision can be obtained solely by looking at patterns in the audio data. This type of supervision can be used to learn rich audio representations without external labels, which allows using potentially unlimited data. We introduce this research direction in Section 1.3.2.

1.3.1 Learning with Noisy Labels

The commonly-used paradigm introduced in Section 1.2 is based on the assumption that the labels accompanying the training data are correct. However, this ideal state may not always be realistic, and sometimes a weaker supervision given by noisy labels is the only feasible choice. In these cases, we can reject the assumption that the supervision is fully reliable, and adopt measures to improve the recognition performance in presence of noisy labels.

As discussed earlier, in sound event classification there is increasing demand for large and varied data resources to exploit the capacity of deep architectures. Creating datasets for supervised learning typically consists of two stages: *i)* data acquisition (e.g., retrieving data from sites like Freesound or Youtube, or doing recordings) and *ii)* data curation (organizing, cleaning and, most importantly, labeling the data). Manual labeling is costly and is typically the limiting factor on audio datasets. Thus, creators are often forced to compromise between dataset size and label quality, especially when annotating large amounts of audio data with a diverse set of categories. Although some sound event datasets are exhaustively labeled, e.g., Salamon et al. (2014); Piczak (2015b); Foster et al. (2015) their size is limited (less than 9h of audio). In contrast, the AudioSet dataset released at the beginning of this thesis consists of over 5000h of audio labeled with 527 classes, but the labeling is not as precise (Gemmeke et al., 2017). In particular, label error in AudioSet is estimated at above 50% for ≈18% of the classes.[5] Thus, we are witnessing a transition away from small and exhaustively labeled datasets, in favour of larger datasets that inevitably include some amount of label noise.

On the other hand, online repositories such as Freesound or Youtube host significant volumes of audio content with associated metadata. In this new perspective, these repositories can be utilized as sources of data for training sound event classifiers directly, without subsequent explicit human curation. Labels can be inferred automatically from user-provided metadata, e.g., tags, or by using pre-trained classifiers on the audio content. This method of gathering data supports rapid collection of large amounts of labeled data, at a much faster pace than the conventional manual dataset creation. However, this is at the likely cost of a substantial level of label noise arising from errors in the user-provided metadata, their transformation into labels, or sub-optimal pre-trained classifiers.

In this context, label noise emerges as a pressing issue for the future of large-scale sound event classification that can affect many practitioners. The effects of label noise can include performance decrease, increased complexity of learned models, or changes in learning requirements (Frénay & Verleysen, 2014), and have been reported to hinder the proper learning of deep networks (Arpit et al., 2017; Zhang et al., 2017). In sum, increasing robustness against label noise is a promising research direction that allows to be less dependent on careful manual annotations, and may lead to performance boosts when audio labels are unreliable. There is a large body of work focused on learning with noisy labels in computer vision, e.g., Reed et al. (2015); Goldberger & Ben-Reuven (2017); Han et al. (2018). However, when the work for this thesis started, there was little prior work on how to improve sound event classification in presence

[5]See https://research.google.com/audioset/dataset/index.html for details on how the quality is estimated, accessed 25th June 2020.

of noisy labels, e.g., Kumar & Raj (2017), and there was a lack of open data benchmarks for the study of label noise.

Consequently, learning with noisy labels has been a relevant topic in this thesis. Chapter 5 describes our contributions to establish learning with noisy labels as a research direction in SER. Specifically, we first report on the creation of an openly-available dataset that supports the investigation of label noise in sound event classification. Then, we explore a number of techniques to mitigate the effect of label noise, and apply them to web audio with noisy labels and to AudioSet (Gemmeke et al., 2017).

1.3.2 Self-supervised Learning

So far, in previous Sections, we have assumed that textual labels accompanying the audio data are always available. As we have seen, these textual labels can be provided by humans with better or worse quality, or, for example, they can be automatically derived from audio metadata, which usually generates a substantial amount of label noise. Either way, the introduced supervised learning paradigm depends on this external supervision. This dependence on external supervision carries important limitations, as discussed in previous Sections. On the one hand, the construction of human-labeled audio datasets for SER is notoriously time-consuming and subjective, imposing practical limitations on dataset size and quality. On the other hand, when generating labels from metadata, one might encounter situations where the metadata is too sparse for a meaningful mapping to a sound event label set, or where the metadata is a severe underrepresentation of the actual acoustic content. In some cases, labels can be too noisy for successful training of classifiers.

The alternative to the supervised learning paradigm consists of leveraging learning algorithms that do not depend on any external supervision, but that have the ability to extract supervision from the audio data. These *self-supervised* learning methods aim at *learning representations* without the need for external supervision. Absent explicit labels, the success of these methods relies on the design of *proxy* learning tasks in which pseudo-labels are generated from patterns in the data. Specifically, these methods solve proxy tasks on unlabeled data to learn mappings from input examples to useful low-dimensional representations. These representations can then be used for downstream tasks such as classification, for example where only few data, or poorly labeled data, are available. An example of the usefulness of self-supervised audio representation learning is that of low-resource languages (Kawakami et al., 2020). After learning unsupervised speech representations using abundant unlabeled speech audio, speech recognition models are trained using the learned representation on much smaller labeled datasets of low-resource languages, improving recognition accuracy. A major advantage of this self-supervision paradigm is

that much larger amounts of data without prior manual labelling or metadata can in principle be exploited—potentially unlimited data. In particular, large amounts of rich everyday sound data from Freesound, Youtube or Flickr could be leveraged, without caring about the existence of labels or the quality of user-provided metadata. Another advantage of unsupervised representation learning is that representations learned this way can be less specialized towards the often-biased human labels. Hence, they may be better suited for generalization to other tasks (Kawakami et al., 2020; Shor et al., 2020).

Self-supervision has seen major progress in computer vision (Chen et al., 2020b,c; Grill et al., 2020) and in speech recognition (Oord et al., 2018; Baevski et al., 2019, 2020). By the time this thesis started, the topic of self-supervised learning of general-purpose audio representations beyond speech had not been very explored. One of the few prior works is Lee et al. (2009), where convolutional deep belief networks are used to learn representations for speech and music, but not for general-purpose audio. One of the main challenges of this learning paradigm is how to effectively extract supervision from the audio data. This self-supervision is not limited to a specific mapping from examples to a series of pre-defined labels, but each training example could provide arguably more diverse information. However, in practice, extracting this information is not a trivial task, and the supervision derived this way tends to be relatively weak. Consequently, at the moment, self-supervised methods in sound recognition underperform their supervised counterparts, although the gap is being progressively reduced.

Self-supervision is a research direction that is beginning to have a major impact in machine listening, mainly due to the availability of very large amounts of unlabeled data. In this thesis, we have significantly contributed to consolidating this research direction in SER, exploring multiple strategies to learn general-purpose audio representations from unlabeled data (Chapter 6). By leveraging two proxy learning tasks and shedding light on new methodologies for self-supervised audio representation learning, we contribute to reducing the gap between unsupervised and supervised learning algorithms for sound event classification.

1.4 Scope and Objectives

In the previous Sections, we have described the context of this thesis and introduced the motivations for the research topics that are covered, using different learning paradigms and leveraging sources of data in different ways. A visual summary is depicted in Figure 1.1. The left side represents the scientific context of sound event classification before this thesis, where research was mostly based on supervised classification using small exhaustively-labeled datasets.

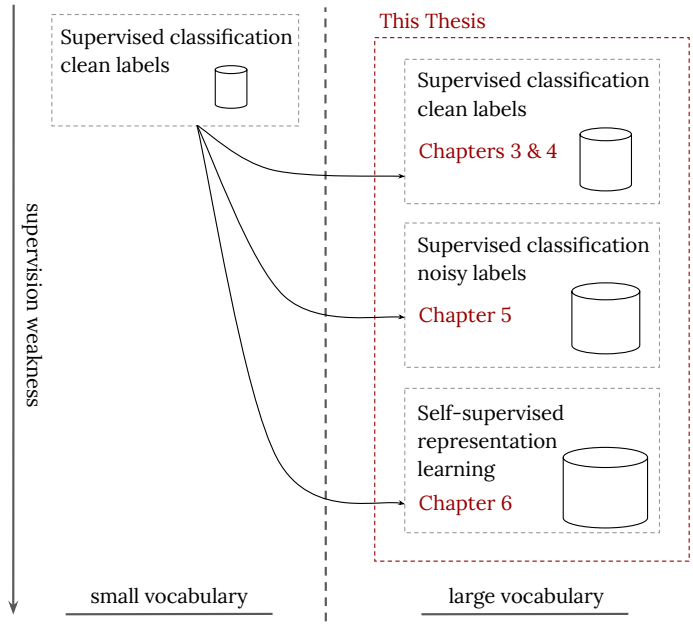

Figure 1.1: Research context of this thesis and conceptual organisation of Chapters 3 to 6 of this dissertation, according to the weakness of the supervision used. Note that the weakness of the supervision given by noisy labels depends on the amount of label noise—the Figure illustrates typical cases.

The right side represents the main research topics covered in this thesis, and how they are structured in Chapters within this dissertation. Figure 1.1 also illustrates the amount of training data typically used in each learning paradigm. In this thesis, we transition from small to large-vocabulary datasets, which already involves certain reduction in label quality. In this context, the first part of this dissertation deals with supervised classification assuming clean labels (Chapters 3 and 4). Then, we cover the topic of supervised classification from noisy labels (Chapter 5). Finally, we focus on self-supervised representation learning from unlabeled data (Chapter 6). In this way, this thesis contributes to a transition in the methodologies and approaches utilized to train models for sound event classification. A more detailed outline of this dissertation is included in Section 1.5.

In light of the above introduction, the work presented in this thesis focuses on advancing the field of SER by creating and using new data resources in novel and alternative ways. Specifically, we research on several strands of dataset creation as well as supervised and unsupervised learning in order to train high-quality large-vocabulary sound event classifiers. The main objectives of this thesis are to:

- **O1**: build an open annotated dataset of sound events of larger coverage and size than existing publicly available datasets.
- **O2**: devise a learning method for sound event classification able to improve generalization to new unseen examples.
- **O3**: develop techniques to mitigate the negative effect of label noise when training sound event classifiers on noisy labels.
- **O4**: develop methodologies for learning sound event representations in unsupervised fashion.
- **O5**: release data and source code as open resources thus fostering open and reproducible SER research.

1.5 Thesis Outline

There are seven Chapters in this dissertation. The main contributions related to the thesis objectives are described from Chapter 3 to Chapter 6, whose conceptual organisation is illustrated in Figure 1.1 according to the weakness of the supervision used for learning. A significant amount of the content in these Chapters is derived from our publications (Fonseca et al., 2018b, 2019b,a,c, 2020b,a, 2021c,b,a), which are listed in Appendix B, along with other publications by the author based on the work of this thesis.

Next, we include a brief description of the structure of this dissertation, along with the main contents and achievements that are reported in each Chapter.

In Chapter 2, we provide the relevant background to support the topics covered in this dissertation, as well as a review of the related literature. We start by describing the context of sound event recognition as a research field, introducing terminology, discussing its main characteristics, contextualizing it with other research fields, identifying its main technical challenges, and describing how sound events can be organized. Then, we present the main components of a sound event classification pipeline based on supervised learning, making emphasis on some aspects that are basis for the work described in subsequent Chapters, including data and labels, input representations, data augmentation, classifiers, loss functions, and evaluation metrics. Finally, we provide a comprehensive literature review centered around the Chapters of this thesis, namely, datasets for sound event recognition, convolutional neural networks for sound event classification, learning with noisy labels, and self-supervised learning. In this Chapter, we also provide a critical perspective in order to further define the scope of the thesis, and motivate some of the choices made along the way.

In Chapter 3, we introduce *FSD50K* (**F**ree**s**ound **D**ataset **50k**): a dataset containing 51,197 Freesound audio clips totalling over 100h of audio manually labeled using 200 classes drawn from the AudioSet Ontology. FSD50K is a dataset primarily designed for the development and evaluation of multi-label sound event classification systems, but that also allows a variety of sound event research tasks. We provide a detailed description of the FSD50K creation process tailored to the particularities of Freesound data, including challenges encountered and solutions adopted. We include a comprehensive characterization of the dataset along with discussion of limitations and key factors to allow its audio-informed usage. We also describe a set of sound event classification experiments to provide baseline systems as well as critical insight on the main factors to consider when splitting Freesound audio data for sound event recognition tasks. This Chapter is useful to researchers using FSD50K (and, in general, Freesound data for machine learning) as it allows making data-informed decisions for design choices of machine listening systems. It may also be useful for researchers working on the creation of large-vocabulary datasets. To our knowledge, this is the largest fully-open dataset of human-labeled sound events, and the second largest after AudioSet.

In Chapter 4, we propose a novel deep neural network for sound event classification. This architecture features several mechanisms aimed at increasing shift invariance in the subsampling operations, such as low-pass filters and adaptive sampling. The proposed architectural changes are evaluated on the large-vocabulary sound event classification task of the FSD50K dataset via ablation experiments. To this end, we use models of small and large capacity, and in presence of a strong regularization. Our evaluation suggests that lack of shift invariance is a problem in sound event classification, and there are benefits in addressing it. Results shows that the proposed architectural changes consistently improve sound event classification in all cases considered. We also demonstrate empirically that the proposed methods reinforce shift invariance in the network, making it more robust against time/frequency shifts in input spectrograms. The outcome is a new state-of-the-art mAP on the FSD50K classification benchmark when not using external training data.

Chapter 5 is dedicated to the study of techniques to improve performance in presence of noisy labels in sound event classification. Our first contribution is *FSDnoisy18k*, an openly-available audio dataset that supports the investigation of real label noise, including an empirical characterization of the noise. To our knowledge, no previous audio dataset has specifically provided for the study of label noise in sound event classification. Then, we explore and develop simple, efficient and network-agnostic approaches to mitigate the effect of label noise. The approaches include external regularization techniques, noise-robust loss functions, and sample rejection strategies to identify and discard potential noisy labeled examples during the learning process. Further, we address the

problem of *missing labels*, which is one of the big weaknesses of large sound event datasets. Specifically, we propose a teacher-student framework with loss masking to identify the most critical missing label candidates, and ignore their contribution during training. These approaches can be easily incorporated into existing deep learning pipelines, requiring minimal intervention and computational overhead. Evaluation of the proposed methods using FSDnoisy18k and AudioSet reveals that they show promise in increasing robustness against noisy labels, especially considering their simplicity and efficiency.

In Chapter 6, we propose multiple strategies to learn general-purpose audio representations from unlabeled data. We focus on self-supervised *contrastive* audio representation learning, where representations are learned by comparing pairs of examples selected by some semantically-correlated notion of similarity. In order to generate pairs of positive examples, we create different *views* of the original input examples. These views are created via compositions of data augmentation methods, which are one of the key elements in our learning pipelines. To our knowledge, this is the first time that some augmentation methods are used for sound event representation learning, such as automatic sound separation. After analyzing the main components of the proposed pipelines via ablation experiments, we evaluate the learned representations using multiple evaluation approaches, including linear evaluation protocol, downstream sound event classification tasks, and query-by-example retrieval. Our best outcome is an unsupervised audio representation that rivals state-of-the-art alternatives on the established shallow AudioSet classification benchmark.

At the end of each Chapter, a summary and discussion of the key results and takeaways is included. In addition, we conclude this dissertation with Chapter 7, where we present an overall summary of the thesis and our main conclusions, list our main contributions, and discuss possible future perspectives of sound event classification with different types of supervision.

In addition, this dissertation contains six Appendix sections. In Appendix A, we describe another significant contribution of this thesis, which is the co-organization of two DCASE Challenge Tasks. Appendix B lists the publications by the author during this thesis. Appendix C lists other academic contributions and merits based on the work of this thesis. In Appendix D, we list and provide links to the most relevant open data and code resources outcome of this thesis. Appendix E contains additional Figures for a better interpretation of our evaluation results. Finally, Appendix F contains the glossary of the abbreviations used in this thesis.

CHAPTER 2

Background

2.1 Introduction

This Chapter provides a description of the Sound Event Recognition field and the relevant background for the topics covered in this thesis, as well as a review of the related literature. Specifically, in Section 2.2, we introduce the concept of *sound event* and the field of *Sound Event Recognition*, discussing its main characteristics, contextualizing it with other research fields, identifying its main technical challenges, and describing how sound events can be organized, with special emphasis in the AudioSet Ontology (Gemmeke et al., 2017). We also introduce a number of relevant concepts and terminology used in this dissertation. Section 2.3 gives a brief overview of the main components of a Sound Event Classification pipeline based on supervised learning, making emphasis on some aspects that are basis for the work described in subsequent Chapters, including data and labels, input representations, data augmentation, classifiers, loss functions, and evaluation metrics. Finally, we provide a comprehensive literature review centred around the work presented in this thesis, namely, datasets for sound event recognition (Section 2.4), convolutional neural networks for sound event classification (Section 2.5), learning with noisy labels (Section 2.6), and self-supervised learning (Section 2.7). These literature reviews provide the related work for Chapters 3, 4, 5 and 6, respectively. Throughout the various Sections of this Chapter, we also provide a critical perspective in order to further define the scope of this thesis, and motivate some of the choices made along the way.

2.2 What is Sound Event Recognition?

Let us first define what we mean by *Sound Event Recognition* in the context of this thesis, starting by the concept of *sound event*. There seems not to be a commonly-agreed, concrete definition of sound event in the literature. From an acoustical perspective, broadly speaking, the term *sound event* denotes a physical phenomenon by which a sound source or a sound production mechanism produces vibrations into a medium (typically, the air) generating sound waves during a given period of time. This definition is intentionally broad in order to encompass a wide range of acoustic signals, and can be better assimilated with some examples. Sound events can be produced by humans (e.g, cough, laughter, or footsteps), animals (e.g., bird chirping, cat purring), machines (e.g., the sound of a jackhammer or a drill), domestic appliances (e.g, the sound of a kettle whistling or a vacuum cleaner), transportation modes (e.g., the sound generated by a train or a bicycle) and a very large etcetera. Note that, according to the definition above, human speech could potentially be considered a sound event; and also the sounds produced by musical instruments, such as guitars, drums or flutes. The physical objects and beings just mentioned in the examples can be denoted as *sound sources*. In addition, other sound events are produced by what we refer to as *sound production mechanisms*. This is common, for example, among sounds that occur in the nature. For instance, a thunder is the sound of lightning, which is an electrostatic discharge. Arguably, this is more the outcome of a sound production mechanism than of a physical sound source, as is the sound produced by the wind flowing through a constrained cavity. Other sound production mechanisms include, for instance, the sound of dragging a hand across a table. To finish the analysis of the definition of sound event, the time duration can vary largely among different sound events. Many sound events have a well-defined and brief duration, lasting only very few seconds. However, some sound events are even shorter and with transient behaviour, e.g., the sound of clapping hands or breaking a glass, lasting only few hundred miliseconds. In contrast, other sound events can be very long, such as the sound of rain or wind, potentially lasting minutes or more.

Beyond the above examples, we want to stress that the definition of sound source or production mechanism can be ambiguous, and the delimitation between them can be blurry. For example, one could argue that a bicycle is a sound source, but the sound being generated could be coming from the bicycle's bell, or from the gears of its chain, or from the sound of the brakes' material grazing the tire (which could also be considered as a sound production mechanism). Finding these limits and establishing thorough definitions is not a trivial task. Regardless of the mechanism that produces them, sound events are typically associated with textual *labels*, so that humans can more easily categorize and

understand them. Following the bicycle example, we could have labels such as *Bicycle bell*, *Gears*, or *Tire squeal*.[6]

While speech and music could be understood as just other types of sound events, the notion of sound events is typically associated to the terms of *environmental sounds* or *everyday sounds*. *Everyday sounds* is typically used to denote the non-speech and non-music sounds that occur naturally in the environment surrounding us (Gygi, 2001). These terms are often used broadly to describe the sounds that humans encounter in everyday life (excluding speech and music), and also other sounds that are perhaps less frequent, but still familiar to an average listener. For example, some people do not listen to the sounds of sea waves or a frog croaking on a regular basis, but these sounds are still familiar to many, e.g., due to cultural reasons. In this thesis, we will use both terms (environmental sounds and everyday sounds) interchangeably.

Once defined sound event, in this thesis we shall use the expression sound event *recognition* (SER) broadly to encompass two high-level tasks: sound event classification and sound event detection. Sound event *classification* or *tagging* (SET) is a task requiring to identify what sound event classes are present in an audio clip, regardless of start time and end time. Some works in the literature refer to *classification* when the task goal is to recognize one single sound event class at a time, whereas *tagging* is reserved for the task where multiple classes can be recognized simultaneously. However, in this thesis, we shall use both terms interchangeably, without restrictions as to how many classes can be recognized simultaneously. Sound event *detection* (SED) is a task requiring to localize and identify sound events in an audio clip with start and end times (sometimes called onset and offset). We advocate for this relationship of SER = SET + SED following the analogous treatment done in computer vision, where object *recognition* encompasses image *classification* and object *detection* (Russakovsky et al., 2015). Sound Event Recognition can be framed within the field of *machine listening* (sometimes also called *machine hearing* (Lyon, 2017)), aimed at studying computational methods for the automatic analysis and understanding of sound (Wang, 2010).

In this thesis, we focus on Sound Event *Classification* for three reasons. First, we opt to annotate the datasets created in this thesis using *weak labels*. Weak labels refer to labels which are provided at the clip-level, expressing the presence of a sound event somewhere in the clip, but without specifying temporal location. The motivation to use weak labels is that gathering them is simpler, less time consuming, and less ambiguous than determining events' onset/offset using timestamps. Weak labels are suitable for classification but not for detection, where labels featuring onset and offset for the sound events are required

[6] These labels are selected from the AudioSet Ontology. See https://research.google.com/audioset/ontology/index.html.

(at least for evaluation purposes). The labels that provide temporal activity of sound events, which are essential for evaluation of detection systems, are usually regarded as *strong labels*. Second, as mentioned in Chapter 1, one of the focus of this thesis is large-vocabulary audio data and classifiers. The largest resource of large-vocabulary audio data is AudioSet, which also features weak labels.[7] Third, the core principle of methods for the tasks of classification and detection is very similar, with the main differences including how the data is fed to the classifier/detector, and how the output predictions are post-processed. A sound event classifier or tagger can be used as a sound event detector by processing incoming audio in consecutive time slices, sufficiently overlapped to produce events' activity with a desired temporal resolution.

2.2.1 Relation of Environmental Sounds to Other Audio Signals

One way to better assimilate the properties of environmental sounds is by contextualizing them with the other two main audio signals that we listen to in our daily lives: speech and music. A broad comparison of the main aspects of speech, music and environmental sounds is illustrated in Table 2.1, which has been elaborated based on Gygi (2001); Yamakawa et al. (2010); Heittola (2021). Table 2.1 shows the difficulty and uncertainty in the characterization of environmental sounds when compared to speech and music. In general, it can be seen that speech and music signals have better defined properties and structure than environmental sounds. Speech signal is constrained to the sounds that can be generated by the vocal tract and tongue, which provide speech with a well-defined structure that can be exploited for its identification. Similarly, although perhaps to a lesser extent, many musical genres of western music present certain hierarchy (e.g., notes, chords) and temporal structure. In contrast, environmental sounds are much more diverse. Their properties are more uncertain as they highly depend on the sound sources or production mechanisms, which are more heterogeneous. This fact hampers the usage of assumptions about their structure for their automatic recognition.

[7] Hershey et al. (2021) released labels with a finer temporal resolution for a small subset of AudioSet shortly before the end of this thesis. See https://research.google.com/audioset/download_strong.html.

Table 2.1: Comparison of some characteristics of speech, music and environmental sounds, based on Gygi (2001); Yamakawa et al. (2010); Heittola (2021).

Characteristic	Speech	Music	Environmental Sounds
Possible sources	human vocal tract	instruments (analogic or synthethic)	naturally-occuring sound sources or sound production mechanisms
No. of sources	finite	finite (though potentially large)	potentially infinite
No. of classes	No. of phonemes	No. of notes, chords, genres, etc.	arbitrarily high
Harmonic structure	largely harmonic; some inharmonic parts	largely harmonic; some inharmonic parts	unknown proportion of harmonic to inharmonic parts
Bandwidth	narrow	wider	unknown proportion of narrow and wide
Window length	short	longer	unknown proportion of short and long
Temporal structure	more steady-state than dynamic	mix of steady-state and transient, strong periodicity	unknown proportion of steady-state and dynamic, variable periodicity

2.2.2 Relation of Sound Event Recognition to Other Fields

SER has only recently gained attention from the research community. However, the analysis of speech and music have been subject of study for a longer period of time. One could argue that SER presents parallels with other tasks from the fields of speech and music analysis (Virtanen et al., 2018). In particular, SER could be related to the task of speaker recognition, where the goal is to recognize the identity of the people talking in an audio stream. Likewise, SER could also be related to some tasks in the field of Music Information Retrieval (MIR), such as musical genre recognition (sometimes called *auto-tagging* as in Won et al. (2020b)), where the goal is to identify the genre(s) in a musical track; or to the task of instrument recognition, where the goal is to recognize the various instruments present in a recording (Lostanlen et al., 2018).

Due to the maturity of speech and music analysis, and the similarities of some tasks with SER, some techniques have been borrowed from these fields. For example, the popular Mel-frequency Cepstral Coefficients (MFCCs), originally proposed for speech processing (Davis & Mermelstein, 1980), and subsequently used in MIR (Tzanetakis & Cook, 2002), have been also used for SER (Lee & Ellis, 2009). In a similar fashion, the Convolutional Recurrent Neural Network (CRNN) type of architecture was initially applied in the context of speech recognition (Sainath et al., 2015), then ported to music genre classification (Choi et al., 2017) and SER (Cakır et al., 2017).

Another field from which SER is borrowing methods and techniques is the field of computer vision. It has been proven that some methods from computer vision work well for SER. For example, the current state-of-the-art for AudioSet classification is achieved by the Audio Spectrogram Transformer (Gong et al., 2021a). This architecture is largely based on the Vision Transformer (ViT) (Dosovitskiy et al., 2021), which is a Transformer architecture for vision tasks. In particular, their differences are mostly related to the different input formats of images and spectrograms, as well as the final classification layer as the classification tasks are different. Besides the architecture, Gong et al. (2021a) also rely on transfer learning from ImageNet (Deng et al., 2009) to improve recognition performance on AudioSet. Using ImageNet-pretrained weights to initialize the training of audio classification models has been previously done (Palanisamy et al., 2020; Guzhov et al., 2021), obtaining performance boosts also in other datasets such as ESC-50 or UrbanSound8K. The above works tend to use default architectures from vision to take advantage of some pre-trained models that are already available from PyTorch or Tensorflow repositories (e.g., Inception (Szegedy et al., 2015), ResNet (He et al., 2016a), DenseNet (Huang et al., 2017), etc.).

Another prominent work in large-scale SER is Kong et al. (2020a), which held the state-of-the-art on AudioSet before the work of Gong et al. (2021b) and

Gong et al. (2021a). This work proposes Wavegram-Logmel-CNN, a novel architecture featuring two convolution branches to enhance the exploitation of the audio signal—one to process spectrograms and another to process raw waveforms. This model achieves a mean average precision (mAP) of 0.439, slightly outperforming a default ResNet-34 (He et al., 2016a) that obtains 0.434 mAP. It can be seen that the reported difference is not large, also considering that Wavegram-Logmel-CNN employs over 7M additional parameters (Kong et al., 2020a). It is also interesting to analyze this difference from an industry perspective. For production systems, sometimes it is preferred to use default and long-tested architectures, even at the expense of some recognition performance. This would probably favour the adoption of architectures such as ResNet in this case.

In light of this discussion, there is evidence that some successful SER methods are largely inspired from computer vision, or adapt techniques from computer vision to the audio domain. In this thesis, sometimes we also follow this trend, borrowing methodologies and techniques from computer vision that we adapt to the SER problem, or that we tailor to problems existing in the audio data.

2.2.3 Technical Challenges in Sound Event Recognition

In light of the discussion done so far about the different aspects of sound events, the following technical challenges can be identified in SER. Sound events have characteristic spectro-temporal patterns resulting from their physical production that, in principle, can be used to identify them. As a result of the large variety of potential sound sources there is a wide heterogeneity of the patterns to be recognized. This problem is aggravated as we increase the set of target sound categories from tens to several hundreds, causing high *inter-class variation*. Also, as discussed in Section 2.2.1, the spectral/temporal structure of these sounds is less clear than in speech or music, which hampers the usage of assumptions for their automatic recognition. At the same time, it can happen that the patterns in some of the classes to be recognized are very similar to those of other related classes, especially as the level of specificity in the classes increases, e.g., if we are trying to distinguish two types of wind instruments or two types of bells. Another particular aspect of sound events is what is sometimes called a high *intra-class variation*, which means that sound events belonging to the same class can be rather different already. A typical example is the vocalizations produced by different dog breeds, which may be substantially different, e.g., comparing a *Chihuahua* with a *Pyrenean Mastiff*.

Further, sound events often present highly varying acoustic characteristics due to factors such as different acoustic conditions in the environment, or the equipment used to capture the sound. For example, some sounds exist in quiet environments almost in isolation, whereas in other frequent cases mul-

tiple sounds co-exist, sometimes overlapping with each other. The degree of *polyphony* complicates the identification of the co-occurring sounds. In addition, different types and intensities of acoustic background noise or non-target audio material can be present in the audio recording. These non-target content can mask the target sounds to some extent, especially in situations where microphones are further away from the target source, which is more frequent when recording environmental sounds than in other domains such as speech applications. Besides, there can also be other types of noise and/or distortions caused during the process of capturing sound. The recording equipment and recording techniques can vary to a high extent from one case to another, which can affect audio quality. This variation can be especially notorious when the audio data is collaboratively contributed by thousands (or millions) of users in platforms such as Freesound or YouTube. Again, all this implies a high degree of variation in the sound patterns that can be encountered in real life.

In summary, all these issues make the SER problem particularly challenging and can affect the performance of sound event recognizers. In particular, the large diversity of sound events becomes a challenge for models' generalization. It can be expected that sound event recognizers operating in real life will encounter sound event examples whose patterns differ to some degree from those seen during training, or that have different acoustic characteristics. Hence, learning representations able to generalize well in these cases is a key factor in SER. Possible solutions to this challenge include increasing the amount and variety of training data, as well as coming up with architectures and methodologies to improve generalization. These are relevant topics in this thesis, which we cover in Chapters 3 and 4 respectively.

Finally, other challenges that apply to SER relate to the fact it is heavily based on machine learning methodologies. Therefore, issues of amount, quality and diversity of training data are important. In this thesis, we put the emphasis on the *supervision* of data. As mentioned in Chapter 1, manually annotating sound events is a costly and difficult task, especially when using a vocabulary of hundreds of categories. This limits the size of datasets and can create issues of label noise. To overcome these limitations, in Chapter 5 and 6 we address the problems of learning with noisy labels, and finding alternative ways to fetch supervision via self-supervised learning.

2.2.4 Organization of Sound Events

To conclude this Section, we briefly discuss how sound events can be organized, with special emphasis in the AudioSet Ontology (Gemmeke et al., 2017), which is used to organize the proposed FSD50K in Chapter 3. Categorization of everyday sounds into meaningful sound categories is a natural instinct of humans to make sense of the diversity of sounds surrounding us. A compre-

hensive review of everyday sound categorization can be found in Guastavino (2018). In this thesis, we will use the term *sound category* and *sound class* interchangeably. Often, everyday sounds are organized using *taxonomies* or *ontologies*, with the aim of bringing a (hierarchical) structure to the wide variety of sounds, based on a given criteria and/or establishing relations among the sound categories and concepts. There are several criteria for the organization of everyday sounds. For example, Schaeffer (1966) suggested three viewpoints to describe sound: *i) causal listening*, related to the identification of physical sound sources and production mechanisms; *ii) semantic listening*, focused on describing the message that is conveyed in the sound, e.g., hearing a fire alarm carries a meaning; and *iii) reduced listening*, related to the inherent characteristics of a sound signal regardless of its cause and meaning, sometimes using morphological sound criteria (Peeters & Deruty, 2010).

Several sound taxonomies have been proposed in the literature that can be more or less associated with the previous viewpoints. In this thesis, we are interested in taxonomies related with *causal listening*, examples of which include Gaver (1993); Salamon et al. (2014); Gemmeke et al. (2017). Gaver (1993) proposes a taxonomy following principles of ecological acoustics and taking into account factors of the physical production of environmental sounds, including interacting materials (solids, gasses and liquids), sound production mechanisms (e.g., wind, rain) and interactions (e.g., scraping, rolling). Salamon et al. (2014) propose a taxonomy for the domain of urban sound research, also focused on sound sources (e.g., engine accelerating, dog bark) and sound production mechanisms (e.g., explosion, wind). Google's AudioSet Ontology, which we describe in detail in the next Section, aims to be of general-purpose and thus it covers a wide range of sound classes (Gemmeke et al., 2017).

Designing taxonomies to organize everyday sounds can be useful when constructing audio datasets, such as UrbanSound8K (Salamon et al., 2014) and AudioSet (Gemmeke et al., 2017). In contrast, in other cases researchers do not make use of an established taxonomy. Instead, they create datasets with a vocabulary of classes based on a more pragmatic approach, e.g., availability of data or relevance for a given research problem or end application (Piczak, 2015b; Turpault et al., 2019).

2.2.4.1 AudioSet Ontology

The AudioSet Ontology[8] consists of 632 sound event classes arranged in a hierarchy with a maximum depth of six levels (Gemmeke et al., 2017). The set of classes covers a very diverse range of everyday sounds, ranging from hu-

[8]It can be explored at https://research.google.com/audioset/ontology/index.html. Sometimes we shall refer to the AudioSet Ontology simply as the ontology.

man and animal sounds, to natural, musical or miscellaneous sounds. Within these main sound families, the content covered by the ontology includes several facets. The predominant classes correspond to sound events produced by physical sound sources, but there are also some generated by sound production mechanisms (e.g., deformation or impact of materials). Then, there is a miscellaneous of classes that, strictly, do not correspond to sound events, such as acoustic scenes, categories describing (perceptual) attributes of sound, mathematical signals or abstract classes. The ontology is provided as a list of 632 objects,[9] each of them including fields such as a Knowledge Graph Machine ID or a textual description, among others. Note that the AudioSet vocabulary is a subset of 527 classes drawn from the ontology, with the remaining classes being blacklisted or excluded as they are considered abstract.

We clarify next some basic (albeit relevant) ontology-related terms used in this thesis. We shall refer to the 632 classes in the ontology as *nodes* (either *leaf* nodes when they are located at the very bottom of the hierarchy, or *intermediate* nodes otherwise). We shall also use the ontological terms *children* and *parents*, as widely used in ontology-related genome research (Jantzen et al., 2011). Note that, by definition, leaf nodes do not have children nodes, while the intermediate nodes do. Similarly, given a node, we refer to all the parent nodes connecting it to the root of the ontology as *ancestors*. As an example, let us consider the hierarchical path: *Root* → *Natural sounds* → *Thunderstorm* → *Thunder*. In this path, *Thunder* is a leaf node; *Natural sounds* and *Thunderstorm* are both intermediate nodes; *Thunderstorm* is child of *Natural sounds* and parent of *Thunder*; and *Thunderstorm* and *Natural sounds* are all the ancestors of *Thunder*. As in the examples above, we use *italic format* when we refer to the textual label of a sound class. For example, the sound of a car passing by in the AudioSet Ontology is labeled as *Car passing by*.

In this thesis, we use the term *general-purpose* to denote a sound vocabulary, or a dataset (and the recognizers that can be trained on it) whose coverage is diverse enough such that it is not tied to a specific audio domain. A general-purpose dataset encompasses a considerable and diverse amount of everyday sounds, typically including several audio domains, as well as several tens or hundreds of sound classes. The main example of a general-purpose vocabulary is the AudioSet Ontology (Gemmeke et al., 2017), counting with 632 sound classes. The main examples of general-purpose datasets are AudioSet (Gemmeke et al., 2017) and FSD50K (Fonseca et al., 2020a), with 527 and 200 classes, respectively, covering human, natural, domestic and urban sounds to name a few of the constituent domains.

[9] https://github.com/audioset/ontology

2.3 Sound Event Classification as a Supervised Learning Problem

In this Section, we give a brief overview of the main components of a Sound Event *Classification* pipeline based on supervised learning, which is the most commonly-adopted approach. We focus mainly on the aspects of some components that are basis for the work described in subsequent Chapters. The interested reader can find a more comprehensive overview of the machine learning approach for sound event analysis in Heittola et al. (2018) and McFee (2018). For a more general perspective not tied to SER of the methods and concepts discussed, the reader is referred to Goodfellow et al. (2016). Section 2.3.1 is based on the aforementioned references.

Broadly speaking, in a sound classification pipeline, the goal of the training stage is to learn the parameters of a classifier, such that it is able to classify input examples into one or more sound classes within a predefined vocabulary. At the beginning of this thesis, in SER we were witnessing a paradigm shift from *feature engineering* approaches to techniques based on *learning representations* from data, similarly to the transition experienced in areas like computer vision or speech recognition.

In the *feature engineering* approach, pre-designed low-level features are extracted from the audio signal and input to a classifier. Examples of typical handcrafted features in audio-related tasks include cepstral features, e.g., MFCCs (Lee & Ellis, 2009), as well as other low-level features computed from the time or frequency domain (e.g., zero-crossing rate or spectral centroid) (Eronen et al., 2006). Typical examples of classifiers include Gaussian Mixture Models (Mesaros et al., 2010) and Support Vector Machines (Foggia et al., 2015). The feature engineering approach relies heavily on the capacity of the pre-designed features to capture relevant information from the signal, which in turn may need significant expertise and effort. The main limitations of this approach are the lack of efficiency and sustainability given the high diversity of particular cases encountered in real world.

Having enabled significant research breakthroughs in other recognition tasks, the data-driven approaches based on *learning representations*—especially deep learning—were rapidly spread across the SER community. In this case, the system is able to learn internal powerful representations from a simpler low-level representation at the input (e.g., mel spectrogram), and the two stages of the feature engineering approach (feature extraction and classification) are optimized jointly. In this thesis, we adopt this trend. All the classifiers used are neural networks of various depths and widths, which will be sometimes referred to simply as *networks* or *models*.

The sound event classification task can be formulated as two different problems. In a *multi-class classification* problem, the goal is to predict only one class label at a time. In this case, examples are labeled with only one class label from the many in the vocabulary, as done in some widely-used datasets such as ESC-50 (Piczak, 2015b) and UrbanSound8K (Salamon et al., 2014). A typical example of a related problem of the same type is *acoustic scene classification*, where each recording is made at one location, that is, classes are mutually exclusive. In a *multi-label classification* problem, however, the goal is to be able to predict multiple class labels simultaneously. In this case, examples are labeled with *one or more* class labels from the vocabulary, as done in AudioSet (Gemmeke et al., 2017) or in the main dataset introduced in this thesis, FSD50K (Chapter 3). The multi-label setting is the most representative for SER as sound events can overlap naturally. It must be noted that variations of these terms can also be found in the literature. For example, Heittola et al. (2018) refer to the just defined multi-class problem as *single-label* problem.

2.3.1 Overview of the Supervised Learning Pipeline

Hereafter we shall focus on sound event *classification* (also typically named *tagging*), and not on detection. To refer to the task of classification, we shall use the acronym Sound Event Tagging (SET) as it is probably more popular than Sound Event Classification (SEC) in the community. Occasionally, we shall still use the acronym SER whenever we explicitly want to refer to both classification *and* detection simultaneously.

In supervised learning, let us consider a dataset $\mathcal{D}$ where the training examples are given by input-output pairs $(\boldsymbol{x}, \boldsymbol{y})$ where $\boldsymbol{x}$ is an input example, and $\boldsymbol{y}$ is the corresponding output target vector representing the labels associated to $\boldsymbol{x}$. Deep networks are usually fed with low-level representations (e.g., some variant of spectrograms or directly raw waveforms), rather than with the acoustic features most commonly used in the feature engineering paradigm. Therefore, we will also refer to the input examples $\boldsymbol{x}$ as *input representation*. In particular, in the publications done during this thesis, all the input representations are 2D time-frequency representations of audio, given by $\boldsymbol{x} \in \mathbb{R}^{T \times F}$ with T time frames and F frequency bands. The output target vector $\boldsymbol{y}$ representing the reference labels is usually expressed in binary format $\boldsymbol{y} \in \{0,1\}^C$ for a task using a vocabulary $V = \{c_i\}_{i=1}^{C}$ of C classes. In $\boldsymbol{y}$, 1 represents that the class c_i is present in the example $\boldsymbol{x}$, and 0 represents otherwise. In multi-class problems, labels are encoded as *one-hot* vectors, where only one element equals 1. In multi-label problems, labels are encoded as *multi-hot* vectors, where more than one element can be 1.

Our goal is to train a deep network $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$ that defines a mapping $f : \mathbb{R}^{T \times F} \rightarrow \mathbb{R}^{C}$ from the input representation $\boldsymbol{x}$ to the vectors $\boldsymbol{y}$ representing the target labels. While training, the *goodness* of this mapping is estimated through a *loss function* $\mathcal{L}(\boldsymbol{p}, \boldsymbol{y})$, which expresses the divergence between the network predictions $\boldsymbol{p} \in \mathbb{R}^{C}$ and the reference labels $\boldsymbol{y}$. The learning problem can be generally seen as minimizing $\mathcal{L}$ over the choice of the network parameters $\boldsymbol{\theta}$. The loss function $\mathcal{L}$ is minimized through an optimization algorithm (e.g., some variant of *gradient descent*). The optimization algorithm is an iterative process in which: first, gradients of the loss are computed with respect to the network parameters $\boldsymbol{\theta}$, and then the network parameters are updated using the computed gradients and a learning rate. By repeating this process iteratively, as learning progresses the sought mapping from $\boldsymbol{x}$ to $\boldsymbol{y}$ is gradually improved and the loss function minimized.

In practice, one learning iteration (or learning *step*) follows the next procedure. First, a small batch with N training examples is constructed with the form $Z = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$. Second, the network $f_{\boldsymbol{\theta}}$ is fed with the input examples within the batch, and a forward pass is carried out, obtaining the corresponding output prediction probabilities for all examples in the batch $\{\boldsymbol{p}_i\}_{i=1}^{N}$. Then, the loss $\mathcal{L}$ is computed by comparing predictions $\{\boldsymbol{p}_i\}_{i=1}^{N}$ and target labels $\{\boldsymbol{y}_i\}_{i=1}^{N}$, according to some differentiable mathematical expression. The choice of loss function depends on the task being performed as well as potentially other factors such as the quality of the labels, as we will see in Chapter 5. Finally, gradients of the loss at each individual parameter of the network are computed via the *back-propagation* algorithm, and the network parameters $\boldsymbol{\theta}$ are updated accordingly in the negative direction of the gradient. This procedure is repeated iteratively until some criteria is met. Typical stopping criteria include maximizing an evaluation metric on a held-out validation set or reaching a minimum loss over a period of time. The training time is usually measured in *epoch*, which represent full passes over the entire dataset $\mathcal{D}$ such that every training example is seen once by the network. Nevertheless, when datasets are very large (e.g., AudioSet) it is also common to measure training time in number of iterations, that is, the total number of batches processed by the network. At the end of the training, the network has learned a mapping from input examples to labels that allows to discriminate examples from different classes with a certain level of proficiency. The trained model can then be used in the *testing* or *recognition* phase to predict sound event classes on unseen examples.

In addition to outlining how a deep sound event classifier can be trained in supervised fashion, we have identified the main components and concepts involved in a typical SET pipeline, which are illustrated in Figure 2.1. The input example $\boldsymbol{x}$ is transformed into a time-frequency input representation via a feature extraction block. Note that we have added a *data augmentation* block as several data augmentation techniques are used in this thesis. The network

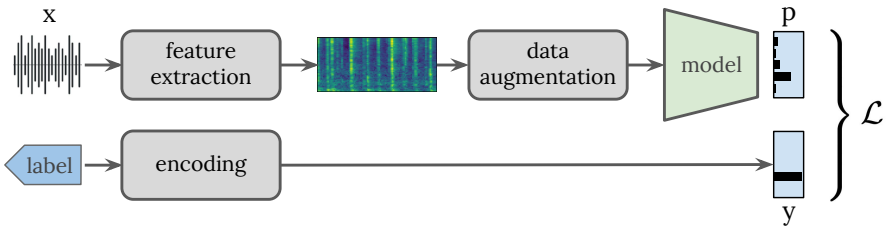

Figure 2.1: Overview of a basic supervised training pipeline for sound event classification. A multi-class classification case is illustrated, where only one label is active at a time.

output predictions $\boldsymbol{p}$ are compared with the target vector $\boldsymbol{y}$ representing the label(s) using a loss function $\mathcal{L}$. Next, we briefly discuss some relevant aspects of these components and concepts that will be helpful for the subsequent Chapters.

2.3.2 Data and Labels

In the current data-driven paradigm for sound classification, audio data and labels are one of the most important factors for successful supervised learning. Arguably, a sound classifier will be as proficient as the audio data and the supervision it is trained on. In Section 2.4 we provide a literature review of the most relevant datasets for SET and Sound Event Detection (SED). In this Section, we discuss some aspects of weak labels for SET.

To allow supervised training and for and systems' evaluation, audio data must be accompanied by corresponding labels describing the sound sources contained in the audio clips. Ideally, in the context of SET, these labels should represent a correct and complete transcription of the audio material. Let us define what we understand as *correct* and *complete* labels for SET in the context of this thesis. Given a dataset of sound events annotated with a predefined vocabulary, a label accompanying an audio clip is *correct* when it accurately and unambiguously identifies or describes a corresponding sound source present in the clip. To some degree, this definition is subject to the choice of vocabulary as, occasionally, the class labels in the vocabulary can fail to describe certain nuances of the audio material.

The notion of *complete labels* relates to a set of labels that completely identify the entirety of some acoustic material according to a given criteria. We can establish two definitions of complete labels for a given audio clip: a *strict* definition and a *vocabulary-based* definition. In the *strict* definition, we define *complete labels* as the ideal set of labels that identify *all* the acoustic material

in the audio clip, regardless of the vocabulary used in the dataset. When not all the acoustic material is identified, the label set is *incomplete*, and the unlabeled material can correspond to classes that exist in the vocabulary ("in-vocabulary" or IV), or to classes that are not covered in the vocabulary ("out-of-vocabulary" or OOV). In the *vocabulary-based* definition, we define *complete labels* as the set of labels that identify *all the target* sound sources in the audio clip, that is, all the sources that are considered in the vocabulary. In this case, the label set is regarded as *incomplete* only when there is unlabeled acoustic material that corresponds to classes existing in the vocabulary. Either one definition or another can be more convenient depending on the analysis to be conducted or the classification task. For example, from the widely-adopted perspective of annotating a dataset with a predefined vocabulary, the vocabulary-based definition makes more sense. The goal in this case is to obtain a *correct and complete* transcription using the class labels of the vocabulary, such that the only audio material that remains unlabeled, if any, is out of vocabulary (which is often downplayed).

However, the reality can differ significantly from a correct and complete transcription due to many reasons. If the annotations are done manually, causes of errors include the difficulty of annotating many classes of different nature, sometimes with ambiguous sounds or suboptimal vocabularies, as well as other flaws in the annotation process. If the annotations are done using automatic methods (e.g., using a pre-trained classifier) the quality of the labels will depend on how competent the model is, which can vary substantially depending on the class. As a result, incorrect labels and incomplete label sets are commonplace.

In particular, incomplete label sets are relatively frequent in recordings of everyday sounds as it is very tedious to exhaustively annotate every sound event present in the recording using a vocabulary of tens or hundreds of classes. Further, as we will discuss in Chapters 3 and 5, to ease the process of large-vocabulary annotation, sometimes automatic methods are utilized to propose candidate labels. Unfortunately, automatic methods often fail to nominate some sound events. For these reasons, a relatively frequent case is that of *correct but incomplete* label sets, where some of the sound sources are correctly labeled, while other sources that are identifiable in the vocabulary remain unlabeled. For example, let us assume that a clip to be labeled with the AudioSet Ontology (Gemmeke et al., 2017) contains sounds of a dog barking, a bird tweeting and human speech. If only the former two are specified by labels, but human speech remains unlabeled, the label set is correct but incomplete as human speech also belongs to the vocabulary. This is a real example occurring, for instance, in the development set of FSD50K, as we will see in in Chapter 3.

The labels accompanying the input examples in a dataset are sometimes called *ground truth labels* (Cartwright et al., 2019c) or *reference annotations* (Mesaros

et al., 2018b). The term *ground truth* has been traditionally used in the literature and it is popular among some DCASE Challenge Tasks. However, it can be somewhat misleading because labels in SET can be inherently noisy, as mentioned. In this thesis, for simplicity we will use *ground truth* labels, *reference annotations*, or often simply *labels* interchangeably, knowing that *our* ground truth can be noisy and hence it is not to be fully trusted.

2.3.3 Input Representations

Deep neural networks for audio are usually fed with a low- or mid-level input representation. For SER, these input representations are usually a 2D time-frequency representation in $\mathbb{R}^{T \times F}$ with T time frames and F frequency bands, or directly the raw audio waveform in $\mathbb{R}^{T}$ with T samples in time. The most widespread choice is to use time-frequency representations, especially some kind of mel spectrogram, after their extensive usage for speech and music. Mel spectrograms are computed by applying a filter bank with triangular frequency-domain filters distributed along the mel-scale to the short-time Fourier transform (Davis & Mermelstein, 1980). Besides mel spectrograms, there exist other representations that, in principle, could be used to feed deep sound classifiers such cochleagrams (Wang & Brown, 2006), gammatone-like spectrograms (Ellis, 2009), or scattering transform (Andén & Mallat, 2014). However, their usage is limited in comparison to that of mel spectrograms.

While most of the works found in the literature opt for 2D time-frequency input representations, the advent of deep learning has recently favored systems that learn representations directly from the audio waveform (sometimes denoted as *end-to-end* learning). When raw waveform is utilized in conjunction with CNNs, the network typically employs 1D convolutions, unlike in the previous case where the choice of 2D input representations determines the usage of 2D convolutions. One of the main advantages of the 1D processing is to avoid the need for handcrafting input representations such as mel spectrograms, which could be considered a manual shrinkage of a solution space. However, these approaches tend to perform worse than other 2D counterparts in datasets such as UrbanSounds8K (Tokozume et al., 2018; Abdoli et al., 2019), or AudioSet, as shown in Kong et al. (2020a). Further, processing raw waveforms can slow down training due to the higher dimensionality of waveforms compared to the more compact 2D representations.

Finally, a third approach sitting between the two aforementioned trends has recently arisen. In this hybrid approach, networks are fed with raw waveforms, and then a trainable front-end within the network learns a 2D representation in a data-driven fashion, optimized jointly with the rest of the network. This learned 2D representation is subsequently processed by a 2D network back-end. In other words, trainable front-ends aim at producing a substitute for

handcrafted 2D representations such as mel spectrograms. Examples of this approach include Park & Yoo (2020) and Won et al. (2020a), both reporting improvements with respect to mel spectrograms, but using datasets of a limited coverage where the diversity of sounds may not be large (10 and 17 classes, respectively). A more recent work adopting this approach for large-vocabulary SET is Zeghidour et al. (2021). The proposed audio frontend, named LEAF, learns several operations of audio feature extraction including filtering, pooling, compression and normalization. However, evaluated on the AudioSet classification task, only small improvements are reported with respect to using conventional mel spectrograms. In this thesis, preliminary experiments were carried out using Gammatone-like spectrograms (Ellis, 2009), LEAF (Zeghidour et al., 2021) and the approach of Won et al. (2020a). Evaluating these approaches on FSD50K, gains over mel spectrograms were not observed.

2.3.4 Data Augmentation

Data augmentation is a technique (or set of techniques) aimed at artificially increasing the size and variety of training data by generating modified versions of existing data (Shorten & Khoshgoftaar, 2019). This is carried out by applying transformations to training examples in order to obtain new examples. In supervised learning, the goal is usually to adopt semantics-preserving transformations such that new examples belong to the same class as the original example (Goodfellow et al., 2016). The benefits of data augmentation include improved generalization by seeing during training a series of more diverse examples that may be encountered in the testing phase. By training on these modified examples the model becomes more invariant to the adopted transformations. In addition, data augmentation adds regularization, which helps to reduce overfitting and, in turn, improve generalization.

A variety of data augmentation techniques have been proposed in the sound recognition literature, each of which applies different transformations. Some of them are based on classic audio processing transformations such as background noise addition or dynamic compression, as well as audio effects such as time stretching or pitch shifting (Salamon & Bello, 2017). Other techniques are inspired from data augmentation methods in computer vision, such as *SpecAugment* (Park et al., 2019), which is partly based on the popular *Cutout* (DeVries & Taylor, 2017). Unlike the mentioned transformations, which consist of modifying only one input example at a time, the idea of mixing different sound examples into a new example has been increasingly popular (Zhang et al., 2018; Tokozume et al., 2018). Next, we introduce the two main data augmentation methods used in this thesis, based on mixing sounds and SpecAugment.

2.3.4.1 Mixing Sounds

While the idea of mixing sounds is simple and allows room for a number of variants and implementations, there are two main methods published in the literature, both published at the 2018 "International Conference on Learning Representations". These methods are *mixup* and Between-Class learning.

Mixup. Mixup is typically understood as a data augmentation technique that acts as a regularizer by favoring linear behavior in-between training examples, encouraging networks to predict less confidently on linear interpolations of examples (Zhang et al., 2018). Specifically, mixup augments the training distribution by creating virtual examples under the assumption that linear interpolations in the feature space correspond to linear interpolations in the label space. This is expressed by

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j, \end{aligned} \tag{2.1}$$

where x_i and x_j are two spectrograms from the training data, and y_i and y_j are their corresponding binary encoded vectors. As proposed in Zhang et al. (2018), we sample λ from a beta distribution $\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$. Since $\lambda \in [0,1]$, we have convex combinations of the input spectrograms, with the hyperparameter α controlling the strength of the interpolation. In principle, mixup seems a reasonable regularization/augmentation strategy for sound events (more than for image classification) as it roughly simulates the general setting of two sound sources (x_i,x_j) present in an acoustic scene. The sound produced by each source will have a certain saliency, i.e., a sound pressure level (controlled by λ in Equation 2.1), due to the attenuation produced by their different source-microphone distances.

Between-Class learning. The strategy proposed by Tokozume et al. (2018) for Between-Class learning (*BC learning*) is similar to that of mixup, but follows a somewhat more thorough process. The main differences of BC learning with respect to mixup are: *i)* the mixing ratio for the training examples, r, is drawn from a uniform distribution $r \sim \mathcal{U}(0,1)$ instead of from a beta distribution; *ii)* the actual mixing ratio applied to the instances is not directly r, but $p = f(r, G_1, G_2)$, where G_1 and G_2 are the sound pressure levels of the audio examples to be mixed, computed using A-weighting to account for human auditory perception; *iii)* the Kullback–Leibler divergence between labels and predictions is used as loss function, instead of the typical cross-entropy loss. Among the above, perhaps the most important distinction with respect to mixup is the consideration of the difference in sound pressure level, which authors claim to be important for BC learning. In addition to implying an

added complexity, results backing this claim are obtained using ESC-50—a multi-class dataset featuring less than 3 hours of audio—which may not be fully representative of large-scale multi-label settings. Further, several state-of-the-art works using AudioSet have turned to mixup augmentation instead of BC learning (Kong et al., 2020a; Gong et al., 2021b). For simplicity, in this thesis we use the mixup-based strategy.

It must be noted that mixup and BC-learning can be interpreted from two viewpoints. On the one hand, they can be seen as *data augmentation* techniques in the sense that new training examples are being generated, thus increasing the variability of the train set. On the other hand, they can be interpreted as *learning methods* because the network is tasked to predict new labels that are mixtures of the labels from the original examples. This is different from most data augmentation techniques which only modify the input examples $\boldsymbol{x}$ (but not the labels $\boldsymbol{y}$) in a dataset $\mathcal{D}$ of input-output pairs $(\boldsymbol{x}, \boldsymbol{y})$.

In this thesis, we leverage the concept of mixing sounds when training classifiers in a supervised fashion, using both clean labels (Chapter 4) and noisy labels (Chapter 5). We also use it in the context of self-supervised learning in order to create different versions of the same example (Chapter 6).

2.3.4.2 SpecAugment

SpecAugment is a data augmentation mechanism originally proposed for speech recognition, which consists of three transformations (Park et al., 2019). The first transformation is *time warping*, which consists of a deformation along the temporal dimension. The other two transformations are *time and frequency masking*, based on the popular augmentation named *Cutout* in computer vision (DeVries & Taylor, 2017). In particular, Park et al. (2019) propose to apply one or more rectangular-shaped masks over blocks of consecutive time frames or frequency bands. The width of the time masks and frequency masks as well as the amount of warping are drawn from uniform distributions bounded by pre-defined maximum values. For example, the width of every frequency mask is drawn from a uniform distribution, $\mathcal{U}(0, F_m)$, where F_m is a hyperparameter representing the maximum number of consecutive bands masked. The reader is refer to the original paper for further details. We use SpecAugment in the context of self-supervised audio representation learning in order to create different versions of the same example (Chapter 6).

2.3.5 Sound Classifiers

As mentioned earlier, at the beginning of this thesis we were witnessing a paradigm shift from feature engineering methods to techniques based on learning representations from data. Early feature engineering works in SER relied on approaches using machine learning classifiers such as support vector machines (Foggia et al., 2015), Gaussian mixture models (Mesaros et al., 2010) or matrix factorization techniques (Cotton & Ellis, 2011). This initial trend was followed by the rapid adoption of deep learning approaches using fully connected neural networks (Cakir et al., 2015), convolutional neural networks (Piczak, 2015a), recurrent neural networks (Parascandolo et al., 2016), or combinations thereof (Cakır et al., 2017). In the last year, some works have proposed the usage of *Transformer* architectures for SER, showing promising results with respect to the most widely-used CNNs. In particular, this type of architectures has achieved top results in sound event detection in the DCASE Challenge (Miyazaki et al., 2020), as well as competitive (Jaegle et al., 2021) or state-of-the-art (Gong et al., 2021a) results in AudioSet classification. In this thesis, following the most common trend in the sound recognition community, we mostly use CNNs. Section 2.5 provides a brief overview of their underlying principle and some relevant aspects.

2.3.6 Loss Functions

As explained in Section 2.3.1, the training of a deep network is based on updating the network weights in order to minimize a loss function that expresses the divergence between the network predictions and the target labels. For multi-class classification, the most commonly-used loss function is the Categorical Cross-Entropy (CCE) loss. The CCE loss is given by

$$\mathcal{L}_{cce} = -\sum_{c=1}^{C} y_c \log(p_c), \tag{2.2}$$

where y_c is the c'th element of the target label (a one-hot encoded vector), p_c is the c'th element of the network predictions (the predicted class probabilities), and C is the number of classes in the vocabulary. Since $\boldsymbol{y}$ is a one-hot encoded vector, only one term of the summation in Equation 2.2 is different from zero. That is, only the prediction for the true class contributes to the loss, and the rest of the predictions are ignored.

In multi-label classification, such as the problem posed by AudioSet or FSD50K, the output layer of the network is usually composed by C independent binary classifiers. In this setting, binary classification loss functions are typically adopted, composed by two terms, one accounting for the positive examples, and

the other for the negative ones. The default option is the Binary Cross-Entropy (BCE) loss, expressed by

$$\mathcal{L}_{bce} = -\sum_{c=1}^{C} y_c \log(p_c) + (1 - y_c) \log(1 - p_c), \tag{2.3}$$

where, again, p_c represents the network output prediction and y_c the target label for class c. Unlike in the multi-class case, here the terms for *all* the classes contribute to the loss, either through the positive term (i.e., the left term in Equation 2.3) or through the negative term (i.e., the right term of Equation 2.3). In Chapter 5 we will modify the negative term of Equation 2.3 in order to discard the loss contributions of missing labels.

2.3.7 Evaluation of Sound Event Classification Systems

In this thesis, sound event classifiers are trained for both multi-class and multi-label tasks. Each of these tasks is evaluated using different evaluation metrics, as explained next.

2.3.7.1 Evaluation Metric for Multi-Class Sound Event Classification

For multi-class classification tasks, systems are evaluated in terms of *classification accuracy*. Classification accuracy measures the number of correctly classified examples divided by the total amount of examples in a validation or evaluation set (Mesaros et al., 2018b). This simple metric has been widely used in sound event classification (Salamon & Bello, 2017) and also in other related multi-class tasks such as acoustic scene classification (Barchiesi et al., 2015). In order to avoid the influence of potential class imbalance, we report *balanced* accuracy (also sometimes denoted as *macro-average* accuracy): per-class accuracy values are computed first, then averaged with equal weight across all classes yielding the reported overall balanced classification accuracy.

2.3.7.2 Evaluation Metrics for Multi-Label Sound Event Classification

In the previous multi-class setting, only one class is to be predicted at a time, hence the class with highest prediction probability is selected. In multi-label settings, however, multiple class labels can be active at the same time. Therefore, when classifiers are deployed, a binarization technique must be applied to the output predictions in order to decide which classes are fired as active. Typically, this binarization is based on some kind of thresholding, which can

be optmized in different ways (e.g., Kong et al. (2020b)). However, the definition of optimal thresholding can be application-dependent, whereas we are interested in evaluating the performance of system in general terms, not tied to a specific application.

Likewise, some popular evaluation metrics for multi-label SET (e.g., F-score or overall error ratio) depend on an operating point, i.e., a decision threshold applied on the per-class output scores. These metrics encompass evaluation of the model's performance *and* of the decision threshold tuning. However, decoupling these two factors is desirable as, strictly, they are two different issues and, as mentioned, the optimality of the latter can be application-dependent. Thus, for the multi-label setting, we propose metrics able to evaluate a model's performance globally, integrating all possible operating points such that setting a decision threshold is not needed. This trend has been adopted in other fields such as speaker recognition (Van Leeuwen & Brümmer, 2007) and also recently in SED (Bilen et al., 2020).

On the one hand, we use common *within-class metrics*, i.e., metrics that rank all test samples according to the classifier score for one given class. These metrics deal with only one classifier output at a time, such that calibration across different classifier outputs is irrelevant. Following Gemmeke et al. (2017) and Hershey et al. (2017), we use Mean Average Precision (mAP) and d'.

mAP is the mean across classes of the Average Precision (AP), which summarises the precision-recall (PR) curve as the classifier decision threshold is varied. AP is calculated as the Precision (i.e., the proportion of positive samples in a ranked list) averaged across all the lists just long enough to recall a new positive sample (Scikit-learn documentation 0.23.2, 2020b; Hershey et al., 2017). AP is very similar to the area under precision-recall curve (PR-AUC), both being the most common ways of summarising a PR curve—the difference between them lies in implementation details (Scikit-learn documentation 0.23.2, 2020a; VLFeat.org, 2020).

d' (d-prime) is a metric for the performance of binary classifiers that can be computed as a monotonic transform of ROC-AUC (Green et al., 1966; Hershey et al., 2017):

$$d' = \sqrt{2}F^{-1}(ROCAUC), \tag{2.4}$$

where F^{-1} is the inverse of the cumulative distribution function for a unit Gaussian. d' measures the separation between the means of two unit-variance normal distributions (corresponding to the scores for positive and negative examples) that would achieve the same ROC-AUC. More details about d' can be found in Green et al. (1966).

To complement the within-class metrics, we propose to use a *between-class metric*, i.e., a metric that evaluates the overall ranking across all classifier outputs for every evaluation sample. Specifically, we use Label-Weighted Label-

Ranking Average Precision (*lwlrap*) (pronounced "lol wrap"), which was devised by Daniel P.W. Ellis of Google Research for DCASE Challenge 2019 Task 2. Intuitively, *lwlrap* measures, for every ground truth label c in the evaluation set, what fraction of the predicted top-ranked labels down to c are among the ground truth. More formally, let $Lab(s,r)$ be the class label at rank r (starting from 1) in test sample s, and $Rank(s,c)$ be the rank of class label c in that list, i.e. $Lab(s,Rank(s,c)) = c$. Then, if the set of ground-truth classes for sample s is $C(s)$, the label-ranking precision for the list of labels up to class c (assumed to be in $C(s)$) is:

$$Prec(s,c) = \frac{1}{Rank(s,c)} \sum_{r=1}^{Rank(s,c)} \mathbf{1}[Lab(s,r) \in C(s)] \tag{2.5}$$

where $\mathbf{1}[\cdot]$ evaluates to 1 if the argument is true, else zero. Therefore, $Prec(s,c)$ is equal to 1 if all the top-ranked labels down to c are part of $C(s)$, and at worst case equals $1/Rank(s,c)$ if none of the higher-ranked labels are correct. In contrast to plain *lrap*, which averages precisions within a sample then across samples, thereby downweighting labels that occur on samples with many labels, *lwlrap* calculates the precision for each *label* in the evaluation set, and gives them all equal contribution to the final metric:

$$lωlrap = \frac{1}{\sum_s |C(s)|} \sum_s \sum_{c \in C(s)} Prec(s,c) \tag{2.6}$$

where $|C(s)|$ is the number of true class labels for sample s. A Python implementation of *lwlrap* is provided in [10].

All metrics follow the behaviour of larger being better. mAP $\in [0,1]$, non-pathological $d' \in [0,\infty)$, and *lwlrap* $\in [0,1]$. As mentioned, all metrics integrate performance over all operating points, that is, none of the metrics requires setting a discrimination threshold. All metrics are computed on a per-class basis, and then averaged across all classes to yield on overall metric for the model. When averaging, equal weight is assigned to each class regardless of its prior, which is commonly referred to as *macro* averaging or *balanced* averaging, following Hershey et al. (2017); Gemmeke et al. (2017).

[10] https://colab.research.google.com/drive/1AgPdhSp7ttY18O3fEoHOQKlt_3HJDLi8

2.4 Datasets for Sound Event Recognition

This Section discusses the most important datasets for SET and SED. The datasets listed here are selected based on number of Google Scholar citations, as well as popularity and/or size for the most recent ones. The basic common aspect in SET datasets is that labels are provided at the clip-level (without timestamps), usually regarded as *weak* labels. This contrasts with SED datasets, where sound events are labeled using also start and end times (usually regarded as *strong* labels).

2.4.1 Datasets for Sound Event Tagging or Classification

Table 2.2 summarizes some aspects of a few most relevant SET datasets. For comparison, the proposed FSD50K is listed at the bottom (see Chapter 3 for details). *m-c* and *m-l* in the *task* column of Table 2.2 correspond to multi-class and multi-label, as previously defined in Section 2.3.

2.4.1.1 Datasets Released Before AudioSet

Before the release of AudioSet, the most widely used datasets for SET have been UrbanSound8K (Salamon et al., 2014), ESC-50 (Piczak, 2015b), and to a lesser extent CHiME-home (Foster et al., 2015). All of them feature short audio chunks and a total duration of less than 9h. Curiously, the two former are one of the few multi-class balanced datasets in SER—most datasets are unbalanced and/or multi-label—and also the most widely used (besides AudioSet). UrbanSound8K and CHiME-home count with a significant amount of clips per class; nonetheless, part of this abundance comes from the fact that many clips are actually time slices coming from the same original recording. For example, the 8732 instances in UrbanSound8K are in fact short slices sourced from 1302 Freesound clips (Salamon et al., 2014). ESC-50 features a large vocabulary (50 classes) when compared to other datasets from 2014/2015, but it suffers from data scarcity (only 40 clips/class). Common to all mentioned datasets is that they provide a k-fold cross validation setup—a practice that tended to disappear with the release of larger datasets such as AudioSet.

Table 2.2: A selection of most relevant datasets for SET. *m-c* and *m-l* correspond to multi-class and multi-label.

Dataset	Clips	Clip Length	Duration	Classes	Task	Source	Domain/Task
UrbanSound8K (Salamon et al., 2014)	8732	≤4s	8.8h	10 bal	m-c	Freesound	urban sounds
ESC-50 (Piczak, 2015b)	2000	5s	2.8h	50 bal	m-c	Freesound	
CHiME-home (Foster et al., 2015)	6138	4s	6.8h	7 unbal	m-l	CHiME[11]	domestic sounds
AudioSet (Gemmeke et al., 2017)	≈2.1M	≈10s	≈5731h	527 (un)bal	m-l	YouTube	
FSDnoisy18k (Fonseca et al., 2019b)	18,532	0.3-30s	43h	20 unbal	m-c	Freesound	noisy labels
FSDKaggle2019 (Fonseca et al., 2019c)	29,266	0.3-30s	103h	80 unbal	m-l	Freesound/ YFCC100M[12]	noisy labels & domain mismatch
SONYC-UST-V2 (Cartwright et al., 2020)	18,510	10s	51h	31 unbal	m-l	SONYC[13]	urban sounds
FSD50K 2020	51,197	0.3-30s	108h	200 unbal	m-l	Freesound	

2.4.1.2 AudioSet

Google's AudioSet is the largest dataset of sound events released to date, consisting of ≈2.1M audio clips manually labeled using 527 classes of the AudioSet Ontology (Gemmeke et al., 2017). AudioSet is the first dataset to put emphasis on general-purpose SER, enabling sound event recognizers to describe a huge variety of sound classes, thus aiming at the transcription of most everyday sounds. AudioSet is split into a train and an evaluation set, and it is highly imbalanced, with some classes being particularly common (e.g. *Music* and *Speech*) while others are much more scarce (e.g. *Toothbrush*). The public release provides a balanced train partition of 22,176 clips in addition to the full unbalanced train set. While the dataset is manually labeled in full (which entails a tremendous endeavour), its unprecedented size and coverage inevitably comes at the expense of a less precise labeling. In particular, the amount of labeling error in AudioSet is estimated at above 50% for ≈18% of the classes.[14] Recently, strong labels for a small portion of AudioSet (≈81k clips) were released (Hershey et al., 2021). The AudioSet Ontology is described in Section 2.2.4.1. We use a subset of this ontology to organize FSD50K.

In our view, AudioSet has the major shortcoming of not being an open dataset, as we explain next. AudioSet is composed of audio tracks taken from YouTube videos, which are not freely distributable due to YouTube Terms of Service. This is the reason why AudioSet is released as a dataset of audio features (instead of audio waveforms),[15] which are extracted at a time resolution of 960ms using a pre-trained model. This limits the adoption and flexibility of a number of SER methods. For this reason, some researchers opt to download and use the audio tracks from the original YouTube videos, despite the intrinsic issues entailed in this process. These issues include the burden of downloading a very large amount of data from a non-official release, and the fact that the constituent videos are gradually disappearing. More specifically, videos can become unavailable due to a variety of reasons such as deletions of videos or user accounts, privacy issues, copyright claims, or country-dependant availability. In an attempt to download the AudioSet audio tracks in May 2020, we could download 18,205 from 20,371 evaluation segments, and 19,862 from 22,160 balanced train segments—a loss of 10.6% and 10.4% respectively.[16] In May 2021, Google researchers reported that only 16,996 evaluation segments

[11] http://spandh.dcs.shef.ac.uk//projects/chime/

[12] https://multimediacommons.wordpress.com/yfcc100m-core-dataset/

[13] https://wp.nyu.edu/sonyc/

[14] See https://research.google.com/audioset/dataset/index.html for details on how the quality is estimated, accessed 25th June 2020.

[15] https://research.google.com/audioset/download.html

[16] Data from May 11th, 2020.

were available from the original release.[17] Two observations can be made from these numbers. First, 1209 evaluation segments have become unavailable during the period of approximately one year, which amounts to roughly 5.9% of the original number of segments. Second, by May 2021, 16.6% of the original evaluation set has become unavailable since its release. It must be noted that these numbers are likely to vary depending on the geographic location from where the download is carried out. The fact that the amount of evaluation and train clips available decreases over time with non-negligible differences limits AudioSet suitability for systems' benchmarking.

2.4.1.3 Datasets Released After AudioSet

After AudioSet, some of the released datasets for SET are task-dependent, designed to enable the study of particular SER problems. Examples include FSDnoisy18k (Fonseca et al., 2019b) or FSDKaggle2019 (Fonseca et al., 2019c), focused on learning in conditions of noisy labels and/or acoustic mismatch. Other datasets are domain-specific, with a vocabulary focused on a specific scope, such as SONYC-UST-V2 for urban sounds (Cartwright et al., 2020). Compared to pre-AudioSet datasets, these are slightly larger, especially in terms of duration as they feature longer clips (sometimes of variable length), but also in terms of vocabulary. In addition, they are unbalanced, and the default data split transitioned to a development/evaluation (or train/test) separation. Beyond those listed in Table 2.2, another large dataset is BirdVox-14SD for bird sounds (Cramer et al., 2020). Lastly, a recent large-vocabulary dataset with a substantial amount of data is VGGSound (Chen et al., 2020a), an audio-visual dataset consisting of ≈200k video clips from YouTube encompassing 300 classes. However, VGGSound presents several shortcomings for SER. The focus is put on audio-visual correspondence since the dataset is created mostly through automatic computer vision techniques—hence some classes have a clear visual connotation, e.g., *people eating noodle*. Also, while the dataset is singly-labeled (one machine-generated label per clip), the authors recognize that clips can contain a mixture of sounds. Upon inspection of the VGGSound vocabulary, it seems likely that sound events from different classes co-occur in the same clip (of 10s length), thus creating missing labels—for example, *cat growling* and *cat meowing*, or a combination of *sea waves*, *sailing*, *wind noise*, or *ocean burbling*. Missing labels are a form of label noise found to impact sound recognizers, as we will see in Chapter 5. While measures can be taken to mitigate their effect on training, in evaluation they can lead to misleading results—an issue that we specifically address in FSD50K (Section 3.2.7). In addition, VGGSound suffers from the intrinsic problems of being based on YouTube as seen in the previous Section.

[17] https://research.google.com/audioset/download_strong.html

2.4.2 Datasets for Sound Event Detection

Early stage datasets for SED were rather small as they were curated through manual annotation of sound events using start and end times (*strong* labels)—this process is especially laborious and sometimes ambiguous. Examples include TUT Sound events 2016 (Mesaros et al., 2016) and TUT Sound events 2017 (Mesaros et al., 2017b), each totalling ≈2h of annotated audio. To overcome this limitation, synthetic datasets became popular for SED, where soundscapes are generated by mixing a set of target sound events taken from other datasets with additional acoustic material. The main advantage of this approach is the larger control of many dataset aspects—in particular, sound event start/end times are reliable as they are determined by dataset construction. Further, provided the generation scripts are available, this paradigm allows for increasing dataset size arbitrarily. The main downside of this approach is that the synthesized soundscapes may not always be representative of real-world recordings, as pointed out by Salamon et al. (2017). This depends on factors such as the user-defined specifications for the generation, or the fact the generated soundscapes are based on combinations of a limited amount of sound event instances.

An early example of this approach is TUT Rare Sound Events 2017 (Mesaros et al., 2017b). URBAN-SED (Salamon et al., 2017) is a dataset synthesized by mixing sound events from the 10 classes of UrbanSound8K with Brownian noise using the Scaper library. An increasingly popular dataset is DESED (Turpault et al., 2019), covering 10 classes of domestic sounds. This dataset is composed of a set of recorded soundscapes from AudioSet (including unlabeled, weakly labeled, and strongly labeled portions), and a synthetic set constructed by mixing sound events from Freesound with additional material. Other instances of this approach include TAU Spatial Sound Events 2019 (Adavanne et al., 2019), for sound event detection and localization, and VOICe (Gharib et al., 2019) for the study of domain adaptation in SED. All SED datasets mentioned are unbalanced, pose a multi-label problem, and feature less than a dozen classes (except TUT Sound events 2016, with 18).

2.4.3 Gathering Reference Labels

Reference labels for sound events or music can be produced manually or with semi-automatic methods. Next, we discuss both approaches.

2.4.3.1 Gathering Labels Through Manual Annotation

Manual annotation of sound events for SET consists of listening to audio recordings in order to manually identify the sound events present in them, and

providing responses according to a given annotation task. This typically requires listening to every audio recording multiple times which makes this task a time-consuming process. Often, a predefined vocabulary is used for the annotation task, although annotation settings with a free vocabulary also exist in the literature (Van Grootel et al., 2009). Manual annotation is the conventional approach to dataset labelling, as done for example in AudioSet (Gemmeke et al., 2017) or ImageNet in computer vision (Deng et al., 2009). While this option is very laborious, when done properly we believe it leads to more reliable results than involving automatic methods in the annotation loop. To our knowledge, all datasets in Table 2.2 are labelled manually (except a portion of FSDnoisy18k and FSDKaggle2019 purposely included for the study of noisy labels).

There are different ways of gathering manual annotations for audio. Next, we cover some of them from the SER and MIR literature. If the amount of data to annotate is small, dataset creators can sometimes afford to annotate the data either by themselves or using a small pool of annotators. For example, in ESC-50 (Piczak, 2015b) and UrbanSounds8K (Salamon et al., 2014) the audio material is verified by the authors although no explicit details are provided about that process. To gather reference labels that can be considered ground truth, a common practice is to aggregate multi-annotator labels via majority voting. This is the case of CHiME-home (Foster et al., 2015), where three annotators are used per clip, and majority voting of their responses is used as reference.

For annotating larger amounts of data, the approach of *crowdsourcing* labels has gained attention in recent years. Crowdsourcing can be defined as a participative online activity where an organizer proposes the voluntary undertaking of a task (in our case, an annotation task) to an open group of individuals (Estellés-Arolas & González-Ladrón-de Guevara, 2012). The main advantage of this process is that it supports rapid collection of annotations as many individuals can contribute to the task. The main drawback of crowdsourcing annotations is a potential lack of control on the annotation task, often conducted by non-expert annotators under potentially heteregenous and unknown conditions, which can lead to inconsistent or incorrect annotations.

Different kinds of crowdsourcing approaches have been explored. A way to categorize them is based on how crowdworkers are rewarded by their contributions, including volunteering-based approaches, games with a purpose, and *paid-for* crowdsourcing (Sabou et al., 2014). In the sound and music computing field, a number of initiatives have explored the use of volunteering-based crowdsourcing approaches (Tsipas et al., 2013; McFee et al., 2016). The audio community has also explored the *gamification* approach, where the annotation task is presented as an engaging and entertaining experience, e.g., TagATune (Law et al., 2007) and MajorMiner (Mandel & Ellis, 2008) for collecting music

labels. Finally, a few *paid-for* crowdsourcing experiences exist, e.g., the SocialFX dataset (Zheng et al., 2016) using Amazon Mechanical Turk to collect annotations for sound effects. In the context of sound events, crowdsourcing has also received attention in the past few years. Cartwright et al. (2019b) compare the efficiencies of several audio annotation tasks in a setting with 22 sound classes of urban sounds. The annotation tasks include multiple-pass binary annotation—where the annotator must decide whether a single sound source is present or not in a recording—and single-pass multi-label annotation—where the annotator is tasked to select all the classes present in a recording. The authors conclude that multi-label annotation with inter-annotator agreement is the recommended choice for that setting and vocabulary size. As we will see in Chapter 3, in the creation of FSD50K we use the two aforementioned types of annotation tasks: a binary annotation task to validate previously nominated candidate labels (Section 3.2.5), as well as a multi-label annotation task relying on experts in order to identify all present sources in a recording (Section 3.2.7). Another recent work studies how to estimate strong labels using crowdsourced weak labels (Martín-Morató et al., 2021). To this end, annotators are presented with highly overlapped audio segments, and asked to provide weak labels following a single-pass multi-label annotation task with 6 sound classes. Then, events' temporal activity is reconstructed based on the (non-)present labels of the individual overlapped segments.

Finally, other works make use of *expert annotators* (Tsipas et al., 2013). The main advantage of this approach is higher reliability of the annotators performing the task, possibly leading to a higher annotation quality. This reliability could be required for certain complex annotation tasks in order to minimize label noise. However, this approach typically means relying on a reduced number of individuals (as opposed to crowdsourcing), which can slow down the annotation process. Further, leveraging experts can imply an economic cost.

As we will see in Chapter 3, for the creation of FSD50K we use the two main annotation strategies identified: crowdsourcing with non-expert annotators for the classes that are considered not difficult, as well as expert annotators that received training to perform the task.

2.4.3.2 Gathering Labels Through Semi-Automatic Methods

Instead of relying solely on manual annotation, other approaches for gathering labels consider the utilization of semi-automatic methods. One example of this methodology is the so-called *active learning*. Active learning aims at maximizing performance with limited labelling budget by selecting the most informative data for the model to learn. Usually active learning is based on an iterative process involving humans in the loop where automatic methods are used to select the samples to annotate (Riccardi & Hakkani-Tur, 2005). Annotated samples

are commonly used to train models that in turn help to select a new batch of samples to annotate. Often, portions of the non-selected unlabeled samples are automatically labelled via propagation of human-provided labels to similar examples, or with semi-supervised learning approaches. Recent works studying active learning methods for SER (Han et al., 2016; Shuyang et al., 2018; Wang et al., 2019b; Shuyang et al., 2020) report reduced annotation effort with good model performance which, in principle, makes active learning appealing for dataset creation. However, these works focus on recognition tasks with less than a dozen classes, and most of them deal with single-label classification and use pre-labeled datasets, where the human annotation step is simulated by a simple assignment of the existing ground truth. In addition, it seems the success of these methods is somewhat problem-specific, depending on factors such as the complexity of the classification task or the annotated data available to train automatic methods, as noted in Han et al. (2016); Shuyang et al. (2020). This casts doubts on the applicability of active learning to our more complex scenario, requiring multi-label annotation of samples with a vocabulary of hundreds of classes (some of them rather ambiguous). Relatedly, previous work in image recognition evaluates an active learning method on two datasets of 10 classes and on CIFAR-100 (of 100 classes) (Sener & Savarese, 2018). The proposed method is found less effective in CIFAR-100 due to the larger number of classes. To our knowledge, there is not any released large-vocabulary sound event dataset that has used active learning in its creation under similar circumstances to ours, and active learning in large-vocabulary settings has not been studied in SER. Thus, this is considered a research problem out of the scope of this thesis (albeit an interesting topic for future research).

Another approach to annotate audio is to rely solely on pre-trained models to produce machine-generated labels. In this way, audio content can be automatically annotated without usage of manual annotation. The main shortcoming of this approach is that the reliability of the annotations depends on the model's proficiency. An example of a dataset built following this trend is VGGSound (Chen et al., 2020a). As mentioned in Section 2.4.1.3, this dataset presents several shortcomings for SER, such as the presence of missing "Present" labels both for training and evaluation.

2.4.4 Freesound

Despite the fact that Freesound is not a dataset strictly speaking, we include its description here as it is the source of data for the datasets created in this thesis. Freesound[18] is an online collaborative audio clip sharing site (Font et al., 2013), with more than 10 million registered users, over 500,000 audio

[18] https://freesound.org/

clips, and an average of 3400 new clips added every month.[19] Audio clips shared in Freesound cover a wide variety of audio content, from music samples to environmental sounds, human sounds or audio effects, to name a few. In addition, the users who upload the clips also provide metadata, e.g., a title, several tags (at least three per clip), and textual descriptions. We use the user-provided tags in the creation of FSD50K (Section 3.2.4). Since Freesound is collaboratively contributed, it is also very heterogeneous in terms of data origin, recording equipment, and acoustic conditions. One of the most popular use cases of Freesound is the exchange of well-recorded audio samples for creative purposes. All of the content is Creative Commons licensed,[20] which conveniently allows distribution and reuse. As we have seen above, several datasets containing Freesound audio have been widely used by the research community, showing its usefulness for dataset creation (Stowell & Plumbley, 2014; Salamon et al., 2014; Piczak, 2015b).

2.5 Convolutional Neural Networks for Sound Event Classification

CNNs are neural networks that use convolution operations instead of general matrix multiplication in at least one layer (Goodfellow et al., 2016). Convolutional layers implement a cross-correlation function by sliding filters (also sometimes called *kernels*) over an input representation, outputting a response often referred to as *feature map* (Goodfellow et al., 2016). CNNs overcome some of the limitations of their preceding Multi-Layer Perceptrons (MLPs). For example, CNNs have *sparse interactions* by using kernels that are smaller than the input, instead of using matrix multiplications that describe interactions between all output and input units. CNNs are *shared-weight* architectures, where each weight of a kernel is used at every position of the input representation, which is more efficient than dense matrix multiplication. Due to the sharing of weights, convolutional layers are *equivariant to translation*—if the location of a specific pattern is shifted in the input representation, its location in the feature map will be shifted accordingly.

CNNs are constructed by stacking convolutional layers in such a way that hierarchical representations can be learned. Typically, high-level representations are learned at the end of the network, based on lower-level patterns extracted at the preceding layers. Convolutional layers are usually interleaved with pooling layers, and also with normalization layers (e.g., Batch Normalization (Ioffe & Szegedy, 2015)) and non-linearities (e.g., Rectified Linear Units (Nair & Hinton, 2010)). Pooling layers are used to downsample feature maps, redu-

[19] Data from August 1st, 2021.
[20] https://creativecommons.org/

cing the dimensionality processed by the network and allowing the integration of larger areas of information by the deeper layers. In addition, they provide the network with some invariance to shifts. For SET, after the convolutional stack, the resulting feature map is typically aggregated through some kind of global *summarization* pooling. Finally, this summarized representation feeds a fully-connected classifier layer with an appropriate activation in order to produce per-class probability predictions. For a more detailed description of CNNs the reader is referred to Goodfellow et al. (2016) and McFee (2018), on which this Section is based.

Regardless of the specific Convolutional Neural Network (CNN) architecture, the activations of the final classifier layer must be chosen according to the classification task. For multi-class tasks, the *softmax* activation is commonly adopted, which squashes all per-class output predictions so that their sum equals unity. For multi-label classification tasks, the classification layer typically uses a *sigmoid* activation per output channel, thus forming a C-way classifier where each channel outputs class probabilities independently. Likewise, these activations are typically associated with the loss functions introduced in Section 2.3.6—softmax activation with CCE loss for multi-class problems and sigmoid activation with BCE loss for multi-label problems. Nonetheless, there exist exceptions to these widely-assumed conventions, such as Mahajan et al. (2018), where softmax activation is used with multi-label data.

In practice, the properties and structure of CNNs provide them with advantages. For instance, CNNs can accommodate inputs of variable size provided appropriate pooling strategies are in place. In addition, compared to MLPs, they are also less sensitive to the relative position of sound events within the context of the input representation (McFee, 2018). However, we will see that this shift invariance property is only partial, and increasing it is beneficial for SET.

CNNs have been one of the cornerstones of SET in recent years (Salamon & Bello, 2017; Fonseca et al., 2019a; Kong et al., 2020a; Fonseca et al., 2020b; Gong et al., 2021b). They have also proved to be effective for several audio related tasks, e.g., speech recognition (Lee et al., 2009), automatic music tagging (Dieleman & Schrauwen, 2014) or acoustic scene classification (Valenti et al., 2017). In the context of environmental sound recognition, works can be categorized into two different trends. Some works adopt off-the-shelf CNN architectures taken from the computer vision field (sometimes with minor modifications) (Kong et al., 2020a; Fonseca et al., 2020b; Gong et al., 2021b). Other works attempt to design *ad hoc* CNN architectures motivated by addressing specific aspects of the audio data, e.g., Fonseca et al. (2018a); Phaye et al. (2019); Cramer et al. (2020).

2.5.1 CNNs vs CRNNs for Sound Event Classification

Convolutional Recurrent Neural Networks (CRNNs) are deep architectures consisting of the combination of convolutional layers and recurrent layers (Cakır et al., 2017). In particular, the network begins with a number of convolutional layers whose purpose is to extract local features from the input representation. After stacking the feature maps resulting from the last convolutional layer, this representation is fed to one or more recurrent layers. The purpose of the recurrent layers is to aggregate the extracted features over time, thus modeling discriminative temporal structures.

Cakır et al. (2017) showed that CRNNs provide boosts over CNNs and Recurrent Neural Networks (RNNs) alone, using three datasets for SED and one for SET. CRNNs became a popular choice in SED works (e.g., Lim et al. (2017); Adavanne et al. (2017)) and also among various DCASE Challenge Tasks. In contrast, other works highlight the similar performance of CRNNs and CNNs. For example, Kong et al. (2020b) report that CRNNs perform on par with CNNs on the SED problem of DCASE Challenge 2017 Task4 (Mesaros et al., 2017b). Salamon et al. (2017) also found that both architectures achieve similar overall performance on URBAN-SED, while each architecture specializes better on certain sound classes. The state-of-the-art on the SET task of AudioSet in the last couple of years has been mostly CNN-based (Kong et al., 2020a; Gong et al., 2021b).

In this thesis, we conducted preliminary experiments comparing typical architectures of CRNNs and CNNs for SET, also finding that they show similar results overall. However, we have observed that CRNNs require longer runtimes due to the extra computation required by recurrent layers, which slows down experimentation. Therefore, we decided to adopt CNNs as the default type of architecture for the majority of our experiments, although we also have published experimental results with CRNNs in Pérez-López et al. (2019) and Fonseca et al. (2020a).

2.5.2 Shift Invariance in CNNs

One of the commonly-assumed properties of CNNs is *shift* or *translation invariance*, by which output predictions are not affected by small shifts (or even small deformations) in the input signal. In theory, this is ensured by the convolution and pooling operations forming the CNNs. However, recent works in computer vision uncover that this is not always the case. Azulay & Weiss (2018) find that small shifts and transformations in the input can change the network's predictions substantially. In particular, they quantify that by shifting or resizing a random input image by one single pixel, the top class predicted can change with a probability of up to 15% and 30%, respectively. This and

other related works (Engstrom et al., 2018; Zhang, 2019) empirically show the brittleness of CNNs against minor input perturbations, and their only-partial invariance to shifts.

These works argue that one of the causes of the lack of shift invariance is a wrongly executed subsampling operation that ignores the classic sampling theorem. This theorem establishes that, for the subsampling to be done correctly, the sampling rate must be at least twice the highest frequency in the incoming signal (Oppenheim et al., 2001). Otherwise, *aliasing* problems can occur, generating lack of shift invariance in the system and potentially causing a certain distortion in the output—some of the highest frequency components can overlay other low frequency ones. To address this issue, the classical signal processing measure is to introduce an *anti-aliasing* low-pass filter before downsampling in order to limit the signal's band (Oppenheim et al., 2001). In CNNs, subsampling operations are prevalent through strided layers, e.g., convolution or pooling layers with a stride larger than one. As anti-aliasing actions are not usually taken, feature maps containing high frequency components may lead to shift invariance and/or distortion problems.

The findings above have led to a growing area of research aimed at increasing shift invariance in CNNs, either through architectural improvements (Zhang, 2019; Chaman & Dokmanic, 2021; Vasconcelos et al., 2020) or via data augmentation (Engstrom et al., 2018). In this work, we are interested in the former, which usually revolves around the idea of improving the subsampling operations. The predominant trend consists of adding anti-aliasing measures to the CNN architectures. Similarly to the signal processing fix, some works adopt different low-pass filter based solutions, mainly for image recognition (Zhang, 2019; Vasconcelos et al., 2020) and more recently also for speech recognition (Bruguier et al., 2020). Zhang (2019) demonstrates that adding blurring to deep convolutional networks before the strided operations (convolution and pooling) provides increased accuracy on ImageNet (Deng et al., 2009) and improved robustness to image perturbations. Vasconcelos et al. (2020) conduct a study to isolate the impact of aliasing within the different modules of a ResNet-50 architecture. Bruguier et al. (2020) insert 1D low-pass filters along the temporal dimension of feature maps in a Recurrent Neural Network (RNN)-based speech recognizer.

In contrast to the anti-aliasing line of work, another alternative is to design architectural changes to explicitly enforce invariance in the network. For example, several previous works focus on increasing the invariance of CNNs to *rotations* in input images, by applying constraints to the convolutional filters (Worrall et al., 2017) or proposing *ad hoc* operations to enforce this property (Dieleman et al., 2016). Recently, to address the lack of shift invariance caused by subsampling operations, Chaman & Dokmanic (2021) propose a downsampling mechanism called *adaptive polyphase sampling*. The key idea

is to avoid using the same fixed sampling grid for subsampling a feature map (as typically done in CNNs), but instead select it adaptively based on some criterion (e.g., choosing the grid that produces a downsampled output with highest energy). To our knowledge, this kind of techniques aimed at fostering shift invariance in CNNs have not been evaluated for sound event classification. This is the subject of Chapter 4.

2.6 Learning with Noisy Labels

As introduced in Section 1.3.1, label noise is a pressing issue for the future of large-scale machine perception in the era of data-driven methods. The effects of label noise can include performance decrease, increased complexity of learned models, or changes in learning requirements (Frénay & Verleysen, 2014). Further, label noise has been reported to specifically hinder the proper learning of deep neural networks (Arpit et al., 2017; Zhang et al., 2017). Next, we provide a brief literature review of the topic of *learning with noisy labels*. First, we focus on the computer vision field, then we move to the SET field, and we conclude by motivating the work done on this topic in the context of this thesis.

2.6.1 Learning with Noisy Labels in Computer Vision

Learning with noisy labels has been an intense area of research in computer vision during the last decade. Recent works have proposed different ways to categorize the many approaches proposed in the literature (Song et al., 2020; Algan & Ulusoy, 2021). Nonetheless, due to the diversity in the existing approaches, the authors acknowledge that there are no sharp boundaries among the proposed group of methods. Some of the most relevant research directions are identified next.

Regularization methods. Regularization aims to prevent overfitting and improve generalization, which can also be beneficial against label noise. If we understand the performance decrease caused by the noisy supervision as a problem of overfitting noisy labels, regularization methods can be used to prevent deep networks from this type of overfitting. Typical examples include dropout (Srivastava et al., 2014), mixup (Zhang et al., 2018), or label smoothing (Szegedy et al., 2016; Pereyra et al., 2017). A recent work has shown that pre-training can also have a regularization effect, improving model robustness to label corruption (Hendrycks et al., 2019a).

Noise-robust loss functions. These loss functions are designed to allow training and convergence of models while ignoring the effect of noisy labels. The goal of these algorithms is to minimize the performance degradation derived from the noisy labels during the process of loss optimization. Some examples are the bootstrapping loss (Reed et al., 2015), the generalized cross-entropy loss (Zhang & Sabuncu, 2018), a loss based on the mean absolute error expression (Ghosh et al., 2017) or its recent improved version (Wang et al., 2019a). One of the great advantages of this approach, along with the regularization methods, is that they require minimal intervention in learning pipelines.

Rejection of noisy samples/labels. The previous approaches are based on *accepting* the noisy labels and mitigating their effect during training. A diferent perspective consists of actively rejecting the contribution of noisy samples (or noisy labels) during training. This rejection-based approach has been widely-adopted (Malach & Shalev-Shwartz, 2017; Jiang et al., 2018; Han et al., 2018; Nguyen et al., 2019), sometimes achieving state-of-the-art performance. The assumption here is that removing the noisy items leads to performance improvement. Methods in this category rely on some noise detection mechanism to identify noisy samples (or labels) that are subsequently discarded. A popular mechanism consists of using multiple networks or model checkpoints to conduct the detection, based on agreement of labels with network predictions (Nguyen et al., 2019), or based on cross-network disagreements using the loss values associated with training instances (Han et al., 2018). Other works opt for a teacher-student framework where the teacher estimates per-example weights that are subsequently used by the student (Jiang et al., 2018). The inherent risk of this research direction is the possibility of removing legit samples in addition to the corrupted ones due to suboptimality of the detection mechanisms, which may lead to performance degradation. These rejection-based methods are also referred to as *instance selection* or *dataset pruning* methods (Algan & Ulusoy, 2021).

Noisy label Correction. This category represents a step further with respect to the previous group, where methods correct the noisy labels after identifying them. These methods typically rely either on a pre-trained network or simply the model being trained in order to predict on train data and use the predictions for label correction. In Yuan et al. (2018) a committee of networks is trained on different subsets of data, and label agreements that differ from actual labels are used as new target labels. Tanaka et al. (2018) propose a joint optimization framework where network parameters are learned and true labels are estimated during training by alternating updates of parameters and labels. Other approaches leverage an additional set of curated clean data, for

example to train a label cleaning network in order to reduce the noise of a dataset (Veit et al., 2017). The noisy label correction approach is appealing as, theoretically, no loss (or minimal loss) of data is suffered. However, this ideal scenario is restricted to the case where true labels for data instances are within the vocabulary of the *oracle* model, so that the model can indeed predict them.

2.6.2 Learning with Noisy Labels in Sound Event Classification

Unlike in computer vision, learning with noisy labels has received little attention in the field of sound event classification, presumably due to the traditional paradigm of learning from relatively small and exhaustively labeled (hence *clean*) datasets. When the work for this thesis started, there was little prior work on how to improve sound event classification in presence of noisy labels. To our knowledge, one of the first works on this topic is Kumar & Raj (2017), where an approach is proposed to train classifiers with weakly labeled web data, and a small amount of strongly labeled data is used to boost performance. However, the data used in this work is not publicly available. Other works focus on self-training to learn from combinations of labeled and unlabeled data (Zhang & Schuller, 2012; Han et al., 2016; Elizalde et al., 2017), but the issue of label noise is not addressed *per se*. In Shah et al. (2018), the effect of label noise on weakly supervised learning is analyzed by introducing noise to AudioSet. However, no measures to mitigate the effect of label noise are proposed.

During the course of this thesis, the topic of learning with noisy labels in sound classification received some more attention. Some of the published works are based on the work carried out in this thesis, including the publications summarized in Chapter 5, as well as the co-organization of two DCASE Challenge Tasks in 2018[21] and 2019[22], where the topic of label noise was included for the first time as a research problem (see Appendix A). We next briefly summarize some of the published works based on our publications and resources. Some works attempt to distinguish between noisy and correct labels with the goal of selecting the latter, for instance, with self-training methods where clean labels are iteratively selected and added to the train set for re-training (Dorfer & Widmer, 2018; Nguyen et al., 2018). Singh et al. (2019) propose to model the label noise distribution in data by inserting a linear layer into a network. Iqbal et al. (2020) presents an approach where first noisy instances are detected using an auxiliary classifier trained on clean data, and then some of them are

[21] http://dcase.community/challenge2018/task-general-purpose-audio-tagging
[22] http://dcase.community/challenge2019/task-audio-tagging

relabelled. Zhu et al. (2020) leverage the interaction of two networks to incrementally select the possibly correctly labeled data from a noisy set of data. In particular, this work utilizes the two datasets created for the aforementioned DCASE Challenge Tasks, *FSDKaggle2018* and *FSDKaggle2019*, which are introduced in Appendix A. Another recent trend is to adopt *Graph Convolutional Networks* from the image recognition literature (Chen et al., 2019) for the task of multi-label SET under noisy labels. For example, Shrivastava et al. (2020) propose a multitask learning module to learn from clean and noisy data from the FSDKaggle2019 dataset, and a Graph Convolution module that utilizes the label co-occurring relationships to regularize the network.

Other works published during the course of this thesis include Kumar et al. (2019), where two networks operating on different views of the data co-teach each other to learn from noisy labels. In Kumar & Ithapu (2020), a cascade of learners is proposed where the training label set at every stage is a combination of the original label set and the combined supervision from the previous classifiers.

2.6.3 AudioSet from the Perspective of Label Noise

AudioSet is composed of over 2M of manually annotated audio clips (Gemmeke et al., 2017). Due to its unprecedented size, the labels are not as precise as in other smaller and exhaustively labeled datasets, as discussed in Section 1.3.1. Thus, AudioSet presents a number of label noise problems. Some of them are due to shortcomings in the annotation process, e.g., missing or incorrect labels. Others are related to the hierarchical structure of the AudioSet Ontology, e.g., a segment may be annotated with a leaf class label but not with its parent one, or annotated with a label that is not the most specific within its hierarchical path. Still other problems arise from the temporally-weak labels (i.e., clip-level labels), where the class label may be active only during a small (and unknown) portion of the audio segment. Finally, some semantic inconsistencies may exist as the ontology allows for several sound attributes to be associated to one type of sound event (while not all of them may have been annotated).

Despite these label noise problems, they have been directly addressed in only a few of the previous works using AudioSet (e.g., Kumar et al. (2019); Kumar & Ithapu (2020)), while the majority of efforts focus on deriving more sophisticated network architectures that ignore or downplay the idiosyncrasies of the labeled audio data (e.g., Ford et al. (2019); Kong et al. (2020a)).

In Section 5.6, we propose a system to address one of the most frequent label noise problems in AudioSet: its missing labels. The study of missing labels in SER has received very little attention. To our knowledge, this specific topic has been covered only by Meire et al. (2019), where robustness to missing labels is studied by simulating them in a synthetic dataset of 20 classes.

2.6.4 Learning with Noisy Labels in This Thesis

At the beginning of this thesis, the lack of publicly available data was hampering label noise research on common public evaluation benchmarks. One of our contributions is *FSDnoisy18k*, an openly-available audio dataset that supports the investigation of real label noise, including an empirical characterization of the noise and a CNN baseline system (Section 5.2).

Furthermore, as seen before, many of the computer vision approaches against label noise are relatively complex. Quite a few of them employ dedicated pipelines, i.e., learning pipelines designed *ad hoc* for the purpose of label noise mitigation, often leveraging two or more networks for sample rejection or label correction, as well as auxiliary classifiers. Others rely on extra data resources, such as an additional set of curated clean data. Given the early stage of this field in SET at the beginning of this thesis, we opted to explore simple and efficient approaches, agnostic to network architectures or learning settings. Specifically, we seek approaches that can be easily incorporated into existing learning pipelines composed by a noisy dataset and a deep network, requiring only minimal intervention of the learning pipeline, and no extra resources. By choosing methods that can be easily inserted into existing pipelines, our goal was to facilitate their adoption. In this way, SET practitioners do not have to renounce their own learning pipelines to adopt label noise mitigation mechanisms. In particular, we focus on regularization techniques, noise-robust loss functions, and approaches based on instance selection that reject potentially noisy samples (Sections 5.3 and 5.4). These approaches address generic inaccuracies of the labels. Finally, we also propose a label-rejection based system to address one of the most frequent label noise issues in everyday sound labeling, and in AudioSet specifically: missing labels (Section 5.6).

2.7 Self-supervised Learning

2.7.1 Terminology

Before diving into the details of self-supervised learning, we first clarify the terminology used in this thesis. The terms *unsupervised learning* and *self-supervised learning* can be found in the literature, sometimes interchangeably. However, their usage has evolved over the years. Traditionally, the term *unsupervised learning* is broadly used to denote *any* machine learning method that operates over data without human labels, that is, using *unlabeled data*. Examples include Lee et al. (2009) and Jansen et al. (2018), in which unsupervised audio representations are learned.

However, the term *self-supervised learning* has become popular recently. Self-supervised learning (or self-supervision) denotes machine learning methods that are also applied over unlabeled data but, in this case, the term specifically conveys the idea that some supervision does exist for the learning task. This supervision is derived from the data itself—in our case, the audio signal—using specific patterns in the data as we will see with some examples below. One could argue that denoting these methods as *unsupervised* can be misleading as, strictly, there exists indeed a supervisory signal for the learning task—not an external supervision from labels, but some sort of *self*-supervision. Hence, sometimes, the term *unsupervised learning* seems to have been set aside to encompass methods related to clustering or dimensionality reduction, where self-supervision is not used.

With that being said, we will use the structure *unsupervised (audio) representation* to denote a representation or *embedding* that is learned from unlabeled data, following Jansen et al. (2020). In a similar fashion, we will use the structure *unsupervised contrastive learning* or *unsupervised contrastive training* to refer to a process of training with unlabeled data, following Chen et al. (2020b) (even though, strictly, the training is *self*-supervised).

2.7.2 Problem Formulation

Self-supervised learning methods aim at learning representations without the need for external supervision. Absent explicit labels generated by humans, the success of these methods relies on the design of *proxy* learning tasks[23] in which pseudo-labels are generated from patterns in the data (i.e., the *self*-supervision). By training networks to solve these proxy tasks on unlabeled data, mappings from inputs to useful low-dimensional representations are learned. The learned representations can then be used for downstream tasks such as classification. Once the self-supervised training has converged, some common real-world use cases include: *i)* The trained embedding network can be used as a feature extractor in order to compute semantically-relevant audio features on few data with available labels. Then, these labeled features can be used to train shallow classifiers. *ii)* Alternatively, it is also possible to use the embedding network's weights as initialization for fine-tuning on (typically scarce) labeled data. A concrete example of the usefulness of self-supervised audio representation learning is that of low-resource languages (Kawakami et al., 2020). First, speech representations are learned using available sources of abundant unlabeled speech audio. Then, speech recognition models are trained using the learned representation on much smaller datasets of low-resource languages. This learning paradigm has led to recognition boosts for languages where labeled data is scarce, e.g., Swahili (Kawakami et al., 2020).

[23] Proxy tasks are also sometimes referred to as *auxiliary* or *pretext* tasks.

More formally, the problem formulation in self-supervised audio representation learning is to train a deep audio embedding network f_θ with parameters θ that maps input examples to a low-dimensional representation. In the context of this thesis, this network defines a mapping $f : \mathbb{R}^{T \times F} \rightarrow \mathbb{R}^d$ from log-mel spectrograms of T time frames and F frequency bands to a d-dimensional representation. We aim at learning audio representations that support sound event classification tasks, for example where only few data, or poorly labeled data, are available.

2.7.3 Related Work

The self-supervised learning paradigm has seen major progress in natural language processing (Devlin et al., 2019), computer vision (Chen et al., 2020b,c; Grill et al., 2020) and in speech recognition (Oord et al., 2018; Baevski et al., 2019, 2020). For example, one of the most iconic self-supervised methods is the BERT model in natural language processing (Devlin et al., 2019). There, the proxy task essentially consists of predicting a series of masked words within a sequence of words. In this setting, the masked words act as a supervisory signal, i.e., as reference labels. With this reconstruction task, powerful text representations can be learned. A similar proxy task applied in the context of learning audio representations is *Audio2Vec*, where the goal is to reconstruct a spectrogram patch from past and future patches (Tagliasacchi et al., 2020). In this work, the authors also propose the proxy task of *TemporalGap*, which consists of estimating the time distance between pairs of audio segments in an audio stream. A variety of proxy tasks has been devised for self-supervised representation learning in various modalities.

Recently, the approach of *contrastive learning* has received particular interest within the self-supervision literature (Le-Khac et al., 2020). The mechanism of contrastive representation learning consists of *learning representations by comparing pairs of examples* selected by some semantically-correlated notion of similarity (Le-Khac et al., 2020). Specifically, comparisons are made between positive pairs of "similar" and negative pairs of "dissimilar" examples, with the goal of learning a representation that pulls together positive pairs and thus reflects semantic structure. While contrastive learning has gained special interest only recently, its origins date back to the 1990s. One of the first works adopting the paradigm of learning by comparing between examples without any external supervisory signal is Becker & Hinton (1992). In this work, representations are learned by maximizing the agreement between separate but related parts of input images.

Prior to this thesis, the topic of self-supervised learning of general-purpose audio representations had not been very explored. The scarcity of studies on learning environmental sound representations may be partly due to the lack

of abundant readily-available sources of audio data. For example, Lee et al. (2009) used convolutional deep belief networks to learn representations for speech and music, but not for general-purpose audio. During the course of this thesis, a few works gradually arose, the majority of them based on contrastive learning using AudioSet. One of the first works in contrastive audio representation learning is Jansen et al. (2018). This work relies on a triplet loss approach, where anchor-positive pairs are created by sampling neighboring audio frames as well as applying other simple audio transformations (e.g., adding noise or mixing examples). In a subsequent work from the same authors, the proxy task of *audio-visual coincidence prediction* is used to learn audio representations to support predicting whether a pair of examples occurs within a certain temporal proximity (Jansen et al., 2020). The coincidence prediction task is a generalization of the *correspondence prediction* task proposed for audio-visual multimodal learning, where the task is to predict time *correspondence* between audio and video frames (Arandjelovic & Zisserman, 2017). Another approach uses the so-called *contrastive predictive coding* together with additive adversarial perturbations applied to the waveform to obtain a harder proxy task (Wang et al., 2020).

A promising trend within contrastive learning that gained attention recently is to learn representations via the proxy task of *similarity maximization*, attaining promising results in visual representation learning with approaches such as the *SimCLR* framework (Chen et al., 2020b). This method consists of maximizing the similarity between differently-augmented versions or *views* of the same data example. Critical to its success is the simultaneous use of a diversity of semantics-preserving, domain-specific augmentation methods to create the different example views. After achieving state-of-the-art in several image recognition tasks (Chen et al., 2020b,c), this approach has been successfully applied for speech recognition tasks (Nandan & Vepa, 2020; Kharitonov et al., 2021). For learning sound event representations, the practice of using augmentations to create different views of audio examples has been adopted by Jansen et al. (2018). However, in that work augmentations are applied individually and not in composition, as recent visual representation learning works suggest (Chen et al., 2020b,c).

In this thesis, we focus on learning sound event representations by contrasting differently-augmented views of sound events. To this end, a number of semantics-preserving augmentations are explored in order to create different example views, some of which have never been applied in this context, to our knowledge. After composing some of these multiple augmentations into a single augmentation front-end, our audio representations are learned by solving the aforementioned proxy tasks of similarity maximization and/or coincidence prediction.

CHAPTER 3

The Freesound Dataset 50k (FSD50K)

3.1 Introduction

As discussed in Section 1.2, most existing datasets for SER are relatively small and/or domain-specific, with the exception of AudioSet. Unfortunately, AudioSet suffers from openness and stability issues. Thus, the SER field lags far behind in terms of dataset availability when compared to fields like computer vision where, in addition to ImageNet, major efforts have been made to collect alternative large datasets, e.g., Lin et al. (2014); Li et al. (2017b); Kuznetsova et al. (2020). Dataset creation initiatives are needed to allow exploitation of deep learning approaches for SER and foster machine listening research.

In addition, we think it is important to document at length the main aspects of data collection and curation when releasing a dataset—a common practice in computer vision (Russakovsky et al., 2015; Kuznetsova et al., 2020) that has also recently been proposed in audio research (McFee et al., 2018a). Making this information available allows researchers to incorporate data-informed decisions in the design of learning pipelines and in the analysis of results, and can also serve as inspiration for potential dataset creators.

To address these issues and foster SER resesarch, in this Chapter we introduce *FSD50K* (**F**ree**s**ound **D**ataset **50k**): a dataset containing 51,197 audio clips totalling over 100h of audio manually labeled using 200 classes drawn from the AudioSet Ontology. FSD50K is a dataset primarily designed for the development and evaluation of multi-label sound event classification systems, but that also allows a variety of sound event research tasks. The audio clips are gathered from Freesound and are licensed under Creative Commons (CC) licenses, which allow easy and stable sharing, thereby making the dataset freely distributable (including audio waveforms). To our knowledge, this is the largest

fully-open dataset of human-labeled sound events, and the second largest after AudioSet. In Section 3.2, we provide a detailed description of the FSD50K creation process tailored to the particularities of Freesound data, including challenges encountered and solutions adopted. Section 3.3 provides a comprehensive characterization of the dataset along with discussion of limitations and key factors to allow its audio-informed usage. In Section 3.4, we include a set of sound event classification experiments to provide baseline systems as well as insight on the main factors to consider when splitting Freesound audio data for SER tasks. Section 3.5 concludes the Chapter with a summary of the key results and directions for future work.

This Chapter is useful to researchers using FSD50K (and in general using Freesound data for machine learning) as it allows making data-informed decisions for design choices of machine listening systems. It may also be useful for researchers working on the creation of large-vocabulary datasets. In addition to the audio waveforms and ground truth, FSD50K includes metadata used during the creation process as well as Freesound metadata for the clips forming the dataset (Section. 3.3.1). All these resources can be downloaded from Zenodo.[24] Likewise, code[25] for baseline experiments and a companion site[26] for FSD50K is also available. The companion site allows exploring the audio content of FSD50K as well as reporting labelling errors.

3.2 The Creation of FSD50K

3.2.1 Design Criteria

As design criteria, we set three basic goals and another three specific goals. The basic goals are: *i)* the dataset must be open and fully distributable, *ii)* it must contain a large vocabulary of everyday sounds, and *iii)* it must be expandable in terms of data and vocabulary. To fulfil these basic goals, we turn to Freesound as a source of data, and to the AudioSet Ontology as a vocabulary to organize the data. Not only do these resources feature a large amount of data and classes, respectively, but Freesound is constantly growing through user uploads, and the ontology is large and was designed to be expandable, allowing dataset expansions. They are described in Sections 2.4.4 and 2.2.4.1, respectively.

In addition, we set three specific goals related to the labeling of the dataset and to the emphasis put on the evaluation set.

[24] https://doi.org/10.5281/zenodo.4060432
[25] https://github.com/edufonseca/FSD50K_baseline
[26] https://annotator.freesound.org/fsd/release/FSD50K/

1) Weak Labels. We opt to label the dataset with weak labels. The main motivation is that gathering weak labels (i.e., at clip-level) is simpler, less time consuming and less ambiguous than determining events' onset/offset (i.e., strong labels). Weakly supervised learning has demonstrated effectiveness to learn sound event recognizers, both for classification and detection (McFee et al., 2018b). Nonetheless, using weak labels imply certain limitations on training and evaluation, which we highlight in Section 3.3.2.

2) Label Quality and Dataset Size. As mentioned earlier, the SER field has witnessed a transition away from small and exhaustively labeled datasets (e.g., Salamon et al. (2014); Piczak (2015b); Foster et al. (2015)), in favour of larger datasets that inevitably include less precise labelling, such as AudioSet (Gemmeke et al., 2017). This occurs mainly because it is not feasible to exhaustively annotate large amounts of sound event data. In our case, we want to seek a trade-off by prioritizing label quality while ensuring a certain amount of data. Yet, label noise problems also appear in FSD50K, as in any sound event dataset of a certain size (Section 3.3.3).

3) Emphasis on Evaluation Set. This is perhaps the design criteria that mostly determines the creation of FSD50K. Essentially, an evaluation set defines the target behavior in a recognition task, which makes it possibly the most critical part of a dataset. Consequently, having a comprehensive, diverse, reliably annotated, and real-world representative evaluation set is important for meaningful systems' benchmarking. The importance of reliable evaluation sets is highlighted by recent research in computer vision which focuses on improving the evaluation and/or validation sets of widely-used datasets. Examples include Barz & Denzler (2020) for CIFAR-10/-100 (Krizhevsky & Hinton, 2009), and Recht et al. (2019) and Beyer et al. (2020) for ImageNet (Deng et al., 2009). In addition, alternative learning paradigms to the traditional supervised learning (using reliably-labelled datasets) start to be promising nowadays. In particular, significant progress is being made in the development of sound event recognizers with noisy supervision (Fonseca et al., 2020b) or self-supervision (Jansen et al., 2018; Fonseca et al., 2021b). While these alternatives can minimize the problems of labelling inaccuracies in the development set, or the need for a labeled development set at all, a carefully curated evaluation set is still critical for benchmarking. Relatedly, abundant data resources for training are already available, either from AudioSet, or directly from web audio repositories such as Freesound or Flickr (provided appropriate learning strategies are used). By contrast, to our knowledge, large-vocabulary, carefully-curated evaluation benchmarks are rare—the most prominent being AudioSet's evaluation set, which suffers from issues of label noise, stability

and/or openness (as discussed in Section 2.4.1.2). By prioritizing the curation of the evaluation set, we contribute to fill this gap.

3.2.1.1 Gathering Reference Labels

To tackle the task of annotating the dataset, two approaches are considered: *i)* manual annotation and *ii)* semi-automatic methods based on *active learning*. Manual annotation is the conventional approach to dataset labelling, as done in AudioSet (Gemmeke et al., 2017) or ImageNet (Deng et al., 2009) in computer vision. While this option is very laborious and time consuming, when done properly we believe it leads to more reliable results than involving automatic methods in the annotation loop. Active learning is a promising alternative to reduce annotation effort. However, while progress has been made in the field of active learning for SER, most works focus on recognition tasks that are simpler than ours, as mentioned in Section 2.4.3.2. Specifically, previous works deal with less than a dozen classes, and most of them focus on single-label classification and use pre-labeled datasets, where the human annotation step is simulated by assigning pre-existing ground truth. Extending previous methods to a large-vocabulary, multi-label setting like ours is not trivial and hence it is considered out of the scope of this work.

Thus, in order to obtain a high-quality labelling, and being aware of the amount of data to annotate and the budget available, we decide to annotate the dataset manually, similarly as done with AudioSet. While this means a higher human effort, it presents two advantages. First, manually annotating FSD50K gives us a deeper insight into the data that would not have been gained otherwise. Second, it allows us to have a greater control of the labels gathered, as well as to specify not only the labels but also an estimate of sound predominance (as we will see in Section 3.2.5). Furthermore, obtaining a set of labels as reliable as possible for this first release is a more favorable starting point for potential future expansions, which could rely on (semi-) automatic methods to scale up more efficiently at the expense of label noise.

3.2.2 Overall Procedure

The overall process of the creation of FSD50K is illustrated in Figure 3.1, starting from Freesound and the AudioSet Ontology, and ending with FSD50K. In every intermediate stage, we progressively filter out a quantity of audio clips and classes in the vocabulary. Each stage is described in the next subsections.

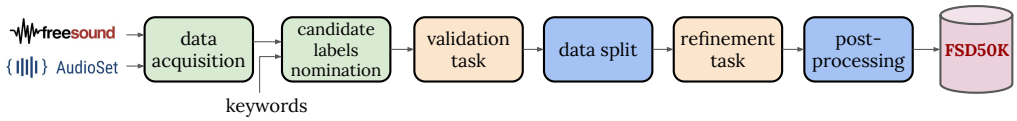

Figure 3.1: Overall process of the creation of FSD50K. The process starts from Freesound and the AudioSet Ontology. Stages in green involve automatic data mining, stages in orange correspond to manual annotation tasks, and stages in blue involve data processing to shape the dataset.

3.2.3 Data Acquisition

The starting point for the creation of FSD50K is an abundant source of audio clips, a vocabulary to annotate them, and an infrastructure where they can be loaded and annotation tasks can be carried out. These items correspond to Freesound, AudioSet Ontology and Freesound Annotator respectively. Freesound is described in Section 2.4.4, and the AudioSet Ontology is detailed in Section 2.2.4.1 Next, we briefly argue why the choice of the AudioSet Ontology for FSD50K, and briefly introduce Freesound Annotator.

3.2.3.1 AudioSet Ontology as Vocabulary for FSD50K

For the creation of FSD50K, we are interested mainly in common physical sound sources, related to the casual listening as defined by Schaeffer (1966), which tend to be more clearly defined and familiar to the average listener. We are less interested in perceptual sound attributes (e.g., bright or deep) or other sound classes that can be more subjective or cultural-dependent, which can lead to ambiguity (e.g., onomatopoeia, musical concepts, or very specific sound production mechanisms). Our motivation lies in several reasons. First, recordings of these more typical and better defined classes tend to be more available in Freesound. In addition, the tags used to describe these recordings may be less heterogeneous, which is convenient as we use them in the dataset creation process. Second, our goal is to allow training of general-purpose classifiers able to recognize the sound events of the most representative and typical classes of different domains (e.g, human, domestic, urban, or nature

sounds). Third, typical and familiar sounds are easier to annotate manually as humans can more easily form a mental picture of the sound (Ballas, 1993). Consequently, choosing this type of sound classes may presumably require less annotation effort and may lead to less label noise than dealing with more ambiguous classes.

Developing an *ad hoc* taxonomy is a tedious task that requires multidisciplinary knowledge and substantial effort, hence it is considered beyond the scope of this thesis. Consequently, for the creation of FSD50K we decided to adopt a subset of the AudioSet Ontology as it is the most comprehensive set of everyday sounds available, which makes it convenient to cover Freesound's heterogeneity. In addition, the rapid acceptance of AudioSet as a resource for SER research has made the AudioSet Ontology a *de facto* standard for everyday sound organization in the academia. Yet, upon careful inspection of the ontology, we realize that improvements could be made in order to make it more consistent as a resource for everyday sound vocabulary, and to make it more suitable for organization of Freesound audio. However, this task was found to be far from trivial and considered beyond the scope of this work. In particular, for FSD50K we focus on a subset of the ontology oriented to the most common physical sound sources and a few production mechanisms, and less oriented to ambiguous or less represented classes in common everyday situations.

3.2.3.2 Freesound Annotator

Freesound Annotator[27] is a website that allows the collaborative creation and curation of open audio datasets based on Freesound content. It serves mainly two goals: the management and exploration of datasets, and the creation and verification of annotations. Originally released on 2017 as the *Freesound Datasets* platform, Freesound Annotator has been the object of continuous development (Fonseca et al., 2017b). The name of the platform was changed to avoid potential confusions with other datasets. Freesound Annotator started by providing basic prototypes for exploring a taxonomy of audio classes and validating automatically generated annotations. Additional features were incorporated progressively, including annotation tools and quality control mechanisms (see Sections 3.2.5 and Section 3.2.7). Monitoring tools allow inspection of a dataset progress as well as debugging capabilities. Freesound Annotator is an open-source project[28] developed and maintained mainly by Xavier Favory and Frederic Font, with significant contributions from the author of this thesis.

[27] https://annotator.freesound.org/
[28] https://github.com/MTG/freesound-datasets/

3.2.4 Candidate Labels Nomination

We started building FSD50K by automatically populating the classes of the ontology with a number of candidate audio clips from Freesound. Candidate clips were selected by matching user-provided tags in Freesound to a set of keywords associated with every class. The goal was to automatically compile a list of candidate labels per clip, indicating potential presence of sound events. The process consisted of two steps.

First, we compiled a list of keywords for almost every class. These are terms related to the class label that are likely to be provided by Freesound users as tags when describing audio clips. Suitable keywords were determined by considering class names and descriptions provided in the ontology, and obtaining the most frequent Freesound tags that co-occur with each target class label. After compiling a first version of the per-class keywords, we manually identified a few classes with very low precision due to pathological inclusion of false positives. For example, in the *Turkey* class many clips were recordings made in the Eurasian country, instead of containing sounds of the large bird. To minimize this issue, a refinement process was performed by blocking some tags (e.g., "turkish" or "Istanbul" for the *Turkey* class). As an example, the keywords for the *Meow* class are: "meow", "meowing", "mew", "miaow", and "miaou".

Second, each class was automatically populated with the corresponding Freesound clips. We use the compiled lists of keywords as a mapping between clips in Freesound and class labels in the ontology. Thus, for each clip, all user-provided tags are examined and, when a tag matches a keyword, the clip becomes a candidate clip for the dataset, and the corresponding class label is nominated as a candidate label for the clip. This process was done by using the Freesound API.[29] We employ the Porter Stemming algorithm for term normalisation to make our matching process more robust (Porter et al., 1980).

In this way we were able to map more than 300,000 Freesound clips to the AudioSet classes. We decided to filter out clips longer than 90s to avoid very large audio clips (this length limit will be further reduced later on, see Section 3.2.5). No other filters were applied at this stage. This left us with a total of 268,261 clips with an average of 2.62 candidate labels. This label nomination system induces potential errors as it depends on factors such as class ambiguity and, especially, the choices of Freesound users when providing tags. However, it has the advantage of allowing easy and rapid retrieval for a large variety of classes without training any classifiers.

The outcome of this stage is a list of automatically-generated candidate labels per clip, indicating the potential presence of sound events.

[29] https://freesound.org/docs/api/

3.2.5 Validation Task

The goal of this stage is to manually validate the candidate labels nominated in the previous stage.

3.2.5.1 Initial Prototype of the Annotation Tool

To this end, we designed and implemented an annotation tool that was deployed in Freesound Annotator. Essentially, human raters are presented with a number of audio clips and, for each clip, they must assess the presence of a given sound class. For each class, the annotation process consists of two phases:

1. a **training phase** where raters get familiar with the class by looking at its location in the AudioSet Ontology hierarchy, the provided textual description, and representative sound examples.

2. a **validation phase**, in which raters are presented with a series of audio clips from that class (up to 72 clips in 6 *pages* of 12 clips) and prompted the question: *Is <class> present in the following sounds?*. (Figure 3.3 shows 3 clips within a page of the final prototype). In this initial prototype, raters must select among "Present", "Not Present", and "Unsure", similarly as done in Gemmeke et al. (2017).

Along with an audio player and its waveform, links to each clip's Freesound page were made available, where the original tags and descriptions could be inspected to aid the process.

3.2.5.2 Internal Quality Assessment

We used the initial prototype to run an Internal Quality Assessment (IQA) with the goal of *i)* assessing the quality of the candidates produced by the nomination system, and *ii)* collecting feedback about the prototype and annotation task for improvements. The IQA consisted of validating 12 candidates (1 page of clips) for every class, covering all classes available. It was carried out by 11 subjects that volunteered to participate, who could leave per-class comments through a text box. The feedback collected in the IQA revealed that the annotation task is complex due to factors such as ambiguity in some class descriptions or the difficulty of annotating sound events with very high inter- and intra-class variation.

3.2.5.3 Final Prototype of the Annotation Tool

Based on the insight from the IQA, we designed the final annotation tool (Figures 3.2 and 3.3) which incorporates the following improvements with respect to the previous version:

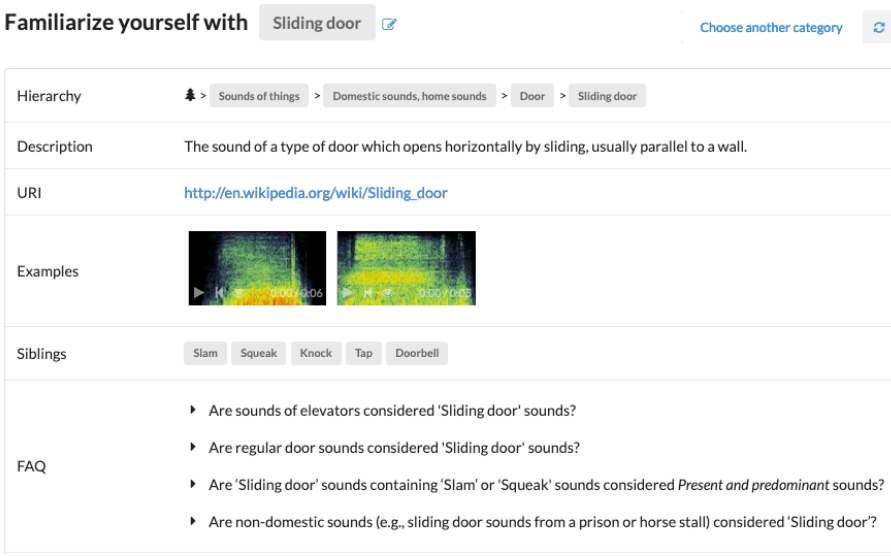

Figure 3.2: Screenshot of the "Training phase" page used for the validation task.

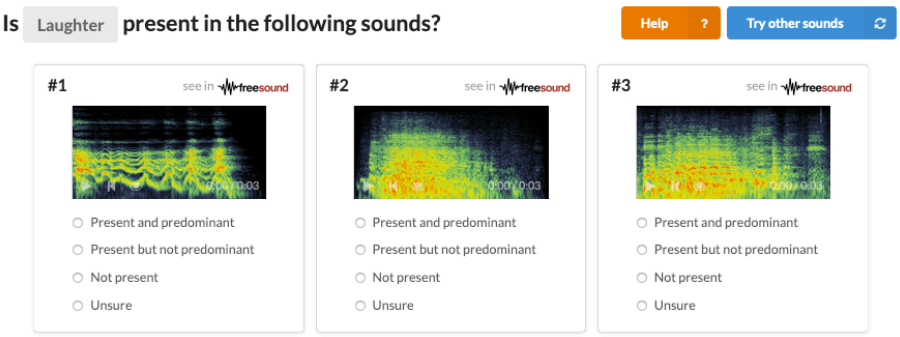

Figure 3.3: Screenshot of the "Validation phase" used for the validation task.

- Some AudioSet class descriptions were found to be ambiguous, allowing multiple interpretations and generating doubts as to the class scope. We decided to include a list of **Frequently Asked Questions (FAQs)** in each class description to help homogenize raters' judgment and gather

more consistent annotations (see Figure 3.2). The full FAQs list is provided with the dataset.

- In some audio clips, several sound events co-existed with different predominance or salience levels, making the "Present" response rather ambiguous for raters. To address this issue, we decided to **split the "Present" response** into "Present and predominant" (PP) and "Present but not predominant" (PNP), as specified in Table 3.1.[30] A similar approach was used by Salamon et al. (2014). The main motivation is to ease the annotation task by mitigating a systematic doubt. As an additional benefit, this distinction allows to separate a subset of clips containing mostly isolated and clean sound events (PP ratings) vs. others featuring events from several classes and/or in more adverse acoustic conditions (PNP ratings). This could allow defining robustness tasks such as training or evaluating with a subset of data of more adverse conditions, similarly as done in Serizel et al. (2020) for SED or with ImageNet-A for image recognition (Hendrycks et al., 2019b). Further, the PP/PNP distinction can be useful for source separation studies (Wisdom et al., 2021). We note, however, that this distinction is subjective and these ratings should be used as a rough indication.

- To automatically assess the reliability of the submitted responses, we added **quality control mechanisms** such as the periodic inclusion of verification clips. Whenever the response for one of these clips is wrong, the responses submitted in a given time span are discarded—a common practice in crowdsourcing platforms.

- To further ensure high quality annotations, we decided to require **inter-annotator agreement**. More specifically, each candidate label is presented to several raters until two different raters agree on a response type. Once an inter-annotator agreement is reached, the label is considered as *ground-truth* and it is no longer presented to other raters. A similar practice is done in Foster et al. (2015); Cartwright et al. (2019c).

- To facilitate the localisation and recognition of sound events within the audio clips, we added **spectrogram visualizations**, thereby easing the annotation task (Cartwright et al., 2017) (the initial prototype featured less-informative waveforms).

- Some audio clips can present highly variable loudness, which can be burdensome for the rater and may affect annotation quality. To mitigate this problem, we **normalize the loudness** of the sound files following EBU Recommendation R 128 (2014).

[30] Hereafter, we shall use "Present" to refer to the union of PP and PNP.

Table 3.1: Response types for the validation task.

Response Type	Meaning
Present and predominant (PP)	The type of sound described is **clearly present** and **predominant**. This means there are no other types of sound, with the exception of low/mild background noise.
Present but not predominant (PNP)	The type of sound described is **present**, but the audio clip also **contains other salient types of sound and/or strong background noise**.
Not Present (NP)	The type of sound described is **not present** in the audio clip.
Unsure (U)	**I am not sure** whether the type of sound described is present or not.

- To select which audio clips to present to each rater, we adopt a **prioritization scheme** that ranks clips according to two criteria: *i)* previously rated label-clip pairs that have not yet reached inter-annotator agreement are prioritized to obtain ground truth labels; *ii)* short clips are prioritized over long ones as shorter clips have a higher label density—considered more informative for learning.

Beyond these improvements, we took two additional measures to improve annotation efficiency. First, the selection of candidates in a number of classes had a very low precision possibly due to sub-optimality of the nomination system. Thus, we decided to discard classes with a rate of "Not Present" responses above 75%, as well as classes with very few candidates and others deemed highly ambiguous for annotation. This left a total of 395 sound classes (a reduction of ≈35%). Second, participants reported that the initial duration limit of 90s was too long for human validation, and a potential cause for fatigue. In addition, the supervision given by weak labels applied to such large lengths is rather vague. Therefore, we decided to discard clips longer than 30s.

3.2.5.4 Annotation Campaign

With the final annotation tool, we launched an annotation campaign to validate the candidate labels at scale. Given that some classes were found to be much more difficult to annotate than others, we decided to gather annotations using both crowdsourcing and hired raters. We divided the classes according to an estimated level of difficulty, based on feedback from the IQA. Table 3.2 lists the annotation strategies adopted for each subset of classes. Crowdsourcing

Table 3.2: Annotation strategies in the validation task.

Class Difficulty	Example	Classes	Annotation Strategy
easy	*Bark*	77	crowdsourcing & hired annotators
medium	*Piano*	100	crowdsourcing & hired annotators
difficult	*Tearing*	218	hired annotators

consists of gathering validations contributed by any voluntary participant. We made the classes of easy and medium difficulty publicly accessible from Freesound Annotator,[31] which was promoted in Freesound forums and social media. The most difficult classes, where certain annotation experience was deemed important to provide reliable responses, were kept private. They were validated by a pool of hired raters who also complemented the crowdsourcing validations in the rest of the classes.

In total, over 350 raters contributed, including voluntary participants, six hired raters, and three PhD students from our research group including the author of this thesis (specifically, Xavier Favory, Jordi Pons, and Eduardo Fonseca). The hired raters were subjects with background in audiovisual engineering, including mostly MSc students from our group, with self-reported healthy hearing. We opted for a small pool of raters in order to have more control over the annotation process and to obtain annotations that are as consistent as possible. We recognize this may induce a certain bias, but we rather have consistent annotations with agreed bias than a certain lack of consistency likely resulting from crowdsourcing annotations for the difficult classes (i.e., label noise). To this end, the hired raters were trained and closely monitored by the authors, discussing doubts and agreeing on the best course of action. The list of FAQs were gradually extended as more insight was obtained. For consistency, raters were asked to validate groups of related classes (e.g., sibling categories), so they could get familiar with specific sections of the ontology (Sabou et al., 2014). They were instructed to perform the task using high-quality headphones in a quiet environment, and taking periodic breaks to mitigate fatigue. During the campaign, the hired raters acquired solid expertise on the annotation task and a deep knowledge of the ontology. Therefore, we consider them experts for this task.

The outcome of the annotation campaign was 51,684 clips considered valid for the dataset, that is, with at least one "Present" label. All the "Present" labels amount to 59,981, all of them being the result of inter-annotator agreement, except 3390 which include labels with: *i)* only one PP rating and one PNP rating (and nothing else). This can be considered inter-annotator agreement

[31] https://annotator.freesound.org/fsd/annotate/

at the "Present" level; *ii)* only one PP rating (and nothing else); *iii)* only one PNP rating (and nothing else). The two latter do not meet our definition of ground truth and could be more prone to errors, but were still considered to slightly increase the amount of data. It must be noted that the set of labels at this point comes from the validation of candidate labels proposed by a simple nomination system, which ultimately relies on the user-provided Freesound tags. Hence, it is to be expected that some sound events are not covered by the user-generated tags, or they are not proposed by the nomination system, leading to missing "Present" labels, a common phenomenon in large sound event datasets (Meire et al., 2019; Fonseca et al., 2020b). That is, the resulting pool of audio clips have labels that are mostly correct (estimated at 94.3% in Section 3.3.3), albeit potentially incomplete which is problematic in evaluation. To address this issue, after splitting the data into development and evaluation sets (Section 3.2.6), the latter is refined using another annotation tool (Section 3.2.7).

3.2.6 Data Split

The input to this stage is a pool of 51,684 audio clips with mostly correct labels (albeit potentially incomplete). The goal is to split the data into two subsets: *development* and *evaluation*. The development set will be used for training and validation. The evaluation set will be used for system benchmarking after exhaustive annotation. As stated in Section 3.2.1, the evaluation set is our priority. A high quality evaluation set must be comprehensive, varied, and representative (Mesaros et al., 2018b), while being free from contamination from the development set in order to allow testing models' generalization capabilities.

3.2.6.1 Split Criteria

We set four criteria for the split.

Non-divisibility of Uploaders. The issue of *contamination* must be considered when splitting audio data, especially if portions of the data share a common pattern that brings acoustic similarity among its constituents. In Freesound, audio content is uploaded by users (in the following, *uploaders*). The uploaders can be very diverse: some are *small*—they upload a small number of audio clips (e.g., up to only few tens)—while other uploaders contribute with hundreds of clips. In the latter case, it can happen that some of the uploaded clips share the same sound source and/or physical location and/or recording gear (e.g., several notes of the same music instrument or vocalizations of the same pet). If some of these recordings are used for training and others

for evaluation, their similarity may lead to overly optimistic performance, reflecting the classifier's ability to overfit development examples. As a result, this classifier may suffer from performance drop when tested on unseen data. This issue can be called *weak contamination* between development and evaluation, although, for simplicity, we will refer to it as *contamination* hereafter.[32] This phenomenon has been detected in computer vision benchmarks like CIFAR-10 and CIFAR-100 (Barz & Denzler, 2020). Another example of this in the field of music recognition is the denominated "album effect" (Whitman et al., 2001; Mandel & Ellis, 2005) or "artist effect" (Flexer & Schnitzer, 2009). Another case of contamination happens when a group of clips captured with the same sensor is split in training and evaluation (Cartwright et al., 2019c). To avoid this issue, we make sure that *all* the content of each uploader is allocated either in the development or evaluation set. By doing this we assume that the evaluation performance reflects the model's ability to generalize to new audio material and recording conditions.

Small Uploaders for Evaluation. To obtain a varied evaluation set, it seems reasonable to allocate the content from small uploaders as it guarantees a higher diversity of sound sources, acoustic environments and recording equipment. In addition, a closer look at the Freesound data distribution revealed that recordings uploaded by small uploaders tend to be slightly longer. It can therefore be expected that, in general, these longer recordings tend to contain more sound events when compared to shorter clips—a considerable portion of Freesound consists of short clips of few seconds featuring a single event. Under this assumption, longer recordings would be more real-world representative (see Section 3.3.2). Also, this is a more interesting content to further annotate exhaustively, and also with timestamps to allow future SED evaluations. Finally, another benefit of using small users for evaluation occurs in the hypothetical worst case scenario of one uploader contributing very similar clips. In this case, it is likely that a competitive model predicting one of them correctly would predict correctly many of them, thus generating a fake performance resolution in the given class(es). By selecting small uploaders we mitigate this problem.

A Coarse Class Distribution is Enough. A fine-level split carefully matching a target class distribution is not needed at this point, as during the exhaustive labelling of the evaluation set we expect some classes to grow (Section 3.2.7). This will create an imbalance that will need to be compensated.

[32] This should not be confused with *data leakage*, which happens when the *same* (not similar) examples are used for both training and evaluation.

Focus on Leaf Nodes. Among the classes available at this point, we focus on the subset of 113 leaf nodes with more than 100 clips as they are considered the most important classes.

3.2.6.2 Split Method

Given the many constraints, off-the-shelf methods such as random sampling, iterative stratification (Sechidis et al., 2011) or combinatorial optimization algorithms like knapsack problems (Toth & Martello, 1990; Cramer et al., 2020) are not well suited. Therefore, we implement an *ad hoc* approach consisting of iteratively allocating uploaders' content to the evaluation set after sorting them appropriately. First, we compute a score per uploader u as:

$$\text{score}^u = \text{n\_labels}^u_{\text{max}} + \frac{1}{K_u} \sum_{k=1}^{K_u} \text{n\_labels}^u_{c_k}, \tag{3.1}$$

where $\text{n\_labels}^u_{\text{max}}$ is the maximum number of labels provided by the uploader u in any class, $\text{n\_labels}^u_{c_k}$ is the number of labels provided by u in the class c_k, and K_u is the number of classes *touched* by u (i.e., those to which u contributes). Uploaders are sorted in ascending score order and the content of low-score uploaders is transferred first. With the first term we prevent uploaders with abundant content concentrated in one specific class, and with the second term preference is given to users with low average number of labels per class for diversity. We found out that, by splitting the target 113 leaf nodes, some content corresponding to the remaining classes is automatically allocated due to the uploaders contributing also to them. This content is deemed sufficient as a fine-level class distribution is not the target at this point. We then proceed to allocate data to the evaluation set following the process shown in Algorithm 1.

We traverse the $C = 113$ classes starting from the least-represented ones since they have less flexibility for data allocation. For each class c_i, we progressively allocate content from the ranked uploaders $\boldsymbol{u}$ until a target amount of data t_{c_i} is reached. t_{c_i} is proportional to the total class label count, and rectified to lie in the range from 50 to 100 labels per class. By default, the maximum uploader size per class c_i (i.e., the maximum number of clips that one uploader u is allowed to contribute to c_i) is set to $0.1t_{c_i}$. Thanks to the proposed sorting, uploaders in the evaluation set do not reach such a maximum in the majority of classes (they often provide one single clip)—if they do, excess clips are discarded in most cases. However, due to the high uploader diversity, the maximum uploader size had to be increased in a few exceptions. Using the proposed scheme, we processed all the 7229 uploaders and we allocated 2794 of them to the evaluation set, totalling 11,466 clips.

Algorithm 1: Data allocation to evaluation set

Data: Empty evaluation set per-class $E = \{e_{c_i} = 0\}_{i=1}^{C}$, uploaders ranking $\boldsymbol{u}$

```
for class c_i ∈ C do
    get current evaluation target t_{c_i}
    while e_{c_i} < t_{c_i} do
        get next uploader u in ranking u with data in c_i
        e_{c_i} ← e_{c_i} + data from u in c_i
        for class c_k ∈ K_u do
            e_{c_k} ← e_{c_k} + data from u in c_k
        end
    end
end
```

Result: A candidate evaluation set

The result of this stage is two pools of clips that are disjoint in terms of uploaders: a candidate development set and a candidate evaluation set. The latter is exhaustively labeled in the refinement task.

3.2.7 Refinement Task

As mentioned in Section 3.2.5, in some clips, the current label sets could be an underrepresentation of the audio content, biased by the idiosyncrasies of the labeling pipeline. This is problematic in evaluation, as classifiers would be penalized when predicting a correct label that happens to be missing from the ground truth. This critical issue would limit the utility of the dataset for system's benchmarking. To address this issue, we refine the labels in the evaluation set. The goal is to obtain an *exhaustive* labelling, that is, a labelling as close as possible to the correct and complete transcription of the audio content (for the considered vocabulary of 395 classes).

3.2.7.1 Annotation Tool

We designed an annotation tool that allows two subtasks: *i)* to review the existing labels, and *ii)* to add missing "Present" labels. The subtask of adding missing "Present" labels has a considerable complexity since audio clips can sometimes contain unrelated sound events. Therefore, the success of this task relies on two key factors: *i)* raters with a deep understanding of the ontology, the agreed FAQs, and the particularities of the audio material; *ii)* an interface that facilitates exploration of the large vocabulary of the ontology. In regard to the first factor we turn to the pool of hired raters (4 of the initial 6 as

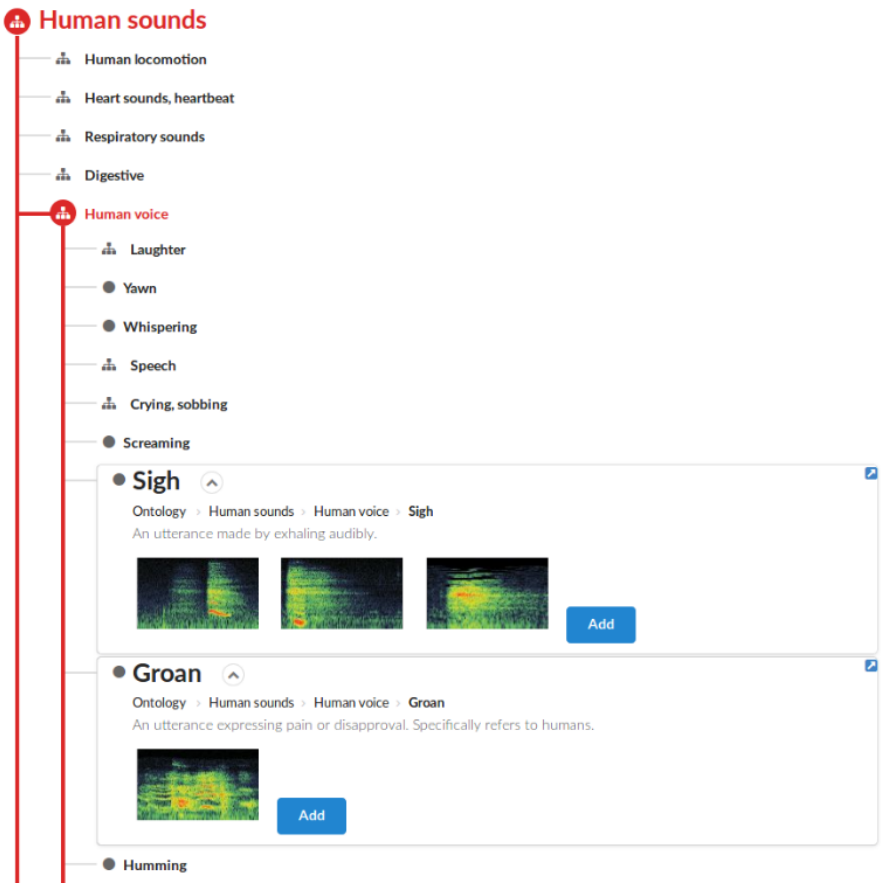

Figure 3.4: Table for exploring the ontology in the refinement task.

the others became unavailable), who acquired a solid expertise by extensive participation in the validation task (Section 3.2.5). As for the second factor, the refinement task we implemented in Freesound Annotator includes a tool to interactively explore different depth levels of the ontology (Figure 3.4).

This tool is based on a previous version described in Favory et al. (2018). A search input box allows to quickly navigate to classes in the table, where their hierarchical context is shown. For each class, textual descriptions and representative sound examples are displayed. The interface facilitates the comparison of different classes by simultaneously displaying their information, which was found useful among the raters.

3.2.7.2 Annotation Process

Clips were presented grouped by sound class to facilitate the task. For every class:

1. raters were instructed to go through a **training phase** (same as in the validation task—see Figure 3.2).

2. For every clip, they would first **review existing labels** and modify them if needed. Then, they would **add any missing labels** by exploring the ontology (Figure 3.4).

Raters were instructed to provide the most specific labels possible (typically leaf labels) as they are the most informative type of supervision. The quality assurance practices described for the validation task were also applied in the refinement task. Following this procedure, each evaluation label was verified or reviewed by between two and five different annotators (considering both validation and refinement tasks), including at least one expert. As a result, labels are expected to be correct and complete in the vast majority of cases (see Section 3.3.3 for a discussion on label noise). The exhaustive labelling carried out has two implications. First, absence of labels means absence of sound events (except human error)—a desired feature. Second, some classes are now much more represented than before as they are prevalent but were underrepresented, thus creating a class imbalance.

The outcome of this stage is a pool of exhaustively labeled clips (for the considered vocabulary), the majority of which will form the evaluation set.

3.2.8 Post-processing

This stage starts from two sets of data: a candidate development set with correct but potentially incomplete labels, and an exhaustively-labeled candidate evaluation set. The vocabulary used so far comprises 395 classes, yet many of them have few data (few tens of clips). While they may not be adequate for deep learning approaches, they can be useful for other practices requiring less data (e.g., few shot learning (Cheng et al., 2019)). Likewise, this information can provide insight as to the specific content of the dataset. Therefore, we provide two different formats for the annotations in FSD50K:

1. The raw outcome of the annotation process, featuring all generated class labels without any restriction. These include classes with few data. We call this the sound *collection* format.

2. The outcome of curating the raw annotations into a machine learning dataset with emphasis in sound event recognition tasks. This process involves, mainly, merging low prior classes into their parents thus ensuring a minimum amount of per-class data. This is what we define as *ground truth* for FSD50K, with a vocabulary of 200 classes.

Next, we explain the post-processing carried out to obtain what's finally released as FSD50K (consisting of a set of audio clips and the corresponding ground truth). Further technical details about the sound *collection* format can be found in FSD50K's Zenodo page.[24]

3.2.8.1 Determine FSD50K Vocabulary

We define *valid* leaf nodes as those meeting two requirements: a minimum of 100 clips and without extreme development/evaluation imbalance. This is a trade-off between abundant per-class data and preserving a lot of leaf nodes.[33] We take the following measures.

Merge Non-valid Leaf Nodes with Their Parents. There are two variants of this process, depending on the type of branch in the hierarchy. First, non-valid siblings of valid leaf nodes are merged with their parents. In these branches, the level of specificity is fixed by the valid sibling. For instance, *Yip*, a class with few data which is sibling of *Bark* and child of *Dog*, is merged with *Dog* and the most specific label in this branch is the valid leaf *Bark*. Then, in branches without any valid leaf nodes, all leaf nodes are merged with their parents, which in turn become new leaf nodes (since they no longer have children). This process is repeated recursively, pruning the branch by moving upwards in the hierarchy, until a new leaf node becomes valid. While we ideally want to prune the branches as little as possible to preserve the most specific nodes, some low-level nodes are inevitably merged with non-specific parents, e.g., *Domestic sounds, home sounds*. The minimum data requirement is enforced at the leaf node of every branch, but not at its ancestors, which are intrinsically valid because the leaf node provides enough data. This means that, occasionally, the data explicitly associated with one ancestor may be scarce. This is due to the nomination system and annotation processes, which favour more specific labels. For example, *Rail transport* is a valid intermediate node despite the fact that it holds very few data, because their children (*Train* and *Subway, metro, underground*) are valid and hold most of the data in this branch.

[33] Given the particularities of some classes, the requirements to consider a leaf node *valid* are relaxed in a few exceptions.

Remove Some Valid Leaf Nodes to Obtain a More Semantically Consistent Vocabulary. As a result of the pruning, some parents with various children in the ontology end up having very few children in the candidate dataset. In most cases, this is not a problem as children are rather independent semantically—e.g., the class *Domestic sounds, home sounds* encompasses 27 children that occur usually in domestic contexts but with weak semantic links otherwise. However, in other cases, children constitute a pre-established subset of closely related classes that makes more sense when all of them co-exist, e.g., the classes *Light engine (high frequency)*, *Medium engine (mid frequency)*, and *Heavy engine (low frequency)*, where only the former is valid. Considering the real operation of trained models, the fact that only one of these children is valid could potentially lead to unnatural predictions biased by the choice of the vocabulary, i.e., recurrently spiking *Light engine (high frequency)*, absent their complementary siblings. To prevent this issue, we merge some "isolated" valid leaf nodes with their parents, thus obtaining a more semantically consistent vocabulary. Hence, despite having a substantial number of light engine sounds, they are not part of the vocabulary—only *Engine* is. Note however that these more specific annotations are indeed available in the sound collection format. Apart from the mentioned engine types, this also occurs with wind instruments and bowed string instruments.

Discard Some Intermediate Nodes. This includes classes of abstract nature or with ambiguous children and few data, e.g., *Digestive* or *Arrow*, respectively. The outcome is a vocabulary of 200 classes (144 leaf nodes and 56 intermediate nodes).

3.2.8.2 Balancing Development/Evaluation Sets

As a result of exhaustively labelling the evaluation set, the proportion of some frequently occurring sound events increased substantially, sometimes exceeding the number of labels in the development set. To obtain a better balance between development and evaluation sets, we first identified a set of 40 leaf nodes which benefit from transferring data from evaluation to development. Then, we selected a set of evaluation clips such that: *i)* their content encompasses mainly the 40 target classes with a minimal impact on the remaining ones—note the clips are multilabel; *ii)* they are disjoint from the remaining set of clips in terms of uploaders. Specifically, we transferred 1182 clips, resulting in an evaluation set of 10,231 clips, and a per-class development/evaluation proportion ranging from 50/50% to 75/25% in the vast majority of leaf nodes. The per-class split proportion depends on data availability, ubiquity of the sound events, degree of multilabelness of the audio clips, and non-divisibility of content from the same uploader. Exceptions include *Chatter*, *Chirp, tweet*

and *Male speech, man speaking*, for which there are more evaluation than development labels due to the exhaustive labelling of these ubiquitous events. With this transfer, we also make available some exhaustively labeled content for validation.

3.2.8.3 Validation Set

Some recent large audio datasets do not provide predefined validation sets (Gemmeke et al., 2017; Mesaros et al., 2018d) allowing dataset users to create their own. Nonetheless, for easier dataset consumption and reproducibility we propose a candidate split of the development set into train and validation. We consider that a validation set should ideally meet the following criteria:

- **Proportion**. The validation set typically amounts to a given proportion of the development set, often between 10 and 20%. Note that due to the multilabel and variable-length nature of Freesound audio, the proportion can be different in terms of audio clips, labels, and duration.
- **Stratification**. It is usually desirable that the class label distribution is similar in both train and validation sets.
- **Contamination**. As explained in Section 3.2.6, contamination across splits should be minimized.

Typical ways to make train/validation splits include random sampling or iterative stratification (Sechidis et al., 2011). Both can produce desired data proportions and class distributions, the latter being popular for multilabel data.[34] However, they fail to keep non-divisibility of uploaders' content, thus generating contamination. The distribution of number of clips per uploader is very varied in the development set. However, since we already allocated a large amount of small uploaders into the evaluation set (Section 3.2.6), preserving uploader non-divisibility at this point means deviating from the target class distribution. In other words, it is difficult to strictly meet the three above criteria simultaneously, hence we need to relax their application.

We focus on the contamination criteria and distinguish two types of contamination: *i) within-class* contamination (WC, when content from the same uploader *and* belonging to the same class is placed at both train and validation sets); *ii) between-class* contamination (BC, when content from the same uploader *but not* from the same class is placed at both train and validation sets). We hypothesize WC is more harmful as it could imply having the same sound

[34]Random sampling does not account for stratification per se, but a workaround is to compute many train/validation splits and choose the one that minimizes a distance between the respective class distributions.

source, physical location and/or recording gear in both sets. By contrast, BC would have less impact as, in most cases, the audio material would be different, and also possibly the acoustic environment. Under this hypothesis, we focus on minimizing WC contamination while being flexible with BC. To do this, we employ a method similar to that of Section 3.2.6. We first define the content from one uploader labeled with the same class label as the minimum non-divisible unit. Then, we adopt an iterative process in which, after sorting the uploaders per-class appropriately, we progressively allocate their content to the validation set.

As preprocessing, we initialize the validation set with most of the data transferred from evaluation to development—this content is well suited for evaluation purposes as it is exhaustively labeled. We then compute a score per uploader and per class. The score for uploader u in class c_i is given by:

$$\text{score}_{c_i}^{u} = \alpha \text{n\_labels}_{c_i}^{u} + \beta \frac{1}{K_u} \sum_{k=1}^{K_u} \text{n\_labels}_{c_k}^{u}, \tag{3.2}$$

where $\text{n\_labels}_{c_i}^{u}$ represents the number of labels provided by uploader u in class c_i, K_u is the number of classes touched by u, and α and β are tunable weights to set the relevance of each term, both $\in [0,1]$. The first term is the amount of data in c_i by u, whereas the second term is the average number of labels per class, accounting for the scattering of u across classes. Uploaders are sorted in ascending score order and the content of low-score uploaders is transferred first. By tuning α and β we aim to promote the uploaders providing a small amount of data in the class under question, c_i, with minimal or no scattering. This facilitates the adjustment to a target class distribution while minimizing contamination (both WC and BC). This first group of uploaders is followed by others with smooth scattering across classes, avoiding uploaders with large contributions concentrated in specific classes. This again facilitates adjusting to a target distribution while minimizing the need to split content from the same uploader in one class (i.e., WC contamination), but allowing BC contamination.

Once the validation set is initialized and the uploaders are sorted per-class, we allocate data to the validation set as shown in Algorithm 2.

We traverse the classes in several passes, and, for each class c_i, we progressively allocate content from the ranked uploaders until a target data amount t_{c_i} is reached. Note that when separating the class c_i, the algorithm does not care about a given uploader u contributing to another class c_j (BC contamination), unless there is at least one clip bearing labels for both c_i and c_j. WC contamination can be produced in lines 6 and 8. We designed the step in line 6 so that, if adding the content from u implies exceeding the validation target t_{c_i} by more than 15%, two things can happen. If the current validation amount

Algorithm 2: Data allocation to validation set

Data: Initialized validation data per-class $V = \{v_{c_i}\}_{i=1}^{C}$, uploaders ranking in development set per-class $U = \{\boldsymbol{u}_{c_i}\}_{i=1}^{C}$

```
1  for pass n = 1,2,...N do
2      for class c_i ∈ C do
3          get current validation target t_{c_i}
4          while v_{c_i} < t_{c_i} do
5              get next uploader u in ranking u_{c_i}
6              v_{c_i} ← v_{c_i} + data from u in c_i
7              if data is multilabel to class c_j then
8                  v_{c_j} ← v_{c_j} + data from u in c_j
9              end
10         end
11     end
12 end
```

Result: A candidate validation set

is $v_{c_i} > 0.75t_{c_i}$, the content is not transferred, v_{c_i} is deemed sufficient and the procedure halted for c_i. This flexibility allows the minimization of WC contamination at the expense of deteriorating stratification. Else, if $v_{c_i} <= 0.75t_{c_i}$, the content from u in c_i is split and the amount needed to reach t_{c_i} is allocated, causing WC contamination. Similar heuristics are adopted for the step in line 8.

Using the proposed scheme, we process clips from the 4936 uploaders in the development set. Due to the high variability of users, the process needs initial debugging with a subset of classes in order to tune the weights α and β. We finally use $\alpha = 0.4$ and $\beta = 0$ when an uploader contributes only to one class, and $\alpha = 0.3$ and $\beta = 0.7$ otherwise. We use $N = 2$ passes starting from classes in need of more validation data, which allows us to reach a reasonable stratification. The target validation proportion is 15% of the development labels per-class, except for the largest 17 classes where we reduced this percentage progressively. The first-pass target is to fill 60% of the 15%-target, which is the goal in the second-pass. We only consider the leaf nodes for this process ($C = 144$). This is done for simplicity and because the leaf nodes are the most specific data that will receive labels from the rest of the ontology levels upon propagation to their ancestors. In this way, validation data at all levels of the ontology is guaranteed.

The outcome is a validation set which represents a tradeoff between stratification and contamination. Composed of 4170 audio clips, it amounts to 13.3% of the content associated with leaf nodes and 10.2% of the entire development set. Its main statistics are listed in Table 3.3.

Table 3.3: Main statistics for candidate validation set.

Clips	Duration	Uploaders
4170	9.9h	2224

Out of the 2224 uploaders with content in the validation set, 641 also have content in the train set—mostly corresponding to BC contamination. Section 3.4.2 describes sound event tagging experiments comparing the proposed split to others obtained with off-the-shelf split approaches. This candidate split is the result of a number of design choices. However, other choices might be desirable (e.g., proportion, contamination, usage of intermediate nodes, etc.) depending on researchers' needs. Alternative validation sets can be created using the clip metadata provided in FSD50K, which includes uploader information.

3.2.8.4 Ground Truth Hierarchical Propagation

At this point, the labels in train, validation and evaluation sets are usually from classes corresponding to lower levels of the ontology, especially for the evaluation set (see Section 3.2.7). To obtain an exhaustive labelling hierarchy-wise, we need to propagate the current labels in the upwards direction to the root of the ontology, determining the ancestors in the hierarchical path and automatically assigning them to the corresponding audio clips. This label propagation process is referred to as *label smearing* in Gao et al. (2017) and Hershey et al. (2021). We describe this process next.

In most cases, this is straightforward as there is one single unequivocal path from a given low-level node to the root. However, in other cases, nodes and root are connected by more than one path. Among these multiple-path cases, some have all the paths valid by default according to the semantics of the node. This allows straightforward propagation as in the single-path case, e.g., *Doorbell* can be directly propagated to *Door* and *Alarm*. However, in the majority of cases, only a subset of the paths is valid (often only one path), or even none of the paths is valid by default due to the parents-node relationship. For instance, *Buzz* cannot be directly propagated to its parents *Fly, housefly* or *Bee, wasp, etc.* unless we have explicit information about the source of the buzz sound. In these cases, we need knowledge of the correct immediate parent(s) to unambiguously infer ancestors for a complete hierarchical labelling. Parents disambiguation can be carried out in different ways depending on the annotation task. In the clips annotated only with the validation task, the disambiguating parents will exist if and only if the nomination system pro-

posed them. For the clips annotated also with the refinement task, raters were instructed to specify the disambiguating parents when needed; however, we detected that they were not always specified.

As a result, in these cases, ancestors cannot be inferred from the leaf node, leading to hierarchical paths featuring missing parts. For example, *Growling* is connected directly to *Animal* in several cases where information of the source animal is not available. The policy followed in case of ambiguous ancestors was to not include these labels (hence potentially creating missing "Present" labels in the mid- or high-levels of the ontology) instead of possibly generating incorrect labels. In the development set, these cases are provided as is since it is less critical. By contrast, because the cases in the evaluation set are more critical, they were partially reviewed and corrected. The potential impact of missing intermediate nodes is restricted to some instances of class labels with multiple-paths where the disambiguating parents could not be determined, and thus we expect this to have a minimal impact.

To finalize the label smearing process we filter out the out-of-vocabulary labels (labels beyond the 200 selected). In the majority of cases, these correspond to abstract or blacklisted classes. This is another reason why some clips have labels up to the ontology root while others only have a portion of the ancestors or even one single label. For example, *Whoosh, swoosh, swish* has no hierarchy as all class labels in its path were either removed previously due to specified constraints (*Arrow*) or removed in this last step (as classes above *Arrow* are abstract). This can be easily spotted in the provided ground truth CSV files.[24]

The number of labels before/after the propagation process can be seen in Table 3.4 (*unpropagated* and *smeared*, respectively). The outcome is a set of *smeared* labels consistently encompassing all relevant levels of the ontology. Note the considerable increase of labels, despite that we are ignoring parts of the ontology. This is the final ground truth provided for FSD50K.

3.3 FSD50K Description

FSD50K is an open dataset of human-labeled sound events containing 51,197 clips unequally distributed in 200 classes drawn from the AudioSet Ontology. This Section discusses its main characteristics, limitations and applications. The dataset is freely available from Zenodo.[24] Hereafter, we refer to *development* (composed of *training* and *validation*) and *evaluation* sets described in the previous Sections as *dev*, *train*, *val*, and *eval*.

3.3.1 Characteristics

FSD50K is composed mainly of sound events produced by physical sound sources and production mechanisms. Hence, the main focus is on the *casual listening* perspective of sound, as defined by Schaeffer (Schaeffer, 2016). It also includes some classes that can inherently encompass several more specific sources (e.g., *Train*), some classes that do not relate to a specific source but to the perception of sound (e.g., *Clatter*), and few abstract classes (e.g., *Human group actions*). The dataset has 200 sound classes (144 leaf nodes and 56 intermediate nodes) hierarchically organized with a subset of the AudioSet Ontology (Gemmeke et al., 2017). The vocabulary can be inspected in Figure 3.7. Note, however, that in some cases one leaf node in FSD50K (e.g., *Camera*) may be an intermediate node in AudioSet due to the fusion of low-occupancy classes (e.g., *Single-lens reflex camera*) with their parents. Following AudioSet Ontology's main families, the FSD50K vocabulary encompasses mainly *Human sounds*, *Sounds of things*, *Animal*, *Natural sounds* and *Music*. The vast majority of the content corresponds to sounds recorded from a sound field, while a small portion corresponds to sounds captured directly from electronic devices, typically in the context of musical instruments, e.g., some bass drums are generated with drum machines. The main characteristics of FSD50K in terms of number of clips, labels, duration and uploaders are listed in Table 3.4.

Table 3.4: Main statistics for FSD50K.

	Total	**dev**	**eval**
clips	51,197	40,966 (80%)	10,231 (20%)
labels (unpropagated)	62,657	45,607 (72.8%)	17,050 (27.2%)
avg labels/clip	1.22	1.11	1.67
labels (smeared)	152,867	114,271	38,596
clips w/ leaf label(s)	40,461	31,310	9151
duration	108.3h	80.4h (74.2%)	27.9h (25.8%)
avg duration/clip	7.6s	7.1s	9.8s
uploaders	7225	4936	2289

The audio clips are grouped into a dev split and an eval split such that they do not have clips from the same uploader. Eval is exhaustively labeled, that is, annotations are correct and complete for the considered vocabulary (except human error). In dev, a small amount of content is exhaustively labeled, but the vast majority is composed of labels that are correct but could be potentially incomplete (see Section 3.3.3 for label noise estimations). The number of labels is expressed in *unpropagated* and *smeared* forms. The number of unpropagated labels includes only the most specific labels per clip. It must be noted that

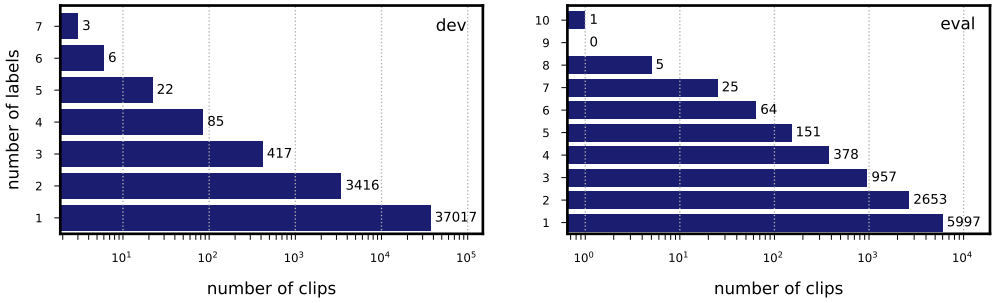

Figure 3.5: Label distributions in dev (left) and eval (right) sets. Clips in eval tend to have more labels (by dataset curation). Xaxis scale is logarithmic. Number of labels is reported in the *unpropagated* form. Note that visualization span differs among plots.

this way of counting labels ignores a few labels in cases where a sound event co-occurs with: *i)* events from low prior siblings that were merged with their parent; *ii)* events that do not fit semantically in any other sibling provided by the ontology, hence they are annotated with their parent. While these cases are not frequent, the true number of human-provided labels describing sound events would be slightly larger than the one reported here. Smeared labels refer to the labels after hierarchical propagation (Section 3.2.8.4). We use the unpropagated version to compute the average number of labels per clip. Note the increased number of labels per clip in eval due to the exhaustive labelling process, as can also be seen by comparing the label distributions in Figure 3.5.

In eval, all classes are present in both the singly-labeled data and the multi-labeled data forming it, except five classes that only appear in the multi-labeled data. In dev, all classes are present in both the singly-labeled data and the multi-labeled data forming it, except four classes that only appear in the singly-labeled data. A total of 31,310 clips are labeled with, at least, one leaf label in dev—the remaining 9656 clips are labeled only with intermediate node labels. This proportion changes significantly in eval, where the majority of clips have leaf labels (9151 out of 10,231)—this is because in the refinement task raters were instructed to provide the most specific labels possible. All provided ground truth labels are smeared, i.e., consistently propagated to their ancestors in the hierarchy. PP/PNP ratings are provided for the labels validated in the validation task. Out of the 108.3 hours of human-labeled audio, 31.5 are exhaustively labelled, most of them used for evaluation purposes (eval and val). The audio clips are of variable length ranging from 0.3 to 30s. Note the increased average duration of eval clips due to the allocation process (Section 3.2.6), which can also be noticed by comparing the clip length distributions in Figure 3.6. The ground truth labels are provided at the clip-level (i.e., weak

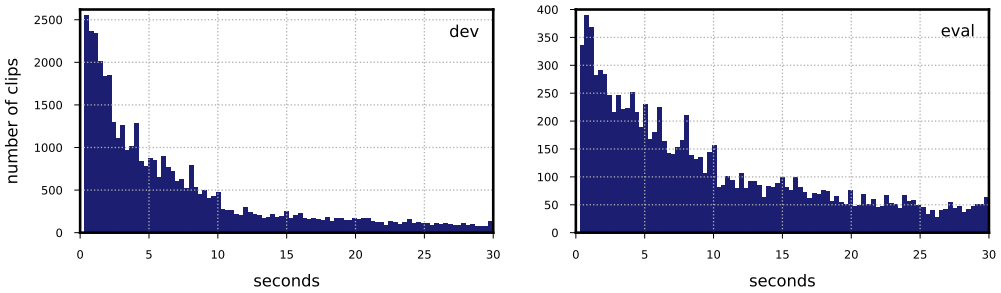

Figure 3.6: Audio clip length distributions in dev (left) and eval (right) sets. Clips in eval tend to last slightly longer (by dataset design). Bins correspond to 1/3 second. Note that visualization span differ among plots.

labels). The dataset is sourced from 7225 Freesound users and the content was uploaded from Freesound's launch in 2005 until early 2019.

The number of clips per leaf class varies, roughly, from 40 to 200 in eval, and from 50 to 500 in dev, with a few exceptions. The number of clips in the intermediate nodes grows much more depending on the hierarchy. Therefore, class imbalance comes from two sources: non-uniform class distribution and variable-length of clips. The dataset is licensed under CC-BY license—nonetheless, each clip has its own specific license (CC0, CC-BY, CC-BY-NC or CC Sampling+, where CC0 and CC-BY amount to 84.7% of the dataset). When original audio clips have more than one channel, they are downmixed to mono. All clips are provided as uncompressed PCM 16 bit 44.1 kHz mono audio files. Further details about data licenses, ground truth format, and additional provided metadata can be found in the FSD50K Zenodo page.[24] For comparison with existing datasets, Table 2.2 summarizes some aspects of a few most relevant SET datasets, including FSD50K.

The FSD50K dataset is a superset of the human-labeled portions of other datasets released during this thesis, except for a few audio clips that have been discarded during FSD50K's curation process. Specifically, these other datasets are FSDnoisy18k for learning with noisy labels (Section 5.2), as well as FSDKaggle2018 and FSDKaggle2019 developed for two DCASE Challenge Tasks in 2018 and 2019 (Appendix A).

3.3.2 Discussion

3.3.2.1 Variable Clip Length and Weak Labels

Labels in FSD50K are provided at the clip-level (i.e., weak labels). However, unlike other sound event datasets featuring audio clips of same or similar lengths (e.g., Salamon et al. (2014); Gemmeke et al. (2017); Salamon et al. (2017)), FSD50K is composed of variable-length clips in the range [0.3, 30] seconds (see Figure 3.6). This provides FSD50K with a particular feature. On the one hand, some clips contain sound events where the acoustic signal fills almost the entirety of the file, which can be understood as strong labels. To give a sense of this, 12,357 clips in the dev set are shorter than 4s and bear one single label validated only with PP ratings. Thus, we estimate that the dev set is composed mainly of weakly labeled data and a portion of strongly labeled data, in a rough proportion of 70%/30%. On the other hand, another small portion of the data presents a much weaker supervision—e.g., 9494 dev clips are longer than 10s (see Figure 3.6). The longer the clips, the lesser the certainty of where the labeled event is actually happening. This is referred to as *label density noise* in Shah et al. (2018), defined as a measure of the weakness of labels for a given weakly labeled clip. The impact and limitations of weak labels in SET are discussed in Turpault et al. (2020a). In the context of deep networks, clips' variable length implies that audio processing must be done either using fixed-length patches or utilizing variable-length inputs. Using fixed-length patches implies two issues: *i)* in training, the weak labels must be inherited by every patch (a practice called *false strong labeling* in Morfi & Stowell), which can generate false positives if the label is not active in a given patch; *ii)* in evaluation, patch-level scores must be aggregated into clip-level predictions to be compared against the weak labels. Utilizing variable-length inputs is free from the issues of the previous approach, but entails certain architectural constraints, such as using fully convolutional networks or appropriate pooling strategies.

3.3.2.2 Audio Quality

Given the heterogeneity of Freesound audio it is difficult to make strong objective claims about audio quality in FSD50K. Nonetheless, upon inspection of the clips' metadata, it can be seen that many Freesound users utilize (semi-) professional recording equipment (e.g., microphones or preamplifiers of brands such as *Neumann*, *Rode* or *Tascam*). Our experience after annotating the dataset is that the audio generally has a relatively high Signal-to-Noise Ratio (SNR) and dynamic range. To put this into context, we note that the notion of audio quality in sound recognition datasets has changed over time. In early DCASE Challenges, datasets recorded with professional equipment dominated, some of

them being recorded with one single microphone model (Mesaros et al., 2016, 2017a; Lafay et al., 2017; Mesaros et al., 2017b). Then, AudioSet became popular, in which a huge variety of devices are used for recording YouTube *videos* (where audio quality is not necessarily a priority), and often including lower SNR conditions. We extracted the global SNR for FSD50K and AudioSet (its eval set and balanced train set) using the ITU-T P.563 (Malfait et al., 2006). It must be noted that P.563 is designed for evaluating human speech, while the content of both datasets is much more diverse. Therefore, strong claims cannot be made based on this measurement. We use it as a common-referenced, rough indication of SNR given that it is not trivial to accurately compute SNR for the different types of audio under consideration. SNR values reported in Table 3.5 are mean and median or per-clip SNR values. The mean SNR for FSD50K is greater than that of AudioSet. For AudioSet, it can also be seen that the mean SNR is greater than the median, suggesting that the SNR distribution is positively skewed, i.e, lower SNR values are more frequent.

3.3.2.3 Real-world Audio

Many clips in Freesound are real-world recordings of sound events happening in the wild, e.g., a car passing by. However, it is not uncommon that some sound events are recorded under careful conditions in order to obtain clean and isolated high-quality sounds, as in a *foley sound* setting (e.g., the sound of tearing paper carefully located in front of a microphone). Further, a few clips in Freesound consist of sound events purposely generated with the sole objective of being recorded, e.g. a faked laughter. While these recordings are valuable for sound design, in some cases they could feature a lack of naturalness or acoustic mismatch with respect to sound events in the wild. This may question the suitability of a portion of the data for learning sound recognizers to be deployed in the wild, where more adverse generation and recording conditions can be encountered. To what extent this affects models' generalization to adverse scenarios is an open question. Mitigating this potential issue could be a research problem involving, for example, data augmentation (Salamon & Bello, 2017) or domain adaptation (Li et al., 2017a) techniques.

3.3.3 Limitations

3.3.3.1 Label Noise

Throughout this paper we have discussed the correctness/completeness of labels in the dataset. While we aimed at full label correctness and completeness, this is somewhat unrealistic as it would mean perfect accuracy of the nomination system that proposes candidate labels (Section 3.2.4), and of the

human-provided labels. In fact, as supervised learning research moves towards larger datasets, issues of label noise become inevitable. For instance, labeling error in AudioSet is estimated at above 50% for ≈18% of the classes.[14] Similarly, ImageNet data are often presumed to have correct labels, but it has been recently estimated that at least 100k images could be labeled incorrectly (Northcutt et al., 2021). In SER, label sets in not-small datasets are inherently noisy due to reasons like sub-optimality of automatic methods used in the creation, or the difficulty of annotating audio—especially without visual cues, with large vocabularies, and because the annotation process is, sometimes, inherently subjective and ambiguous. Consequently, recent works have shown the efficacy of label noise treatment in large datasets such as AudioSet (Kumar & Ithapu, 2019; Fonseca et al., 2020b) and mid-size datasets (Fonseca et al., 2019b,a; Iqbal et al., 2020).

Despite our efforts to mitigate label noise in FSD50K, there are still a few label noise problems. The main problem is the existence of *missing* "Present" labels (false negatives). These are labels that would be included in an ideal exhaustive annotation but which are missing from the current set. Our recent work identifies this as a pathology in AudioSet as well, and proposes a method to tackle it (Fonseca et al., 2020b). This problem affects the dev set more due to the annotation process based on validation of previously nominated labels—if sound events are not nominated by the system, they lack labels (Section 3.2.5). This may happen with sound events that tend to be less represented by the Freesound user-provided tags, such as human or bird sounds when they are not the most relevant events in a clip. Because the eval set received exhaustive annotation, this problem is minimized there. To a much lesser extent, two additional sources of missing labels exist. First, the impossibility of propagating labels in the hierarchy when multiple ambiguous paths are encountered—again, this affects more the dev set, as explained in Section 3.2.8.4. Second, missing labels can occur as a result of annotating with a finite vocabulary—there may be additional acoustic content *out-of-vocabulary*. Apart from missing labels, the other label noise problem is *incorrect* "Present" labels (a false positive, and potentially a false negative if the true class is *in-vocabulary*). This would be the result of human annotation errors. Because we adopted mechanisms to bootstrap human annotation quality (Sections 3.2.5 and 3.2.7), we expect incorrect labels to be rare (see below). Both missing and incorrect labels would be class-conditional as some classes are clearly more ambiguous than others. When labelling errors occur, the non-existent true labels can be either in-vocabulary or out-of-vocabulary. Further details about label noise characterization can be found in Chapter 5.

We can use the refinement task processing to quantify the label noise at the output of the validation task. A total of 11,847 audio clips were processed with the refinement task, the majority of which ended up in the eval set. The

processing undergone by these clips in this task in order to approach a complete sound event transcript is summarized next. A total of 6030 (50.9%) received at least one additional label, indicating that there was some unlabeled material. For these 6030 clips, a total of 10,473 labels were generated. One must be careful when extrapolating these numbers to the dev set. As explained earlier, clips selected for dev tend to have, on average, less number of sources per clip. The incoming 11,847 audio clips featured a total of 13,681 labels, out of which 773 (5.7%) were rejected by the annotators. This means that 94.3% of the incoming labels were verified as correct. This gives a sense of the level of label correctness obtained with the validation task (and therefore, the dev set). For the eval set, we have not quantified the amount of correctness and completeness due to lack of resources. However, the amount of correctness is expected to be not very different from the 94.3% estimated at the output of the validation task. We expect this because the hired raters are more qualified at the refinement task than during the previous validation task, as they have gained more annotation experience and a deeper knowledge of the ontology.

Labelling errors in FSD50K can be reported via its companion site.[26] In this way, future dataset releases can include fixes reported in a collaborative way.

3.3.3.2 Data Imbalance

While some classes are abundant, others are much less represented due to the data scarcity in Freesound and/or low performance of the nomination system. Another source of imbalance is the variable length of clips—some classes tend to contain shorter/longer clips depending on the sound events and the preferences of Freesound users when recording them. Finally, the hierarchy of the ontology favours data imbalance between classes at different levels.

3.3.3.3 Data Bias in Development Set

Because we prioritized the allocation of small uploaders in the eval set to increase its diversity (Section 3.2.6), the development portion of a few classes is dominated by a few large uploaders. Under the assumption that this signifies similar training examples in certain cases, this could create a data bias, which could be learned by models (Lei et al., 2011). This happens mainly in a few musical instruments, e.g., *Trumpet*, due to the fact that Freesound users tend to upload many clips for these classes. Further analysis would be needed to determine if and how much this potential bias causes lack of generalization for these classes.

3.3.3.4 Lack of Specificity in the Vocabulary

Some leaf nodes in the ontology were merged to their parents due to data scarcity. For instance, leaf nodes such as *Blender*, *Chopping (food)*, and *Toothbrush* had to be merged with their parent *Domestic sounds, home sounds*. This motivated us to keep the latter class as a valid class despite that it is blocked in the AudioSet Ontology (Gemmeke et al., 2017). A natural extension of FSD50K is to grow these merged leaf nodes by adding more data.

3.3.4 Applications

FSD50K allows evaluation of approaches for a variety of sound recognition tasks. The most evident is multilabel sound event classification with large vocabulary (Fonseca et al., 2019c, 2021a). In this context, the proposed dataset supports several approaches such as learning sound event representations directly from waveforms (Cakir et al., 2016; Park & Yoo, 2020); analysis of label noise mitigation methods leveraging the non-exhaustive labeling of the dev set (Fonseca et al., 2019b,a, 2020b); multimodal approaches using audio and text information (e.g., using the provided Freesound tags, title, and textual description for the clips) (Elizalde et al., 2019; Favory et al., 2020a); evaluation of hierarchical classification via ontology-aware learning frameworks (Jati et al., 2019; Cramer et al., 2020; Shrivaslava et al., 2020); or approaches specifically combining strong and weak labels (Kumar & Raj, 2017). By leveraging the common vocabulary between FSD50K and AudioSet, we hope that a number of tasks become possible, such as experimenting with domain adaptation techniques (Gharib et al., 2018), or cross-dataset evaluation (Bogdanov et al., 2016) under different acoustic conditions. Other tasks include search result clustering in large vocabulary datasets (Favory et al., 2020b) or universal sound separation (Kavalerov et al., 2019a). Given its large vocabulary and diversity, FSD50K is well suited to learn unsupervised general-purpose audio representations (Niizumi et al., 2021; Tsouvalas et al., 2021), and the audio data can serve as stimuli for listening experiments in cognitive sciences.

In addition, the collection of FSD50K has already accomplished several high-impact milestones. A subset of the curated data has been used for a number of smaller datasets for sound event classification (Fonseca et al., 2018b, 2019b,c; Abeßer, 2021), source separation (Wisdom et al., 2021), and as a soundbank to generate synthetic data for sound event detection (Turpault et al., 2019; Yadav & Foster, 2021). Likewise, from the beginning of its creation, subsets of FSD50K have enabled several sound recognition Challenges, specifically: *i)* DCASE 2018 Task 2 "General-purpose tagging of Freesound audio with AudioSet labels" (Fonseca et al., 2018b), *ii)* DCASE 2019 Task 2 "Audio tagging with noisy labels and minimal supervision" (Fonseca et al., 2019c), *iii)* DCASE

2019 Task 4 "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis" (Turpault et al., 2019), *iv)* DCASE 2020 Task 4 "Improving sound event detection in domestic environments using sound separation" (Turpault et al., 2020b), and *v)* DCASE 2021 Task 4 "Sound Event Detection and Separation in Domestic Environments" (Turpault et al., 2019; Wisdom et al., 2021). These multiple contributions showcase the value of this effort.

3.3.5 FSD50K and AudioSet

Because FSD50K and AudioSet are based on the same ontology and thus are partially compatible, we discuss the main similarities and differences between both. Table 3.5 summarizes some of them.

Table 3.5: Comparison of some properties of FSD50K and AudioSet.

	FSD50K	**AudioSet**
classes	200	527
content	waveform	features
dev clips	40,966	≈2M
eval clips	10,231	20,383
clip length	0.3-30s	≈10s
dev labeling	CpI	CpI
eval labeling	exhaustive	CpI
source	Freesound audio	YouTube video
train/val split	✓	-
P.563 SNR (mean, median) [dB]	(26, 25)	(14, 10)

Both datasets use the AudioSet Ontology for organization, however FSD50K uses a smaller subset. All classes in FSD50K are represented in AudioSet, except *Crash cymbal* as well as four classes that are blacklisted in AudioSet but not in FSD50K (*Human group actions*, *Human voice*, *Respiratory sounds*, and *Domestic sounds, home sounds*). The official AudioSet release consists of audio features pre-computed at a time resolution of 960ms, released under CC-BY-4.0 license. FSD50K provides audio waveforms under several CC licenses as decided by Freesound users. In terms of stability, FSD50K is downloadable as several zip files from its Zenodo page.[24] AudioSet features can be downloaded as a tar.gz file from the AudioSet website.[4] The original YouTube video soundtracks, however, are gradually disappearing as they are subject to deletions and other issues, and their usage may be affected by copyright policies. As seen in Table 3.5, AudioSet's dev set is significantly larger than FSD50K's whereas AudioSet's eval set is roughly twice that of FSD50K. Since AudioSet

has a vocabulary 2.6 times larger, this means that in some classes there is more evaluation content in FSD50K. Clips in AudioSet last ≈10s, whereas in FSD50K their length varies from 0.3 to 30s. Therefore, label weakness is more homogeneous in AudioSet, whereas it varies significantly in FSD50K, yielding quasi-strong labels as clips get shorter, and much weaker labels in the longest clips.

In terms of labeling, FSD50K provides event predominance annotations ("Present and predominant" & "Present but not predominant", Section 3.2.5) while AudioSet only provides presence annotations ("Present"). While it is not easy to objectively compare label quality in both datasets, we speculate that the labeling of both dev sets could be generally regarded as *Correct but Potentially Incomplete* (CpI), i.e., both dev sets would be affected by a certain amount of missing labels. However, it seems reasonable to assume that, in the FSD50K portion of rather short sounds with PP annotations (see Section 3.3.2), the amount of missing labels is minimal. The eval set of FSD50K was exhaustively annotated; therefore, absence of labels means absence of sound events (except human error). By contrast, the eval annotations in AudioSet would be in general CpI, similar to those of the AudioSet train set. Unlike AudioSet, FSD50K consistently provides all relevant labels in a hierarchical path, except in a few specific cases of ambiguous ancestors. As additional resources, we provide additional metadata (e.g., Freesound tags and class-wise annotation FAQs) and allow flagging labeling errors.[26]

Finally, despite both datasets being highly heterogeneous, we make the following conjectures. Freesound clips are typically recorded with the goal of capturing audio, which is not necessarily the case in YouTube videos. Additionally, given the AudioSet size, its audio clips are presumably recorded with a higher diversity of devices. This would provide AudioSet with a higher diversity of audio qualities, often including more real-world and lower SNR conditions than Freesound audio (see Table 3.5 for a rough SNR estimation described in Section 3.3.2). Thus, a certain acoustic mismatch between both datasets may be expected. In our view, both datasets suppose complementary resources for sound event research.

3.4 Experiments

In this Section, we conduct a set of multi-label SET experiments to give a sense of the performance that can be achieved with FSD50K using a baseline pipeline (Section 3.4.1), and to learn about the main challenges to consider when splitting Freesound audio for SER tasks (Section 3.4.2). For reproducibility, implementation details of evaluation metrics, learning pipeline, and networks can be inspected in the open-source code.[25]

3.4.1 Baseline Systems

Next, we benchmark several commonly used deep networks on the proposed FSD50K.

3.4.1.1 Learning Pipeline

Incoming audio is downsampled to 22.050 kHz and transformed to 96-band, log-mel spectrogram as input representation. To deal with the variable-length clips, we use Time-Frequency (T-F) patches of 1s (equivalent to 101 frames of 30ms with 10ms hop)—thus the input to all models is of shape TxF=101x96. Clips shorter than 1s are concatenated until such length, while longer clips are sliced in several patches with 50% overlap inheriting the clip-level label (a.k.a. false strong labeling (Morfi & Stowell)). We adopt the train/val split designed in Section 3.2.8.3. We implement a learning pipeline in TensorFlow (Abadi et al., 2015). Models are trained using Adam optimizer (Kingma & Ba, 2015) to minimize binary cross-entropy loss, with initial learning rate depending on the network (see Table 3.7), which is halved whenever the validation PR-AUC plateaus for 5 epochs (no tolerance). Models are trained up to 100 epochs, earlystopping the training whenever the validation PR-AUC is not improved in 10 epochs. We use a batch size of 64 and shuffle training examples between epochs. Once the training is over, the model checkpoint with best validation PR-AUC is selected to predict scores and evaluate performance on the eval set. We optimize PR-AUC (instead of other metrics based on ROC curves) because PR curves can be more informative of performance when dealing with imbalanced datasets (Davis & Goadrich, 2006). Likewise, we use PR-AUC (instead of mAP) for simplicity as it is a built-in metric in TensorFlow. For inference, we pass each (eval or val) T-F patch through the model to compute output scores, which are then averaged per-class across all patches in a clip to obtain clip-level predictions, as in Gemmeke et al. (2017). We note this aggregation must be done also for validation—preliminary experiments validating at patch-level using inherited clip-level labels revealed misleading results. Extensive hyper-parameter tuning (beyond learning rate) is not conducted.

3.4.1.2 Network Architectures

Current trends in SER encompass mainly CNNs (Hershey et al., 2017; Kong et al., 2020a) and CRNNs (Cakır et al., 2017; Pérez-López et al., 2019). We run experiments with the following networks, all of them ending with a fully connected layer of 200 units (the vocabulary size) with sigmoid activation to support multi-label classification. The main hyperparameters for the CRNN and

Visual Geometry Group (VGG)-like architectures are set via non-exhaustive preliminary experiments.

CRNN. This is one of the most used architectures for SED (Cakır et al., 2017), and to a lesser extent for SET (Ebbers & Häb-Umbach, 2019). Our model, inspired by Cakır et al. (2017), has three convolutional layers of 128 filters with a receptive field of (5,5), each of them followed by Batch Normalization (Ioffe & Szegedy, 2015), Rectified Linear Unit (ReLU) activation and max-pooling. The max-pooling sizes are $(t, f) = (2, 5)$, $(2, 4)$ and $(2, 2)$—since we are not interested in detecting events' timestamps, we pool also in the time dimension which reduces dimensionality without harming performance in our experiments. To model events' temporal structure in the incoming feature maps, the convolutional stack is followed by a bidirectional GRU layer of 64 units, returning the last output of the output sequence.

VGG-like. VGG-based architectures have been widely used for both SET (Dorfer & Widmer, 2018) and SED (Kim & Pardo, 2019). We use a model inspired by the original architecture (Simonyan & Zisserman, 2015) from computer vision, but reduced to a much smaller size. In particular, this model has three convolutional layers of 32 filters, two convolutional layers of 64 filters, and one convolutional layer of 128 filters. All convolutional layers have a receptive field of (3,3) and are followed by Batch Normalization and ReLU activation. Between each group of convolutional layers with same number of filters, max-pooling of size (2,2) is applied. Output feature maps are summarized by concatenating global max pooling and global average pooling per channel. Summarizing the learnt audio representation via combination of these two poolings provided a small mAP boost with respect to using either of them individually. Then, the outcome is passed through a fully connected layer of 256 units.

Finally, we also experiment with two architectures taken off-the-shelf from the computer vision literature. While the two previous networks received some tuning in their design, the next ones are the original architectures without any tuning whatsoever—only the input/output shapes to match our task.

ResNet-18. ResNets (He et al., 2016a) have been sucessfully used for SET (Jansen et al., 2018; Kong et al., 2020a; Fonseca et al., 2020b).

DenseNet-121. DenseNets are reported to outperform ResNets for image recognition (Huang et al., 2017), and have been recently used for SET (Fonseca et al., 2019a; Iqbal et al., 2020).

3.4.1.3 Results

Table 3.6 lists the results for the considered architectures, and Table 3.7 lists the learning rates used (after tuning on val set) as well as the number of weights for each architecture.

Table 3.6: Evaluation performance for the architectures considered.

Model	mAP	d'	lwlrap
CRNN	0.417 ± 0.003	2.068 ± 0.015	**0.519** ± 0.002
VGG-like	**0.434** ± 0.002	**2.167** ± 0.011	0.514 ± 0.003
ResNet-18	0.373 ± 0.001	1.883 ± 0.020	0.465 ± 0.001
DenseNet-121	0.425 ± 0.002	2.112 ± 0.032	0.505 ± 0.004

Table 3.7: Learning rates used (after tuning on val set) and number of weights for the architectures considered.

Model	lr	Weights
CRNN	5e-4	0.96M
VGG-like	3e-4	0.27M
ResNet-18	1e-5	11.3M
DenseNet-121	5e-5	12.5M

Each network is trained from scratch three times with different random initialisation and different orderings in the training data. We report average and standard deviation of the evaluation performance across the three trials. The following results and discussion are based on the particular train/val/eval split used. Interestingly, the best overall model across all metrics is VGG-like, despite being less modern and more lightweight than the other architectures. This result accords with similar recent findings in music genre recognition (Won et al., 2020b). The building blocks of VGG-like are very similar to those of the *CNN14* network in Kong et al. (2020a), which rivals state-of-the-art results in AudioSet classification (Kong et al., 2020a). However, CNN14 is much deeper and heavier (81M weights). The VGG-like model is closely followed by DenseNet-121, which has many more weights, and then by the CRNN, which shows the best *lωlrap*. CRNN architectures are also used in some top SED systems, e.g., in recent DCASE Challenge Task 4 editions (Turpault et al., 2019, 2020b). ResNet-18 is found to be the worst performing model. Curiously, we also observe that the optimal learning rate tend to be rather low for this architecture (Table 3.7). We also tried ResNet-34 in preliminary experiments, obtaining similar results (at the expense of many more weights). Our results

contrast with the successful results of Ford et al. (2019); Kong et al. (2020a), which achieved state-of-the-art performance for AudioSet classification using ResNet architectures. Factors possibly influencing this different behaviour include the different amount of training data (much larger in AudioSet) or the data itself. Overall, the results of Table 3.6 suggest that, at our scale of data, smaller models with basic tuning and audio-informed design choices can outperform much larger off-the-shelf computer vision architectures; however, DenseNet-121 with no tuning provides good performance.

It is also interesting to compare the results of DenseNet-121 and ResNet-18, which have an order of magnitude more weights than the other models (12.5M and 11.3M, respectively). It can be seen in Table 3.6 that the performance achieved by DenseNet-121 outperforms that of ResNet-18 by a large margin. If we take into account the good AudioSet classification performance of ResNet architectures reported in Ford et al. (2019); Kong et al. (2020a), the contrast of results suggests that the ResNet-18 architecture requires more data or regularization than DenseNet-121 in order to show superior performance. Possible ways to evaluate this hypothesis include analyzing the impact of increasing training data or using data augmentation to train these networks.

Figure 3.7 shows the per-class AP (averaged across three trials) for all classes in FSD50K, using the best-performing VGG-like model (dark blue), and the CRNN model (light blue). Leaf nodes with top recognition include *Applause*, *Burping, eructation*, *Purr*, and *Computer keyboard*, with AP over 0.75. The worst performance is shown in *Boat, Water vehicle*, *Cowbell*, *Speech synthesizer*, *Tap* and *Tick*. After inspection of the latter classes, we conjecture this is due to aspects such as high intra-class variation, confound with other similar classes, ambiguity in the class definitions, or very short length of sound events—all of them being relevant challenges in SER. Finally, it can be seen that most per-class APs by the CRNN are slightly lower than those of the VGG-like model—as expected since VGG-like has a higher overall mAP. However, there are a few exceptions in which the CRNN performs better, such as in different types of speech (either spoken, sung, screamed, yelled or whispered). This is interesting as CRNNs were originally proposed for speech recognition (Sainath et al., 2015) before being adapted for SER (Cakır et al., 2017). Other exceptions include some human sounds and animal vocalizations of marked temporal behaviour, e.g., types of laughter (*Chuckle, chortle* or *Giggle*), *Gasp*, or *Crying, sobbing*; *Bark*, *Meow* or *Chicken, rooster*. This highlights the different behaviour, for some classes, of a model including a recurrent layer with respect to another relying only on convolutional layers.

3.4.2 Impact of Train/Validation Separation

In Section 3.2.8.3 we discussed some factors to consider when splitting Freesound audio data for machine learning, and we designed a validation set emphasizing the issue of data contamination. Here, we experimentally analyze the impact of contamination in this setting. To this end, we pick one architecture from Section 3.4.1 (the CRNN) and we train and evaluate it utilizing three different train/validation splits in order to show their differences. Specifically, let us consider three candidate validation sets obtained with different approaches:

1. **val_random** is computed via *random* sampling. We run 3000 trials of a train/validation separation and we select the validation set with minimum Jensen-Shannon (JS) divergence[35] with respect to the development set.
2. **val_is** is computed via *iterative stratification* (Sechidis et al., 2011). We run 3000 trials of a train/validation separation and we select the validation set with the minimum number of shared uploaders between training and validation.[36]
3. **val** is the validation set proposed in Section 3.2.8.3.

In all cases, the validation set is initialized with most of the data that was transferred from the first evaluation set prototype to the development set for balancing purposes (Section 3.2.8.2). Since this content is exhaustively labeled, it is well suited for evaluation purposes. The main characteristics of the three validation sets are listed in Table 3.8.

Table 3.8: Main statistics for the considered validation sets.

Validation Set	Clips	Duration	JS	Shared Uploaders	PR-AUC Drop
val_random	4697	9.7h	1.8×10^{-2}	930	0.15
val_is	4543	9.3h	6.8×10^{-3}	857	0.14
val (proposed)	4170	9.9h	2.1×10^{-2}	641	≈ 0

The sets *val_random* and *val_is* amount to $\approx 15\%$ of the development data associated with leaf nodes (see Table 3.4); *val* is slightly lower (13.3%) due to

[35] The JS divergence is based on the Kullback-Leibler divergence but it is symmetric. We use it as a distance metric to measure similarity between the development and validation distributions, similarly as in Cramer et al. (2020).

[36] Minimizing the JS divergence is not needed here as stratification is already the objective of this method, hence all separations have a fairly consistent JS divergence.

allocating less validation data for the most abundant classes as well as some approximations (Section 3.2.8.3). All validation sets have a similar duration. In terms of stratification, the split done through iterative stratification, *val_is*, yields more similar class distributions than the other two, which are on par. The main differences lie in the uploaders "shared" between train and validation, both in number and in their nature. In particular, *val_random* and *val_is* suffer from within-class (WC) and between-class (BC) contamination as no measure was taken to prevent them. By contrast, *val* was designed to minimize WC contamination while being relatively flexible with BC contamination. Therefore, not only is the number of shared uploaders less in the proposed *val*, but also the contamination is limited mostly to BC.

To compare the candidate splits, we train the CRNN of the previous Section using the three of them (in this case, with a learning rate of 1e-4 and no learning rate scheduling). Figure 3.8 illustrates the learning curves (PR-AUC for train, validation, and evaluation) using each of the splits.

We display 60 training epochs allowing validation and evaluation performance to roughly stabilise. From Figure 3.8 and Table 3.8 several observations can be made. In the left and middle plots of Figure 3.8 (corresponding to *val_random* and *val_is*), validation performance is substantially better than evaluation performance. In these cases, the classifier is trained and validated on clips from the same uploader *and* the same class (i.e., WC contamination). We call this the "uploader effect" (following the analogy of the "album effect" (Mandel & Ellis, 2005) or "artist effect" (Flexer & Schnitzer, 2009)). In Table 3.8, it can be seen that the number of uploaders shared between train and validation is positively correlated with the validation-evaluation PR-AUC drop. In the cases of *val_random* and *val_is* we observe substantial performance drops, whereas with *val* the performance drop is negligible. Results from Figure 3.8 and Table 3.8 suggest that, when contamination is considered and minimized, validation performance is a good proxy of evaluation performance—otherwise, it can be overly optimistic. Consequently, if the model is tuned using the validation set it may occur that, depending on the type and amount of contamination, the tuning reflects model's ability to partially overfit train data rather than to generalise to unseen data. In addition, our results indicate that the distinction between WC and BC contamination seems reasonable in the context of Freesound audio organized with a large vocabulary, confirming our initial hypothesis that WC is the most harmful type while BC has lesser impact (Section 3.2.8.3).

Lastly, we observe a slightly higher train performance and slightly lower validation and eval performances when using *val* (right plot of Figure 3.8), which content comes mostly from a variety of small uploders. Under the assumption that not all training examples are equally informative (which is the basis for disciplines like instance selection (Liu & Motoda, 2002)), this may occur

because the content transferred to *val* includes some highly informative examples. Yet, we propose this train/validation split for systems' benchmarking because we deem it more methodologically correct than the others given that data contamination is minimized. In summary, carefully splitting Freesound audio is important as it can have a non-negligible impact on learning and performance. Therefore, for reproducibility and fair comparability of results, system benchmarking should be done explicitly specifying the validation split that was used.

3.5 Summary and Conclusion

In this Chapter, we introduced *FSD50K*, a dataset containing 51,197 Freesound clips totalling over 100h of audio manually labeled using 200 classes drawn from the AudioSet Ontology. The audio clips are CC-licensed, thereby making the dataset freely distributable (including audio waveforms). We proposed a methodology for creating datasets of sound events based on *i)* human validation of previously nominated candidate labels, followed by *ii)* a refinement process where labels are reviewed and completed to approach a complete transcription of the audio material for the vocabulary under consideration. In order to gather human annotations, we employed a mixture of crowd-sourcing strategies and recruited trained annotators. During this process, we experienced how human labeling of everyday sounds is a laborious and complex task, especially when using a large vocabulary encompassing multiple audio domains. Special emphasis was put on the careful curation of the evaluation set content and labels, so that it can serve as a reliable benchmark. To our knowledge, a large-vocabulary, stable and exhaustively labeled evaluation set of this size is unprecedented. This evaluation resource can be valuable for benchmarking sound event classification systems, regardless of the methodology used for their training. Overall, in this Chapter we showed that it is important to acquire solid knowledge of the specifics of the source data—in our case, Freesound audio and metadata, and the AudioSet Ontology—and identify data challenges and limitations, so that the dataset creation process can be adapted to these particularities, and pitfalls in the creation can be avoided.

Through FSD50K classification experiments, we showed that smaller models with basic tuning and audio-informed design choices can outperform larger off-the-shelf computer vision architectures. Further, motivated by data constraints encountered when splitting FSD50K's development set into training and validation sets, we considered the issue of contamination, and distinguished two types: within-class and between-class contamination. Within-class contamination occurs when content from the same uploader *and* belonging to the same class is placed at both train and validation sets. In contrast, between-class

contamination occurs when content from the same uploader *but not* from the same class is placed at both train and validation sets. Our results show that, in presence of within-class contamination, validation performance is overly optimistic, substantially better than evaluation performance. This is attributed to the fact that the classifier is trained and validated on clips from the same uploader *and* the same class. In contrast, when this type of contamination is minimized, validation performance is a good proxy of evaluation performance, even in presence of some between-class contamination. These results confirm our hypothesis that within-class is the most critical type to be avoided, while between-class has a lesser impact. Thus, within-class data contamination must be considered when splitting Freesound audio for machine learning tasks as it can have a considerable effect on the evaluation of sound event classifiers.

FSD50K is an open and stable dataset aimed at complementing AudioSet in order to foster reproducible large-vocabulary SER research. In the future, dataset extensions could be carried out. In fact, FSD50K has already been used and adapted for other research tasks than sound event classification, as discussed in Section 3.3.4. More data could be added via semi-automatic methods by leveraging models trained on FSD50K to scale up efficiently. Likewise, the vocabulary could be extended by growing the merged leaf nodes in FSD50K. We hope FSD50K and its creation process to be useful as an example model for open audio datasets.

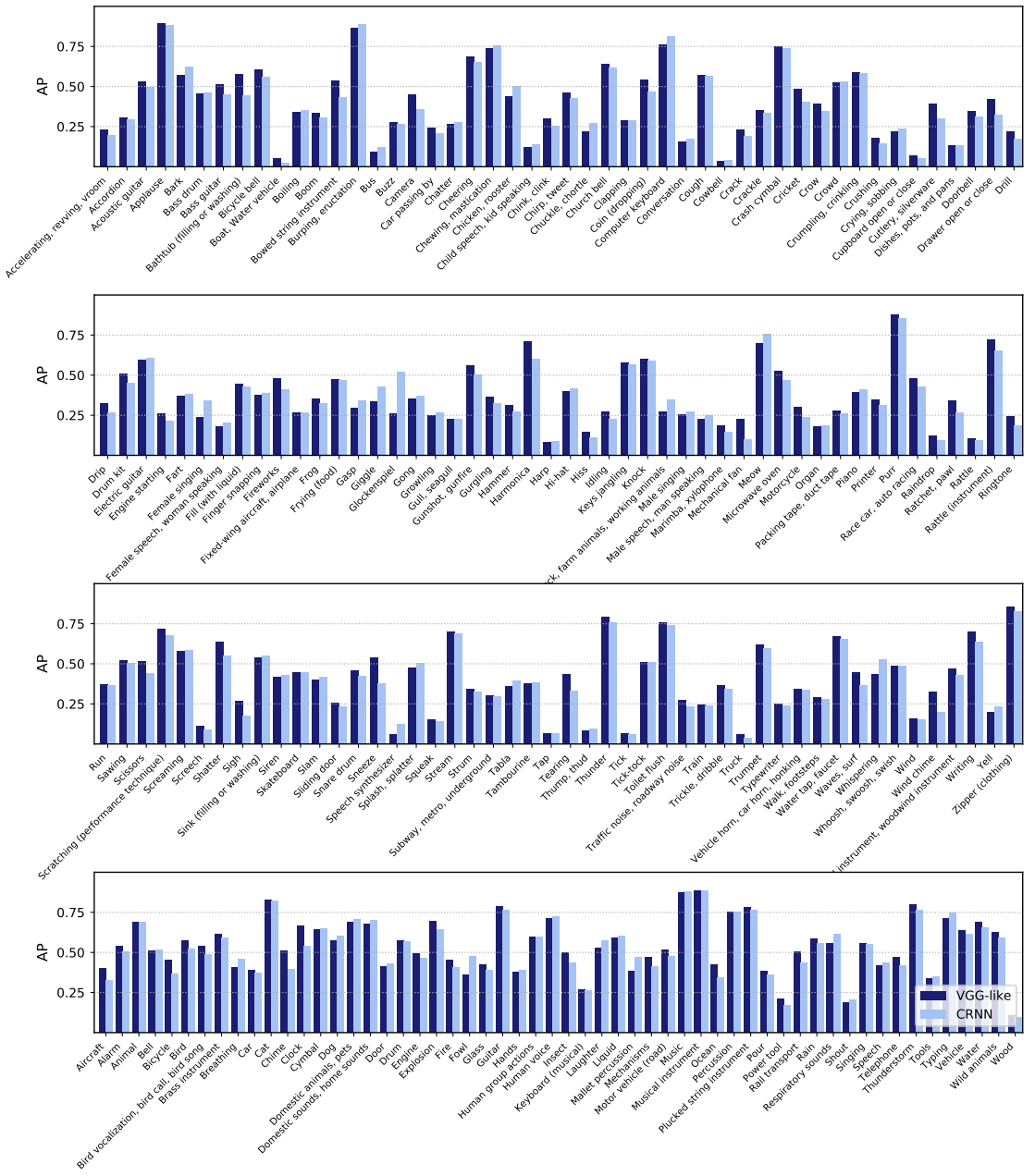

Figure 3.7: Per-class average precision for all classes in FSD50K, using the best-performing VGG-like model (dark blue) and the CRNN model (light blue). Top 3 rows show the 144 leaf nodes and bottom row comprises the 56 intermediate nodes.

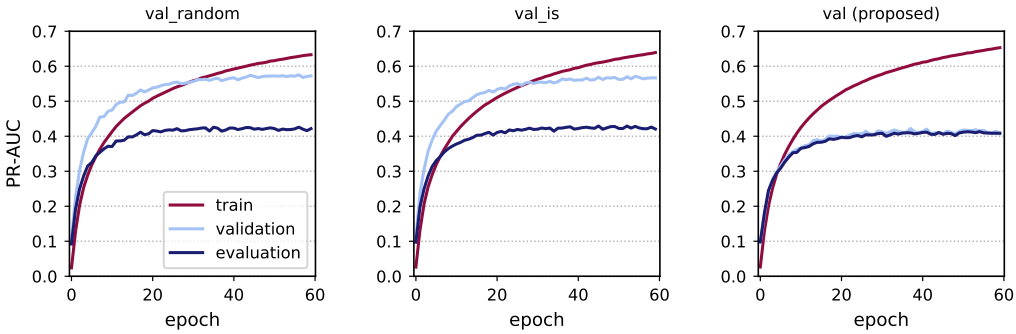

Figure 3.8: Learning curves (PR-AUC for train, validation, and evaluation) for the CRNN model using the three train/validation splits specified in Table 3.8 (*val_random* (left), *val_is* (middle), and the proposed *val* (right)). Validation performance is substantially better than evaluation performance when using *val_random* and *val_is*, in which the classifier is trained and validated on clips from the same uploader *and* the same class (WC contamination). When this type of contamination is minimized (*val*), validation performance is a good proxy of evaluation performance.

CHAPTER 4

Improving Sound Event Classification by Increasing Shift Invariance in Convolutional Neural Networks

4.1 Introduction

At the end of the previous Chapter, we evaluated a series of *off-the-shelf* and commonly-adopted CNNs on FSD50K in order to give a sense of the classification performance possible with this dataset. Results in Section 3.4.1 show that a VGG-like architecture achieves the most promising results among the CNNs considered. In this Chapter, we focus on improving the generalization capabilities of this architecture by increasing its invariance against shifts in input spectrograms, and also by training it with mixup augmentation. The adoption of these approaches, together with a larger capacity VGG network and a more suitable summarization pooling, allows us to obtain a new state-of-the-art performance on the FSD50K classification benchmark.

As mentioned in Section 2.5.2, recent studies have put into question the commonly-assumed shift invariance property of CNNs, showing that small shifts in the input can affect the output predictions substantially (Azulay & Weiss, 2018; Engstrom et al., 2018; Zhang, 2019). These works empirically show the brittleness of CNNs against minor input perturbations, and their only-partial invariance to shifts. These works also argue that one of the causes of the lack of shift invariance is a wrongly executed subsampling operation that ignores the classic sampling theorem, yielding *aliasing* problems. To address

this issue, the predominant trend consists of adding anti-aliasing measures to the CNN architectures, adopting low-pass filter based solutions, usually for image recognition tasks (Zhang, 2019; Vasconcelos et al., 2020). A different line of work is to design architectural changes to explicitly enforce invariance in the network, for example via adaptive sampling of incoming feature maps (Chaman & Dokmanic, 2021). To our knowledge, this kind of techniques aimed at fostering shift invariance in CNNs have not been evaluated for sound event classification.

In this Chapter, we ask whether lack of shift invariance is a problem in sound event classification, and whether there are benefits in addressing it. To this end, we apply several mechanisms aimed at increasing shift invariance in the subsampling operations of CNNs, and evaluate them on the large-vocabulary sound event classification task of FSD50K. Specifically, we adopt mechanisms from the two trends mentioned above, namely, low-pass filters (non-trainable as proposed in Zhang (2019), as well as a trainable version proposed by us), and adaptive polyphase sampling (Chaman & Dokmanic, 2021). We insert these architectural changes into the max-pooling layers of VGG variants (Simonyan & Zisserman, 2015), and we evaluate their effect on FSD50K using models of small and large capacity, and in presence of a strong regularizer such as *mixup* (Zhang et al., 2018). We show that these simple changes consistently improve sound event classification in all cases considered. We also demonstrate empirically that the proposed pooling methods increase shift invariance in the network, making it more robust against time/frequency shifts in input spectrograms. This is achieved without adding any (or adding only few) trainable parameters, which makes the proposed mechanisms an appealing alternative to conventional pooling layers. The outcome is a new state-of-the-art mAP of 0.541 on the FSD50K classification benchmark when not using external training data, outperforming the models evaluated in the previous Chapter by a large margin.

The rest of this Chapter is organized as follows. In Section 4.2 we describe the proposed methods to reinforce shift invariance in CNNs. Section 4.3 outlines the experimental methodology followed to evaluate the methods, and also details the base architecture where the pooling mechanisms are inserted. The results of our evaluation are reported in Section 4.4, including a characterization of the increase of shift invariance achieved (Section 4.4.4). This Chapter ends with a summary and final remarks in Section 4.5.

4.2 Methods

Our focus is on evaluating mechanisms to improve shift invariance applied to the subsampling operations within max-pooling layers in CNNs. A max-pooling layer with squared size k and stride s can be understood as the cascade of two operations, as illustrated in the top diagram of Figure 4.1: a densely-evaluated (i.e., with unit stride) max-pooling operation with size k, followed by a subsampling operation with stride s greater than unity.

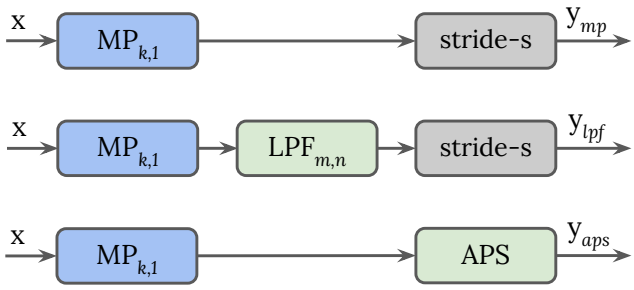

Figure 4.1: Max-pooling layer and proposed methods to improve shift invariance. *Top*: A max-pooling layer can be decomposed into a densely-evaluated max-pooling operation with size k, followed by a subsampling operation with stride s. *Middle*: Inclusion of a low-pass filter before subsampling. *Bottom*: Adaptive Polyphase Sampling (APS) can be used instead of naive subsampling.

4.2.1 Low-pass Filtering Before Subsampling

We focus on the effect of low-pass filtering feature maps before subsampling in the context of a max-pooling layer, inspired by Zhang (2019). The subsampling operation may incur in aliasing problems as the incoming signal (the feature map) is not band-limited. The classic signal-processing fix is to add a low-pass filter before subsampling (Oppenheim et al., 2001). A simple manner to realize this filter is through a 2D kernel, $LPF_{m,n}$, of size m x n, such that the max-pooling layer for an incoming feature map x becomes

$$y_{lpf} = \text{Subsample}_s(LPF_{m,n}(\text{MaxPool}_{k,1}(x))), \tag{4.1}$$

where $\text{MaxPool}_{k,1}$ is a max-pooling operation across areas of size k x k and unit stride, $LPF_{m,n}$ applies a low-pass filter of size m x n, and Subsample_s denotes naive subsampling with a stride s, as illustrated in the middle diagram of Figure 4.1. A graphical example of these concepts is illustrated in Figure 4.2 for a case of max-pooling of size 2x2, a $LPF_{5,5}$, and a subsampling operation

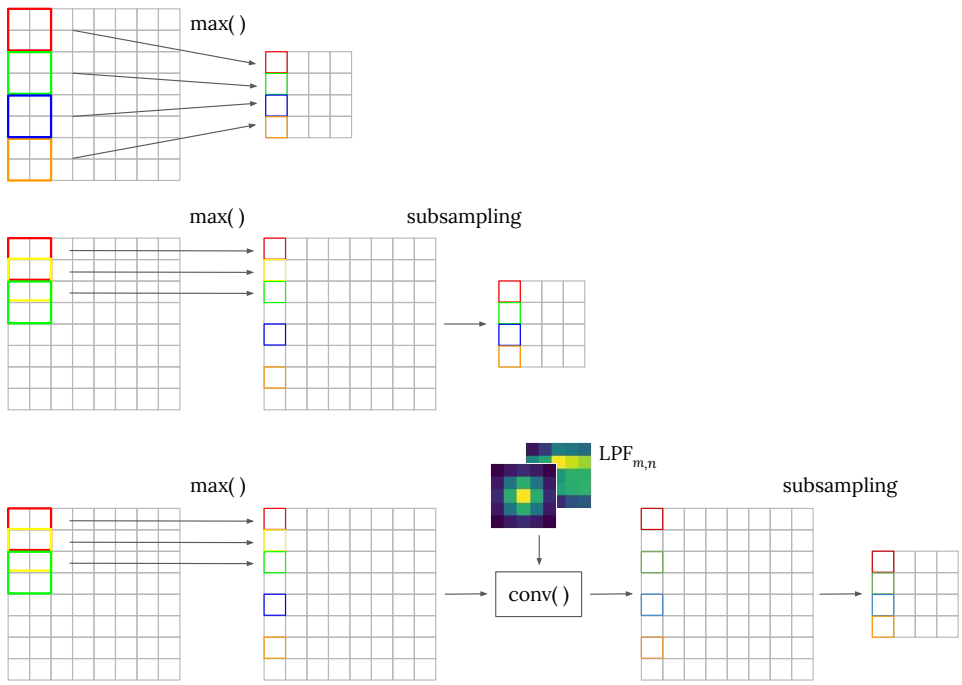

Figure 4.2: Introducing low-pass filtering before subsampling within a max-pooling operation. *Top*: A typical max-pooling operation of size 2x2. *Middle*: The max-pooling operation in the top can be decomposed into a unit-stride max-pooling operation of size 2x2, followed by a subsampling operation with stride $s = 2$. *Bottom*: A low-pass filter $LPF_{m,n}$ with $m = n = 5$ is applied before subsampling using a convolution operation. As a result, the energy of bins in the output feature map changes, as depicted by the different colour scheme.

with stride $s = 2$. Figure 4.2 also depicts a non-trainable *LPF* (front) and a trainable *LPF* (back).

Inserting low-pass filtering before subsampling can have different benefits when applied within CNNs. First, in case the feature maps present energy variations of too high frequency for the subsampling operation to be carried errorless, $LPF_{m,n}$ may help to mitigate aliasing.[37] This can reduce the amount of corrupted information flowing through the network. Second, the signal processing

[37]By high-frequency energy variations in the feature map we refer to rapid spectro-temporal modulations or sharp patterns in the 2D signal formed by a feature map. This should not be confused with the frequency components of the input audio signal. The high-frequency energy variations are not necessarily constrained to a specific region of the feature map. For example, a sequence of human clapping sounds forms a time-frequency representation with a series of transients. In its corresponding feature map, the energy variations given by such a sequence of transients can generate high frequencies, even at the lowest end of the spectrum.

literature demonstrates how preventing aliasing can favour shift invariance in a given process (Oppenheim et al., 2001). One way to see it is that $LPF_{m,n}$ spreads possible sharp patterns across neighbouring feature map bins. Intuitively, when subsampling differently-shifted versions of a spectrogram, the subsampled feature maps are likely to be more structurally similar if they have been previously low-pass filtered. This could provide the network with improved generalization to this kind of small shifts, potentially increasing classification performance. Third, $LPF_{m,n}$ is essentially blurring or smoothing out the incoming feature map, which could be understood as a form of regularization. For example, L2 regularization is a common way to penalize outlier weights with large absolute values, driving them close towards zero (Cortes et al., 2012). It could be argued that the proposed $LPF_{m,n}$ inflicts a similar effect on the feature map bins, smoothing out the most drastic energy variations—in other words, attenuating the high frequency components in the 2D signal formed by the feature map. In Section 4.4 we discuss through experiments which of these hypotheses seem more plausible.

To implement $LPF_{m,n}$, one of the most basic characteristics of common 2D image-oriented low-pass filter kernels is: non-negative weights that add up to unity (Distante et al., 2020). This can be realized in several ways.

Non-trainable Low-pass Filters. These are commonly defined as binomial filters, which are in turn discrete approximations of Gaussian filters. To generate 1D binomial filters, a simple manner is to repeatedly convolve the base averaging mask [1,1] with itself, in order to get filter masks such as [1,2,1], [1,3,3,1], or [1,4,6,4,1], for one, two and three convolutions, respectively. Then, a 2D squared binomial mask can be obtained simply by convolving a 1D binomial filter with its transpose (Distante et al., 2020). An example of this type of low-pass filter with 5x5 size can be seen at the bottom diagram of Figure 4.2 ($LPF_{m,n}$ at the front). In Section 4.4 we denote this type of filters as *BlurPool* for consistency with Zhang (2019) as the non-trainable low-pass filters that we use in our experiments are largely inspired by this work.

Trainable Low-pass Filters. These can be defined by randomly initializing a kernel with dimensions m x n, and learning their weights through back propagation. In order to imprint the low-pass nature to the filter, its weights can be passed through a softmax function to ensure non-negativity and normalization. An example of this type of low-pass filter with 5x5 size can be seen at the bottom diagram of Figure 4.2 ($LPF_{m,n}$ at the back). In Section 4.4 we denote this type of filters as Trainable Low-pass Filter (TLPF). A potential alternative is to create auxiliary loss functions to encourage the filter weights to adopt a low-pass behaviour through loss penalization. Trainable low-pass filters have also been used recently within a learnable audio frontend (Zeghidour et al., 2021).

In this Chapter, we compare non-trainable low-pass filters (BlurPool) and trainable low-pass filters (TLPF) constrained via softmax function, for simplicity. As illustrated in Figure 4.2, a given low-pass filter can be applied over the incoming feature map via a convolution operation. This convolution operation can also incorporate the required subsequent subsampling stride s. More specifically, in our case the term $\text{Subsample}_s(LPF_{m,n}(\cdot))$ in Equation 4.1 is implemented via a depthwise separable convolution using either a trainable or non-trainable $LPF_{m,n}$, and a stride s.

4.2.2 Adaptive Polyphase Sampling

Adaptive Polyphase Sampling (APS) is a downsampling mechanism that directly addresses the lack of shift invariance caused by subsampling operations (Chaman & Dokmanic, 2021). The underlying principle of APS is based on a simple observation: the result of subsampling a T-F patch and subsampling its shifted-by-one-bin version can be different when bins are sampled at the same fixed positions (see top diagram in Figure 4.3). This happens because the energies captured by the same grid over two shifted patches are likely to be different. However, when subsampling a feature map, multiple candidate grids could actually be used instead of always using the same grid (as typically done). Intuitively, a time/frequency shift applied over an input patch could be seen conceptually as translating its energy bins from one grid to another. One way to be robust to these shifts is to select the subsampling grid adaptively based on some criterion, such that the grid follows the shift at the input (see bottom diagram in Figure 4.3).

More formally, given an input feature map x, and considering a subsampling operation[38] with stride $s = 2$, there are four possible grids that can be used for subsampling, depending on which bin from the four options in each 2x2 area is passed to the output. The bottom diagram in Figure 4.3 shows two grids of the four possible in this case. Subsampling with each grid will yield one of the four possible candidate subsampled feature maps, termed *polyphase components* (Chaman & Dokmanic, 2021), which can be denoted as $\{y_{ij}\}_{i,j=0}^{1}$. Analogously, if we consider a shifted-by-one-bin version of the input feature map, $\tilde{x}$, its polyphase components are given by $\{\tilde{y}_{ij}\}_{i,j=0}^{1}$.

The conventional course of action consists of always choosing the same subsampling grid and consequently returning the same polyphase component (e.g., y_{00} by picking the top left bin in each 2x2 area). However, as mentioned, depending on the input patch, this will likely cause different downsampled outputs when the patch is simply shifted by one bin ($y_{00} \neq \tilde{y}_{00}$). It can be

[38] This subsampling operation would follow a densely-evaluated max pooling operation in order to form a typical max-pooling layer, see Figure 4.1.

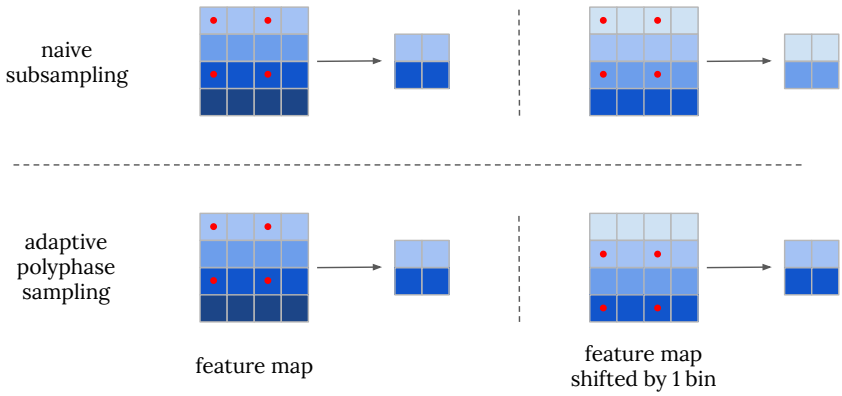

Figure 4.3: Underlying principle of adaptive polyphase sampling with stride $s = 2$. The Figure illustrates a feature map (left side) and a version of itself shifted downwards by one bin (right side). Different levels of blue represent the energy in every bin. Red dots represent the sampling locations in a given grid. *Top*: When the same fixed sampling grid is always used (naive subsampling), a small shift in the input can cause a change in the output feature map (top right case). *Bottom*: When the grid is selected adaptively based on input's energy (adaptive polyphase sampling), the output feature map remains the same (i.e., the subsampling process becomes more robust to input shifts).

demonstrated that the set $\{\tilde{y}_{ij}\}$ is a re-ordered version of $\{y_{ij}\}$ (which may be potentially shifted, but carrying identical energy values) (Chaman & Dokmanic, 2021). Therefore, by adaptively choosing a polyphase component in a permutation invariant way, a very similar subsampled output, $y_{i_{aps}j_{aps}}$, would be obtained regardless of sampling from x or $\tilde{x}$. The adaptive selection can be done by maximizing a given criterion, for example, maximizing some norm l_p, as given by

$$i_{aps}, j_{aps} = \underset{i,j}{\arg\max} \left\{ \|y_{ij}\|_p \right\}_{i,j=0}^{1}, \tag{4.2}$$

where $p \in \{1,2\}$. In this way, by substituting the naive subsampling in a max-pooling layer by APS (as illustrated in the bottom diagram of Figure 4.1), robustness to incoming time/frequency shifts is increased.

The benefit from APS comes from the generalization to shifts embedded in the network's architecture, in a fashion conceptually similar to what is done with $LPF_{m,n}$ (Section 4.2.1). APS, in contrast, provides no explicit measures against potential aliasing problems.

4.3 Experimental Setup

4.3.1 Evaluation and Training Details

We evaluate the proposed methods on the large-vocabulary sound event classification task posed by the FSD50K dataset introduced in Chapter 3. We follow the evaluation procedure described in Section 3.4 (with minor deviations). We outline it next for convenience. Incoming clips are transformed to log-mel spectrograms using a 30ms Hann window with 10ms hop, and 96 bands. To deal with the variable-length clips, we use T-F patches of 1s, equivalent to 101 frames, yielding patches of $T \times F = 101 \times 96$ that feed the networks. Clips shorter than 1s are replicated while longer clips are trimmed in several patches with 50% overlap inheriting the clip-level label. We train, validate and evaluate using the proposed *train* set, *val* set and *eval* set. Models are trained using Adam optimizer (Kingma & Ba, 2015) to minimize binary cross-entropy loss. Learning rate is 3e-5, halved whenever the validation metric plateaus for 10 epochs. Models are trained up to 150 epochs, earlystopping the training whenever the validation metric is not improved in 20 epochs. We use a batch size of 128 and shuffle training examples between epochs. Once the training is over, the model checkpoint with the best validation metric is selected to predict scores and evaluate performance on the eval set. For inference, we compute output scores for every (eval or val) T-F patch, then average per-class scores across all patches in a clip to obtain clip-level predictions. Our main evaluation metric is balanced mAP, that is, AP computed on a per-class basis, then averaged with equal weight across all classes to yield the overall performance, following Gemmeke et al. (2017); Fonseca et al. (2020a). In addition, we report d' and *lwlrap* for the top systems.

4.3.2 Baseline Model

As a base network, we use a VGG-like architecture (Simonyan & Zisserman, 2015) for the following reasons: *i)* this type of architecture has been widely used for SET (Hershey et al., 2017; Dorfer & Widmer, 2018; Kong et al., 2020a; Ebrahimpour et al., 2020) and is the most competitive baseline for FSD50K when compared to others of higher complexity (see Section 3.4.1), which also accords with recent music tagging evaluations (Won et al., 2020b); *ii)* due to its limited size compared to the other baselines of Section 3.4.1, it allows faster experimentation; *iii)* this architecture conveniently features several max-pooling layers that allow the study of the proposed pooling mechanisms (Simonyan & Zisserman, 2015). Specifically, the network that we use for the majority of our experiments is similar to a VGG type A (Simonyan & Zisserman, 2015). The network, denoted as *VGG41*, consists of 4 convolutional blocks, with each block

comprising two convolutions with a receptive field of (3,3), and each convolution followed by Batch Normalization (Ioffe & Szegedy, 2015) and ReLU activation. Between the blocks, max-pooling layers of size 2x2 (and same stride) are placed by default—they will be substituted by the proposed pooling mechanisms. A densely-evaluated max pooling operation (of size 3x3 and unit stride) will sometimes be inserted between the convolutions within each block—we will refer to it as Intra-block Pooling (IBP). This provides partial translation invariance but not dimensionality reduction, allowing the same (max) element to be transferred to the output in adjacent spatial locations. This tweak has been applied in various non-audio applications (Goodfellow et al., 2016), and to a lesser extent also in SET tasks (Ebrahimpour et al., 2020). Finally, in order to summarize the final feature map information before the output classifier, we use a global pooling in which we first aggregate information along the spectral dimension via averaging for every time step, then max-pool the outcome in the time dimension. We found out that aggregating first spectral and then temporal information in this manner is the most beneficial for our task among other combinations. VGG41 has 1.2M weights, which allows for relatively fast experimentation. The baseline and topline configurations using VGG41 are also evaluated using *VGG42* (of 4.9M weights), where we double the width of the network with respect to VGG41 (i.e., using twice the number of filters in every convolutional layer).

4.3.3 *mixup*

We evaluate the baseline and top performing methods proposed in Section 4.2 with or without *mixup* augmentation (Zhang et al., 2018), which was introduced in Section 2.3.4. This is done in order to further improve the generalization capabilities of our system, and also to analyze the methods' behavior in presence of a strong regularizer. Following (Zhang et al., 2018), we sample λ from a beta distribution $\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$ (see Section 2.3.4 for details). The hyperparameter α, which controls the interpolation strength, is set to 1.25 after tuning on the val set.

We choose mixup because the concept of mixing sounds is an audio-informed operation (unlike other data augmentation methods), and it has been proven useful for SET in several previous works (Fonseca et al., 2019a; Kong et al., 2020a; Gong et al., 2021b). In our view, mixup can be interpreted from two different perspectives, as mentioned in Section 2.3.4. First, it is a regularizer to mitigate overfitting, which can be important at our scale of data, especially for some classes that present less than a hundred training clips. Second, mixup is a mechanism that allows to cover during training a diversity of examples that may be encountered in evaluation, hence improving generalization. In particular, upon the creation of FSD50K, audio clips with multiple sound sources

were prioritized to some extent for the eval set, whereas the dev set presents a higher proportion of single-source clips (see Section 3.2.6). It can therefore be argued that a kind of domain shift exists between both sets, which is being partially compensated through mixup. Hence, this type of augmentation is specially well aligned with the recognition task of FSD50K.

It must be noted that the original formulation of mixup and BC learning considers multi-class problems, where the training examples feature one single label per example, hence using one-hot encoded vectors to represent the labels (Zhang et al., 2018; Tokozume et al., 2018). In contrast, our current problem deals with multi-label data and thus the binary vectors that we are combining are potentially multi-hot (see Section 2.3.1). To better understand the key ingredients of mixup, in Section 4.4.2 we provide an ablation study comparing different variants of the original algorithm.

4.4 Experiments

We evaluate the methods proposed in Section 4.2 on the SET task posed by FSD50K, using VGG41 (Section 4.4.1) and also using mixup and VGG42 (Section 4.4.3). In Section 4.4.4 we demonstrate that the methods are increasing the network's robustness to input shifts. In Section 4.4.5 we include a per-class analysis to show how the proposed methods affect the different classes in FSD50K. Sections 4.4.6 and 4.4.7 provide discussion and comparison with previous work on FSD50K. For all the classification results (Tables 4.1, 4.2, 4.3 and 4.4), we report average and standard deviation of the evaluation performance across three trials of each experiment. In each trial, the network is trained from scratch with different random initialisation and different ordering in the training data. Code for the experiments is available.[39]

4.4.1 Evaluation using a Small Model

Table 4.1 shows the results of inserting the pooling mechanisms individually into VGG41 (left section) as well as in some pairwise combinations (right section). By looking at the left section, it can be seen that all the evaluated methods outperform the baseline system. That is, inserting each of the methods alone into a standard VGG-like architecture improves recognition performance. The mAP boosts range from 0.003 in the worst case (APS l_2) to 0.023 in the best case (APS l_1). If we focus on the low-pass filter based solutions, we observe that this classical signal processing technique is beneficial for CNN-based sound event classification. While it may seem that blurring the

[39] https://github.com/edufonseca/shift_sec

Table 4.1: mAP obtained by inserting different pooling mechanisms into the VGG41 baseline. TLPF = Trainable Low-pass Filter, APS = Adaptive Polyphase Sampling, IBP = Intra-block Pooling.

Method	**mAP**	**Method**	**mAP**
VGG41 (baseline)	0.457 ± 0.003	+ BlurPool 5x5 + IBP	0.479 ± 0.003
+ BlurPool 3x3	0.475 ± 0.002	+ TLPF 5x5 + IBP	0.481 ± 0.002
+ BlurPool 5x5	0.476 ± 0.003	+ TLPF 5x5 + APS l_1	**0.484** ± 0.002
+ TLPF 3x3	0.476 ± 0.003	+ APS l_1 + IBP	0.478 ± 0.001
+ TLPF 5x5	0.479 ± 0.003		
+ TLPF 6x6	0.477 ± 0.001		
+ APS l_1	**0.480** ± 0.001		
+ APS l_2	0.460 ± 0.002		
+ IBP	0.472 ± 0.002		

feature maps can smooth out relevant detailed information (thus leading to performance degradation) results indicate that it is indeed helpful. The choice of trainable vs. non-trainable low-pass filters does not seem critical, yet the trainable version TLPF seems to produce slightly higher mAP values. The different sizes of these filters allow to find a trade-off between high-frequency smoothing and loss of information in the incoming feature maps (the larger the size, the stronger the smoothing effect). Results seem to indicate that larger smoothing areas (5x5 vs. 3x3) are beneficial. By looking at results with APS, we observe that l_1 outperforms l_2 as norm criterion. We also did preliminary experiments with other metrics such as l_∞, l_0 and variance, but we found l_1 to be the best choice overall. Interestingly, a naive tweak like IPB also shows some impact, although more modest than that of the other methods. The two top methods when applied individually are APS l_1 and TLPF 5x5, showing on par performance.

We set out to combine some of the methods in pairs in order to see if they are complementary (right section). Combining low-pass filtering (which operates before subsampling between convolutional blocks) and IBP (which operates between convolutions within every block) seems to provide a small but consistent boost, for both BlurPool and TLPF. When joining the top performing methods, specifically, low-pass filtering the incoming feature maps with TLPF 5x5, followed by subsampling them with APS, we observe a small performance boost. A possible explanation for their complementarity could lie in TLPF addressing aliasing issues while APS is agnostic to it. Finally, joining APS and IBP does not yield further boosts.

Table 4.2: mAP obtained by exploring different low-pass filter shapes in TLPF over the baseline of Table 4.1. Filters can be 2D squared (*m* x *n*), or 1D in frequency (1 x *n*) or time (*m* x 1). IBP is always applied.

TLPF *m* **x** *n*	**mAP**
TLPF 3x3	0.478 ± 0.002
TLPF 4x4	0.480 ± 0.004
TLPF 5x5	**0.481** ± 0.002
TLPF 6x6	0.480 ± 0.001
TLPF 1x4	0.475 ± 0.005
TLPF 1x5	0.480 ± 0.001
TLPF 1x6	0.480 ± 0.002
TLPF 4x1	0.469 ± 0.004
TLPF 5x1	0.470 ± 0.004
TLPF 6x1	0.472 ± 0.002

Table 4.2 shows the results of exploring different low-pass filter shapes in TLPF. In previous work, low-pass filters are usually adopted for computer vision tasks, hence they are of squared size, e.g., Zhang (2019); Vasconcelos et al. (2020). Here, we seek to find out if there is one axis of the audio spectrogram feature maps (time or frequency) along which low-pass filtering is more beneficial. To this end, we run experiments using 1D trainable low-pass filters applied only along the frequency axis (filters of size 1 x *n*) or the temporal axis (filters of size *m* x 1). At the top section of Table 4.2, we first report the results by progressively increasing the area of squared filters. A sweet spot in the size 5x5 can be observed. Then, we report results by low-pass filtering only along the frequency axis. Interestingly, we find out that much of the performance obtained with squared filters is already achieved by smoothing out the spectral variations alone. In contrast, when we apply the low-pass filters only along the time axis the performance is noticeably worse (bottom section of Table 4.2).

4.4.2 Ablation Study of *mixup* Strategies

In this Section, we explore several variants of mixup to better understand its behaviour. Table 4.3 shows the results of exploring different mixup variants using the VGG41 baseline of Table 4.1. The top section in Table 4.3 lists the results for the default mixup augmentation as described in Section 2.3.4.1, applying the mix operation over log-mel spectrograms and linear-mel spectrograms. The processing pipeline is identical in both cases except that, in the linear case, we apply the mix over the linear mel energies, and then apply log compression to the mixed outcome. The motivation to run this experiment is

Table 4.3: mAP obtained by exploring different mixup variants over the baseline of Table 4.1.

Approach	**mAP**
mixup over log-mel spectrograms (*mixup*)	0.497 ± 0.003
mixup over linear-mel spectrograms	0.485 ± 0.004
sum log-mel spectrograms & OR(labels)	0.492 ± 0.003
sum linear-mel spectrograms & OR(labels)	0.490 ± 0.002
mixup & 1-epoch warm-up	**0.499** ± 0.001
mixup & 4-epoch warm-up	0.499 ± 0.003

that, arguably, mixing items in the log domain does not seem the most correct course of action. However, in practice it can be seen that the best results are achieved by mixing in the log domain.

In the middle section, we explore an alternative to mixup, in which the mixture is not carried out via convex combination of input spectrograms and labels, as given by Equation 2.1. Here, instead, the mixture is simply done by summing up the spectrograms, and applying the *OR* operation to the original target vectors. The spectrograms can be summed up either in the log domain or in the linear domain, in an analogous manner to the processing done for the results of the top section of Table 4.3. Due to the OR operation applied to original target vectors, the resulting target vector is still a binary vector $\boldsymbol{y} \in \{0,1\}^C$ for a vocabulary of C classes. Note that this is different in mixup, where the convex combination of binary vectors leads to final target vectors with real values for the active classes, tasking the network to predict less confidently. Consequently, in this alternative there is no longer need for a λ parameter. The experimental results obtained are inconsistent. When operating directly in the log domain, the default processing of performing mixtures via convex combinations seems to be helpful (see first and third rows of Table 4.3). In contrast, when mixing in the linear domain, better results are achieved by the proposed alternative that does not employ convex combinations (see second and fourth rows of Table 4.3).

Finally, in the bottom section of Table 4.3, we explore the inclusion of an initial warm-up training period in which mixup is disabled (1 and 4 epoch), before activating it. This is motivated by a similar strategy adopted in Zhang et al. (2018), where it is reported to speed up convergence in speech recognition experiments. It can be observed that this warm-up period seems to provide a marginal lift with respect to the default mixup reported at the top row of Table 4.3. However, such lift is within the standard deviation ranges obtained by running three trials of each experiment. For simplicity, in the following we

shall adopt the standard mixup operation applied over log-mel spectrograms, corresponding to the top row of Table 4.3. We shall refer to it simply as *mixup*.

4.4.3 Evaluation using Regularization and a Larger Model

Next, we select the best setups of the two pooling mechanisms considered on VGG41 (one based on low-pass filtering and another based on APS), as well as their combination. Table 4.4 shows the results using these setups, now adding mixup augmentation and also doubling the width of the network, which means multiplying its number of weights approximately by four. As mentioned, the adopted mixup implementation is that of the top row of Table 4.3. Table 4.4 also shows results when substituting TLPF by the analogous BlurPool in our best system (bottom row).

Table 4.4: mAP obtained by using top performing pooling mechanisms in presence of mixup and with the larger capacity VGG42. Values in parenthesis are absolute improvements over the corresponding baseline. TLPF = Trainable Low-pass Filter, APS = Adaptive Polyphase Sampling, IBP = Intra-block Pooling.

Method	VGG41	VGG41 + mixup	VGG42 + mixup
Baseline	0.457 ± 0.003	0.497 ± 0.003	0.523 ± 0.002
+ APS l_1	0.480 ± 0.001 (+0.023)	0.513 ± 0.003 (+0.016)	0.538 ± 0.004 (+0.015)
+ TLPF 5x5 + IBP	0.481 ± 0.002 (+0.024)	0.511 ± 0.003 (+0.014)	0.539 ± 0.002 (+0.016)
+ TLPF 5x5 + APS l_1	**0.484** ± 0.002 (+0.027)	**0.515** ± 0.003 (+0.018)	**0.541** ± 0.002 (+0.018)
+ BlurPool 5x5 + APS l_1	0.478 ± 0.002 (+0.021)	0.512 ± 0.003 (+0.015)	0.538 ± 0.002 (+0.015)

The left column of Table 4.4 lists the best results from Table 4.1, showing mAP boosts from 0.023 to 0.027 with respect to the baseline. When we train VGG41 using mixup (center column), substantial performance improvements are observed, demonstrating the good alignment of this operation with SET in general—which accords with Kong et al. (2020a); Gong et al. (2021b)—and with the FSD50K classification task in particular, as discussed in Section 4.3.3. All methods perform in the same ballpark, showing boosts of up to 0.018 with respect to the baseline, where the combination of TLPF and APS yields top mAP.

Our motivation to combine the proposed methods with mixup is twofold. In addition to improving generalization, we want to analyze the methods' behaviour in presence of a strong regularizer. If the proposed methods act solely as a general form of regularization, we would expect them to provide limited boosts when combined with mixup. We do observe a certain improvement decrease when combined, but the boosts are still solid, both when using VGG41 and VGG42 (center and right columns). These results suggest that the proposed methods are addressing problems beyond lack of regularization, presumably reinforcing robustness to time/frequency shifts at the input. In Section 4.4.4, we demonstrate that this is indeed the case by systematically applying time/frequency shifts to a set of input spectrogram patches, and analyzing the network's robustness against these shifts with and without the proposed pooling mechanisms. Finally, when inserting these pooling methods into VGG42 in presence of mixup (right column), we see that they are also beneficial within a larger-capacity model where performance is more competitive (in our case, increasing the capacity from 1.2 to 4.9M weights). In particular, combining TLPF and APS yields the top mAP again, showing a boost of 0.018 over the baseline.

Finally, the last row of Table 4.4 lists results when substituting TLPF 5x5 by the analogous BlurPool in our best system. Similar to the results reported in Table 4.1 when comparing trainable vs. non-trainable low-pass filters, the observed performance differences are not large, yet the trainable version TLPF seems to yield slightly higher mAP values. Figure 4.4 shows an example of a non-trainable fixed low-pass filter used in BlurPool 5x5, as well as a selection of four learned low-pass filters used in TLPF 5x5. Specifically, the filters in Figure 4.4 are extracted from the VGG41 baseline after insertion of the corresponding filtering approach (BlurPool 5x5 or TLPF 5x5). In BlurPool, all low-pass filters in the network are the same by construction, hence we only show one of them (left side). In TLPF, multiple filters are learned that feature different patterns (right side).

Figure 4.4: Example of low-pass filters extracted from the VGG41 baseline after insertion of BlurPool 5x5 (left) or TLPF 5x5 (right). In BlurPool all low-pass filters in the network are fixed by construction. In TLPF multiple different filters are learned.

4.4.4 Characterizing the Increase of Shift Invariance

In previous experiments we have seen that classification performance is improved when we adopt the proposed pooling mechanisms, presumably due to the increase of shift invariance. Here, we demonstrate empirically that these pooling mechanisms are indeed addressing this problem. To this end, we apply shifts to a set of input spectrogram patches and analyze the network's robustness against these shifts with and without the proposed pooling mechanisms. The set of data for this evaluation consists of 1000 audio clips from FSD50K's eval set.[40] They are selected at random after applying the following constraints for a more controlled experimental scenario: *i)* we choose clips with one single label (i.e., presumably containing one single sound event); *ii)* we choose clips with a minimum length of 2s, so that we can always select a patch in the time span [0.5, 1.5] s, allowing the discard of potential preceding silence that sometimes occurs.

The shifts applied to the input 1s T-F patches of $T \times F = 101 \times 96$ obey one of the two following protocols. The first protocol (denoted as *time-*n_f) is simply a time shift of the patch by n_f frames, with $n_f \in \{1,3,5\}$. Each unity of n_f corresponds to 10ms (the hop size when framing the input audio signal). The second protocol (denoted as *freq-*n_b) consists of shifting the input patch by n_b mel bands upwards in frequency, with $n_b \in \{1,3,5\}$. The n_b original highest bands are discarded, and the n_b lowest bands in the new patch are filled with white noise centered at the mean value of the original lowest band. By doing this, we analyze the effect not only of frequency shifts but also of small artificial perturbations in the input.

For every input patch, we *i)* apply one shift according to one of the above protocols; *ii)* compute network predictions for both original and shifted patches, and *iii)* measure the predictions' sensitivity to the shift using two metrics, namely, *classification consistency* and *mean absolute change*. Classification consistency refers to the percentage of cases in which the network predicts the same top class for both original and shifted patches (Zhang, 2019; Chaman & Dokmanic, 2021). Mean Absolute Change (MAC) is a metric that measures the absolute change of the probability predicted for the top class after the shift, averaged across the 1000 examples (Azulay & Weiss, 2018). The motivation to use MAC is to rule out the possibility that variations in classification consistency are originated by tiny differences between the top class and the second most likely class predicted.

Table 4.5 shows the result of this evaluation for the time and frequency protocols. In Table 4.5, *baseline* corresponds to the VGG41 baseline of Table 4.1,

[40] For future reproducible evaluations, this list of 1000 files is released at https://github.com/edufonseca/shift_sec.

whereas *proposed* corresponds to the same model after incorporating TLPF 5x5 and APS l_1.

Table 4.5: Classification consistency (in %, higher is better) and mean absolute change (MAC) (lower is better) when applying time and frequency shift protocols over input patches. Models evaluated are the *baseline* of Table 4.1 and the same model after inserting TLPF 5x5 and APS l_1 (*proposed*). The proposed model exhibits higher robustness to shifts.

Protocol	**Consistency (%)**		**MAC**	
	baseline	proposed	baseline	proposed
time-1	82.0	92.5	0.078	0.030
time-3	75.2	87.2	0.106	0.048
time-5	74.0	83.2	0.110	0.062
freq-1	63.5	79.8	0.175	0.078
freq-3	53.3	67.6	0.234	0.145
freq-5	49.0	60.0	0.263	0.199

In the results of the time protocol (top section), we see that by applying a time shift of only 1 frame (10ms), the baseline network changes its top prediction 18% of the time. Note that this is a minor modification in the time framing of the input audio signal, which can be regarded as imperceptible to a human in most cases. The proposed network (including the mechanisms to increase shift invariance) shows higher classification consistency than the baseline for all the cases considered. As the time shifts increase, the network becomes less consistent (according to our definition). These findings are confirmed by the proposed network consistently showing smaller MAC values, i.e., the output probability for the top class is more robust to the time shifts. By looking at the results of the frequency protocol (bottom section of Table 4.5), we observe a similar trend, with the proposed network showing increased robustness to frequency shifts and small perturbations (i.e., larger classification consistency percentages and smaller MAC values than the baseline model). Interestingly, we see that the classification consistency values in the frequency protocol are overall lower than those observed for the time protocol (and vice versa for the MAC values). For example, the minimal shift applied in time yields consistency values of 82.0% and 92.5% for baseline and proposed models, respectively. In contrast, the minimal shift applied in frequency leads to analogous values of 63.5% and 79.8%. This can be due to several reasons. First, the network considered may be more sensitive to frequency shifts than to time shifts. This could be linked to results in Section 4.4.1, where low-pass filtering along the frequency axis is shown to be more effective than along time. Second, it could happen that the frequency shifts affect the semantics of the input examples to

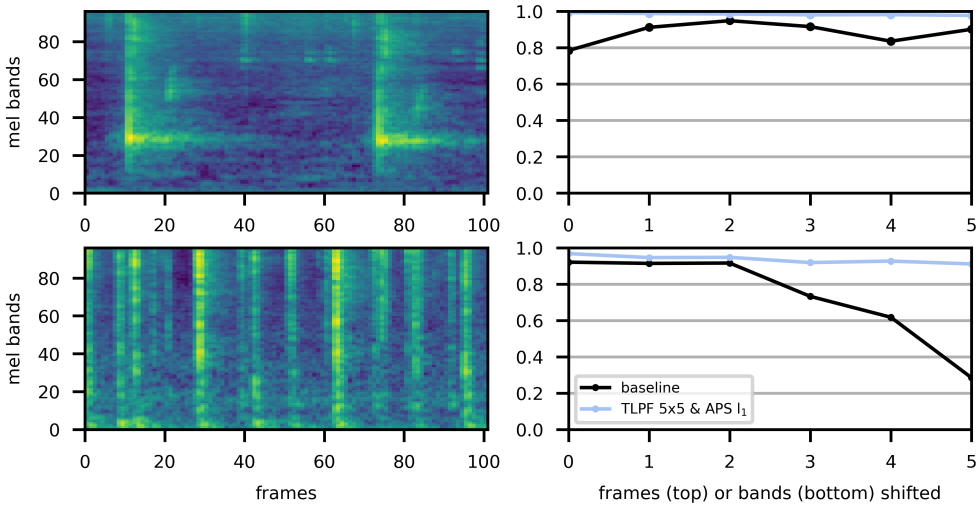

Figure 4.5: Predicted score for the correct class of water dripping (top) and computer keyboard typing (bottom) examples, as a function of shifted time frames (top) and mel bands (bottom). Inserting the pooling mechanisms (TLPF 5x5 + APS l_1) makes the predictions more stable against spectrogram shifts.

some degree (which is more unlikely with the time protocol). Third, in this protocol we are introducing small artificial perturbations never seen at training time, which may confuse the network. Regardless, the proposed approach exhibits higher robustness to the applied shifts in all cases analyzed. Similar trends are observed when using the proposed methods alone (either TLPF or APS). In summary, results of Table 4.5 demonstrate that the proposed pooling mechanisms increase shift invariance in the network.

Figure 4.5 shows the classification stability of the same models used for Table 4.5 with two examples: applying the time shift protocol over a water dripping sound (top) and the frequency shift protocol over a computer keyboard typing sound (bottom). In the top plots, predictions for the *Drip* class are stable when we insert the proposed pooling mechanisms, as one would expect upon shifting the signal framing by few miliseconds. However, the baseline predictions show certain fluctuations. Similarly, in the bottom plots the proposed network exhibits higher robustness against the frequency shifts and the induced perturbations.

4.4.5 Per-class Analysis

In this Section we include a per-class analysis to show how the proposed methods affect the different classes in FSD50K. Figure 4.6 shows the scatter plot

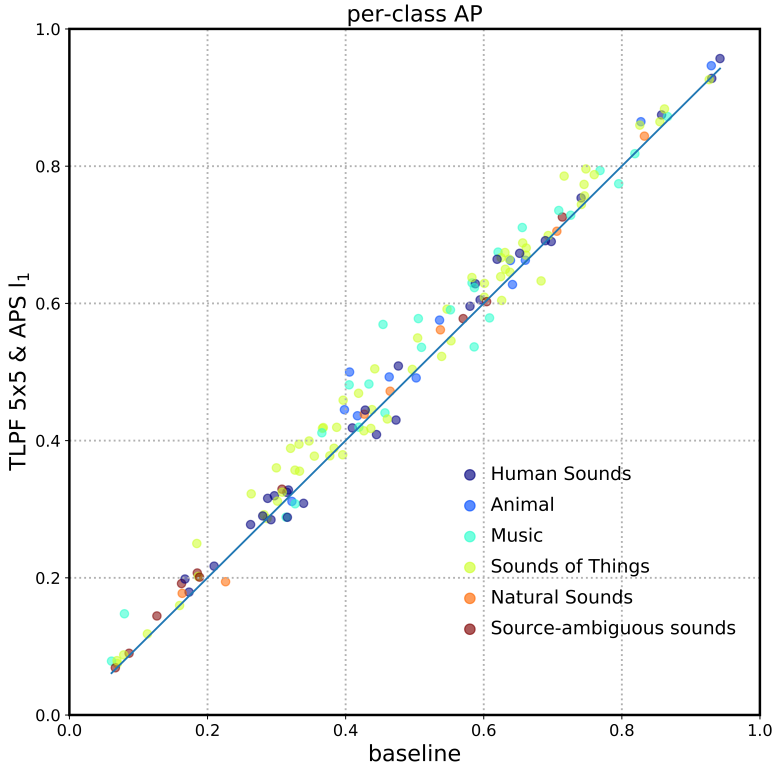

Figure 4.6: Per-class AP for the baseline using VGG42 and mixup vs. the same model after inserting TLPF 5x5 and APS l_1 (see Table 4.4). Only the 144 leaf nodes in FSD50K are displayed, grouped by main sound family in the AudioSet Ontology.

of per-class AP values for the baseline vs. topline systems using VGG42 and mixup. In particular, the baseline has no explicit measures to increase shift invariance, whereas the topline features TLPF 5x5 and APS l_1 (see right column of Table 4.4). The diagonal line divides the space into classes improved (above the line) or worsened (below the line) by the proposed method. Classes in Figure 4.6 are grouped according to the main sound families of the AudioSet Ontology (Gemmeke et al., 2017).[41] Only the 144 leaf nodes in FSD50K are displayed, i.e., we ignore the 56 intermediate nodes in this visualization (see Section 3.3.1 for more context on the distinction between leaf and intermediate nodes). It can be seen that the point cloud is slightly above the diagonal, with the majority of the classes undergoing an AP improvement, which can reach up to 0.1 approximately. Yet there are a few classes that are worsened by the inclusion of the proposed method. From the viewpoint of sound families, it

[41] In Figure 4.6, the sound family *Channel, environment and background* from the AudioSet Ontology is missing because none of the classes in FSD50K belongs to this family.

is hard to make interpretations as all the families feature both improved and worsened classes, and also the intra-family diversity can be very high.

Table 4.6 lists the 10 classes that benefit the most and the 10 classes that are harmed the most due to the inclusion of the proposed method. These are the classes showing the largest AP deviations from the diagonal in Figure 4.6 (which are also listed in Table 4.6 as ΔAP). By looking at Table 4.6, it is

Table 4.6: List of the 10 most benefited classes and 10 most harmed classes by inserting TLPF 5x5 and APS l_1 on VGG42 with mixup (see right column of Table 4.4). ΔAP is the increment (left)/decrement (right) of AP observed.

Top 10 Classes	**ΔAP**	**Worst 10 Classes**	**ΔAP**
Gong	0.115	Tick-tock	-0.050
Growling	0.094	Bass drum	-0.049
Accordion	0.076	Giggle	-0.043
Tambourine	0.072	Female singing	-0.036
Bicycle bell	0.069	Wind	-0.032
Wind chime	0.068	Finger snapping	-0.031
Harp	0.068	Drum kit	-0.030
Mechanical fan	0.065	Ratchet, pawl	-0.029
Fixed-wing aircraft, airplane	0.062	Sigh	-0.027
Cutlery, silverware	0.062	Organ	-0.026

not easy to observe very consistent patterns. Classes typically featuring a higher density of sharp spectro-temporal modulations can be found among the top classes (e.g., *Accordion* or *Harp*) but also among the worst classes (e.g., *Finger snapping* or *Ratchet, pawl*). Analogously, classes usually presenting smoother noise-like spectro-temporal variations are also found in both sides of the Table (e.g., *Mechanical fan* and *Fixed-wing aircraft, airplane* in the top classes; *Wind* and *Sigh* among the worst classes). A complete list of per-class increments/decrements due to the insertion of the proposed method is available in Figure E.1.

To gain a better grasp of what TLPF is doing, in Table 4.7 we list the analogous data, considering now the inclusion of TLPF 5x5 (but not APS l_1). Here, it seems that a somewhat more consistent pattern can be observed. Most of the top classes are prone to having sharp spectro-temporal modulations (with the addition of *Tabla* or *Frog* or *Acoustic Guitar*). Also, among the worst classes, some featuring noise-like spectrograms with smoother variations are more prevalent (e.g., *Boiling* or *Chirp, tweet* or *Bathtub (filling or washing)*). A possible explanation is that low-pass filtering feature maps with higher frequency components tends to be beneficial (as discussed in Section 4.2.1), whereas further

Table 4.7: List of the 10 most benefited classes and 10 most harmed classes by inserting TLPF 5x5 and IBP on VGG42 with mixup (see right column of Table 4.4). ΔAP is the increment (left)/decrement (right) of AP observed.

Top 10 Classes	**ΔAP**	**Worst 10 Classes**	**ΔAP**
Gong	0.085	Bass drum	-0.048
Tabla	0.082	Ratchet, pawl	-0.034
Frog	0.074	Boom	-0.030
Tambourine	0.072	Boiling	-0.029
Wind chime	0.071	Tick-tock	-0.028
Acoustic guitar	0.070	Drum kit	-0.026
Accordion	0.064	Vehicle horn, car horn, honking	-0.018
Growling	0.062	Chirp, tweet	-0.018
Idling	0.060	Female singing	-0.015
Bicycle bell	0.057	Bathtub (filling or washing)	-0.015

blurring time-frequency variations that are already smooth can be detrimental. It is conceivable that this kind of insight is less observable when combining low-pass filtering with another approach such as APS. However, a more systematic analysis is needed to make stronger claims. This could be carried out by estimating a metric of how sharp or *edgy* spectrograms are, and determining whether there is a correlation between the most benefited classes and the sharpness of their spectrograms.

Finally, in order to see the different behaviour of the two proposed methods, Figure 4.7 shows the scatter plot of per-class AP for TLPF 5x5 & IBP vs. APS l_1 using VGG42 and mixup (see right column of Table 4.4). In this case, we observe that the point cloud is centered around the diagonal line, showing that both methods perform similarly overall, with each method being slightly more proficient in a certain subset of classes. This suggests that joining both methods can lead to small boosts, as previously discussed from Table 4.4.

4.4.6 Discussion

We have seen that two methods with different underlying principles targeting the increase of shift invariance yield improvements within the same ballpark in our task. Further, we have empirically shown that they increase model's robustness to spectrogram shifts. These facts indicate that there is indeed some lack of this property in the CNN under test, and suggest that reinforcing shift invariance is beneficial for sound event classification. One interesting observation is that while anti-aliasing measures are helpful to increase performance and shift invariance, they do not seem strictly necessary in light of the overall similar performance attained by APS.

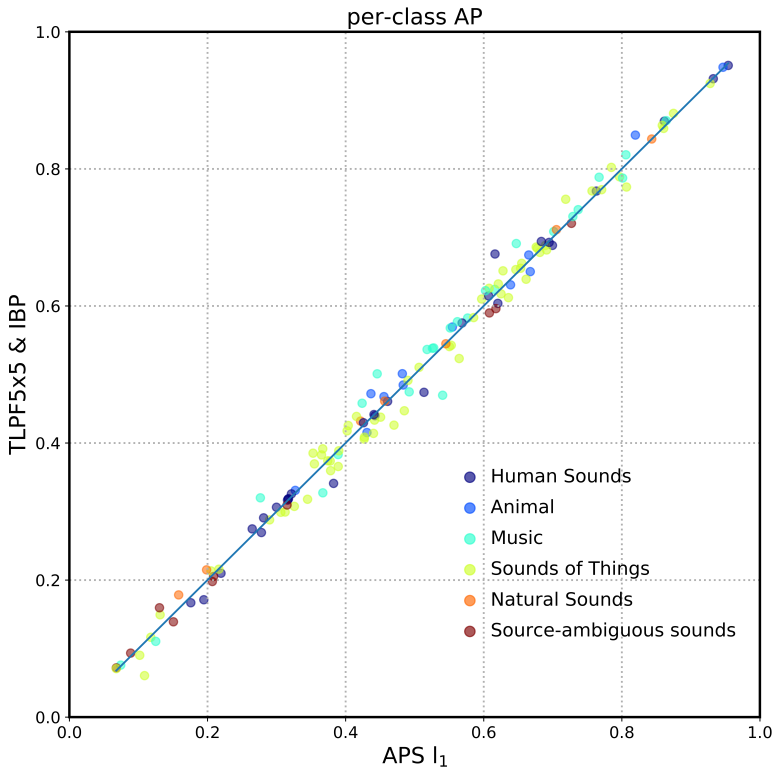

Figure 4.7: Per-class AP comparing the top configuration of each proposed method alone inserted into VGG42 with mixup: APS l_1 vs. TLPF 5x5 & IBP (see Table 4.4). Only the 144 leaf nodes in FSD50K are displayed, grouped by main sound family in the AudioSet Ontology.

In terms of model size, the impact is negligible for all evaluated methods. Specifically, TLPF 5x5 adds 6k (0.50%) and 12k (0.24%) trainable parameters over VGG41 and VGG42 respectively. Its non-trainable counterpart (BlurPool 5x5) adds the same number of non-trainable parameters. APS does not require any additional parameters (trainable or non-trainable). The additional compute required by the methods is also limited. For the low-pass filtering methods, one additional convolution is needed to apply the low-pass filter over the incoming feature maps for every subsampling operation. Analogously, the only additional compute required by APS is the computation of the polyphase components and their norms in every subsampling operation. Thus, the proposed architectural modifications (which apply only to the pooling layers) yield consistent recognition boosts when inserted into a well-known CNN, with minimal additional computation. This makes them an appealing alternative to conventional pooling layers.

Table 4.8: State-of-the-art on FSD50K.

Method	mAP	d'	lwlrap
Baseline VGG-like (Section 3.4.1)	0.434	2.167	0.514
PSLA (not using ImageNet) (Gong et al., 2021b)	0.452	-	-
Audio Transformers (Verma & Berger, 2021)	0.537	-	-
VGG42 + APS l_1 (ours)	0.538	2.415	0.575
VGG42 + TLPF5x5 + IBP (ours)	0.539	2.417	0.581
VGG42 + TLPF 5x5 + APS l_1 (ours)	**0.541**	**2.431**	**0.582**
PSLA (using ImageNet) (Gong et al., 2021b)	0.567	-	-

4.4.7 Comparison with Previous Work

Table 4.8 lists previously published results on FSD50K, including mAP, d-prime (d') and *lwlrap* when available. Our proposed systems outperform the baseline models evaluated in Chapter 3 by a large margin. In addition, our best system obtains state-of-the-art mAP of 0.541, slightly outperforming recent Transformer-based approaches (0.537) (Verma & Berger, 2021), as well as the PSLA approach when trained only on FSD50K (0.452) (Gong et al., 2021b). PSLA makes use of a collection of training techniques (ImageNet pretraining, data balancing and augmentation, label enhancement, weight averaging, and ensemble of several models) (Gong et al., 2021b). Among all of them, the key ingredient seems to be ImageNet pretraining, without which the performance decreases dramatically. While using transfer learning from ImageNet seems to provide substantial boosts, we consider transfer learning from external datasets a different track. Our proposed state-of-the-art approach consists of simple architectural changes inserted into a widely-used CNN at minimal computational cost along with simple augmentation.

4.5 Summary and Conclusion

In this Chapter, we have evaluated two pooling methods to improve shift invariance in CNNs in the context of a sound event classification task. These methods are based on low-pass filtering and adaptive sampling of incoming feature maps, and are implemented via small modifications in the pooling layers of CNNs. We have evaluated the effect of these architectural changes on the FSD50K dataset, using models of different capacity and in presence of strong regularization.

Results show that the models evaluated indeed present a problem of only-partial shift invariance—a property that is often taken for granted in CNNs—and that adopting the proposed methods to improve it yields recognition boosts. The improvements observed are within the same ballpark for both methods, despite them having different underlying principles, which allows for small further boosts via their combination. Inserting these pooling methods into VGG variants makes the networks exhibit higher robustness to time/-frequency shifts and small perturbations in the input spectrograms. These facts suggest that reinforcing shift invariance in the models evaluated is beneficial for sound event classification, which could also apply to other CNN architectures. For example, this improved property can be useful to enhance the recognition of sources producing sounds with slightly different time/frequency patterns. This is a convenient skill given the intra-class variation of everyday sounds, as introduced in Section 2.2.3. The proposed architectural changes applied to a widely-used CNN yield consistent recognition improvements with minimal additional computation, which makes them an appealing alternative to conventional pooling layers. Our best system incorporates these architectural changes and simple mixup augmentation in order to achieve a new state-of-the-art mAP of 0.541 on FSD50K.

CHAPTER 5

Training Sound Event Classifiers With Noisy Labels

5.1 Introduction

Until now, in this thesis, we have assumed that the labels accompanying training data are correct. However, as introduced in Section 1.3.1, this ideal state may not always be realistic. Large-scale audio datasets inevitably bring in label noise issues, since it is intractable to exhaustively annotate large amounts of audio. On the other hand, gathering data from web repositories and inferring labels automatically from metadata supports rapid train data collection, but at the likely cost of a substantial level of label noise. Consequently, label noise emerges as a pressing issue for the future of SET that can hinder the proper learning of classifiers, especially if they are based on deep networks (Arpit et al., 2017; Zhang et al., 2017).

This Chapter is dedicated to the study of techniques to improve performance in presence of noisy labels in SET. Our first contribution is *FSDnoisy18k*, an openly-available audio dataset that supports the investigation of real label noise, including an empirical characterization of the noise and a CNN baseline system (Section 5.2). The dataset is singly-labeled and it consists of a small amount of clean data, and a much larger amount of noisy data containing a substantial amount of real-world label noise. To our knowledge, no previous audio dataset has specifically provided for the study of label noise in SET.

Then, we explore simple and efficient approaches to mitigate the effect of label noise, that are agnostic to network architectures or learning settings. The main advantage of these approaches is that they can be easily incorporated into existing deep learning pipelines, requiring only minimal intervention of the learning pipeline, and no extra resources. First, we consider regularization techniques in Section 5.3. Common regularization methods against label noise

include weight decay and dropout, which act on the weights or hidden units of the network. Here, we study regularization techniques external to the model that can mitigate the effect of label noise. To regularize the model from the outside, we consider Label Smoothing Regularization (LSR) and *mixup*.

In Section 5.4 we first provide an empirical evaluation of noise-robust loss functions, originally proposed for image recognition. To our knowledge, this is the first time that these loss functions have been used for SET. In addition, using the information provided by loss functions, we explore sample rejection strategies to first identify and then discard potential noisy labeled examples during the learning process. This is done without additional networks or data resources, unlike other works in the literature, e.g, Han et al. (2018).

It must be noted that the techniques mentioned above address generic problems of labels. However, labelling everyday sound data has inherent problems, some of which are described in Section 5.2. To conclude this Chapter, in Section 5.6 we address the problem of *missing labels*, one of the big weaknesses of large audio datasets, and one of the most conspicuous issues for AudioSet. We propose a simple and model-agnostic method based on a teacher-student framework with loss masking to first identify the most critical missing label candidates, and then ignore their contribution during the learning process. Section 5.7 ends this Chapter with a summary of the main results and a discussion about our findings.

5.2 FSDnoisy18k: a Sound Event Dataset for the Study of Label Noise

In this Section, we introduce *FSDnoisy18k*, an openly-available audio dataset that supports the investigation of real label noise. The dataset contains 42.5 hours of audio across 20 sound classes, including a small amount of manually-labeled data and a larger quantity of real-world noisy data. In addition, we characterize the label noise of the dataset empirically, and provide a CNN baseline system. Experiments suggest that training with large amounts of noisy data can outperform training with smaller amounts of carefully-labeled data.

5.2.1 Dataset Creation

FSDnoisy18k was created before the release of FSD50K presented in Chapter 3. The annotations for FSDnoisy18k were gathered through Freesound Annotator using an early version of the final procedure and tools described in Sections 3.2.4 and 3.2.5, which we summarize next.

The source of audio content to create FSDnoisy18k is Freesound, as done with FSD50K. The 20 classes of FSDnoisy18k are drawn from the AudioSet Ontology: *Acoustic guitar*, *Bass guitar*, *Clapping*, *Coin (dropping)*, *Crash cymbal*, *Dishes, pots, and pans*, *Engine*, *Fart*, *Fire*, *Fireworks*, *Glass*, *Hi-hat*, *Piano*, *Rain*, *Slam*, *Squeak*, *Tearing*, *Walk, footsteps*, *Wind*, and *Writing* (Gemmeke et al., 2017). They are selected based on data availability as well as on their suitability to allow the study of label noise (see Section 5.2.2 for some specific examples). As a first step, we did a mapping of Freesound clips to the selected classes: we assigned a number of Freesound tags to every class and, for each class, we selected the Freesound clips tagged with at least one of the tags. This process led to a number of automatically-generated *candidate annotations* indicating the potential presence of a sound class in an audio clip. These annotations are at the clip level and hence are considered weak labels (although for some files the target signal fills the file entirely, which could be considered strongly-labeled). Next, a small portion of the candidate annotations was human-validated. We used an early version of the validation task described in Section 3.2.5 deployed in Freesound Annotator.[42] In this task, users verify the presence/absence of a candidate sound class in an audio clip with a *rating* mechanism. For every class, users are presented with a series of audio clips, and asked: *Is <class> present in the following sounds?* Users must select one of the responses: Present and Predominant (PP), Present but not Predominant (PNP), Not Present (NP) and Unsure (U), as defined in Table 3.1. Participants in the validation task included voluntaries from the Freesound community as well as researchers and students from the Music Technology Group.

Audio clips that ended up with multiple labels had all but one label removed in order to foster a type of label noise (see Section 5.2.2). Next, we defined a *clean* portion of the dataset consisting of *correct* labels, obtained by a second verification of the clips marked as PP. The remaining portion is referred to as the *noisy* portion. The **clean portion** of the data consists of audio clips whose annotations are rated as PP (almost all with full inter-annotator agreement), meaning that the label is correct and, in most cases, there is no additional acoustic material other than the labeled class. A few clips may contain some additional sound events, but they occur in the background and do not belong to any of the 20 target classes. This is more common for some classes that rarely occur alone, e.g., *Fire*, *Glass* or *Wind*. The **noisy portion** of the data consists of audio clips whose candidate annotations received no human validation, i.e., the only supervision comes from the user-provided tags. Hence, the noisy portion features a certain amount of label noise, which is characterized next.

[42] https://annotator.freesound.org/

5.2.2 Label Noise Characteristics

The label noise literature typically deals with synthetic noise imposed on the data (Reed et al., 2015; Tanaka et al., 2018; Zhang & Sabuncu, 2018). Whereas synthetic label noise allows precise control of noise conditions, it may result in unrealistic conditions. FSDnoisy18k features real label noise that can be representative of audio data retrieved from the web, particularly from Freesound. In Frénay & Verleysen (2014), a generic taxonomy of label noise from a statistical viewpoint is proposed, including models of label noise that differ in the dependencies among the agents involved. In Shah et al. (2018), two types of label noise are proposed (a generic label corruption noise, and a label density noise) for multi-label data based on AudioSet. We propose a taxonomy of label noise for singly-labeled data following an empirical approach. The taxonomy is shown in Figure 5.1 and includes the noise types identified through manual inspection of a per-class, random, 15% of the noisy data in FSDnoisy18k. Its concepts are explained next along with the main use cases found in FSDnoisy18k.

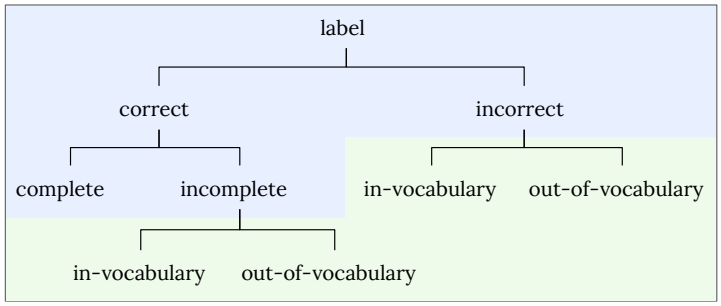

Figure 5.1: Taxonomy of label noise based on the analysis of the noisy data in FSDnoisy18k. The taxonomy considers two facets. Top levels (blue) of the taxonomy describe the observed label in terms of correctness and completeness. Bottom level (green) of the taxonomy categorizes the nature of the true or missing label, for an observed label that is incorrect or incomplete, respectively.

The taxonomy uses the terms *correct* and *complete* as defined in Section 2.3.2. In particular, the term *complete label* follows the *strict* definition introduced in Section 2.3.2: a label that fully describes *all* the acoustic content of the audio clip such that there is no additional acoustic material present. Then, we distinguish between additional events that are already part of our target class set (In-Vocabulary (IV)), or are not covered by those classes (Out-Of-Vocabulary (OOV)). Zhang & Sabuncu (2018) and Wang et al. (2018) use the terms *closed-set* for IV, and *open-set* for OOV. Given an observed label that is incorrect or incomplete, the true or missing label can then be further classified as IV or OOV.

Some classes are prone to include incorrect labels when the clips are retrieved only on the basis of their existing user-provided tags, e.g., *Bass guitar*, *Crash Cymbal*, or *Engine*; typically, the true label does not belong to the list of considered classes (*incorrect/OOV*). Other classes are prone to have audio clips with acoustic material that is additional to the provided (and correct) label, e.g., *Rain*, *Fireworks* or *Slam*, and, again, the missing label usually does not belong to the list of considered classes (*incomplete/OOV*). Finally, a few classes are related to each other. It can happen that one class contains clips that actually belong to another class in the dataset, e.g. *Wind* and *Rain* (*incorrect/IV*). Alternatively, two sound classes can co-occur in an audio clip, e.g. *Slam* and *Squeak*, despite only a single label is available (*incomplete/IV*). For completeness, correct and complete labels mean no label noise, i.e., clean data.

In addition to the aforementioned noise types, two more types arise in the context of web audio and Freesound in particular. First, determining whether a sound class is present in an audio clip can be subjective, even for an expert. This happens with human imitations or heavily processed sounds (e.g., with sound effects). We refer to these clips as *ambiguous* as it is unclear whether the label is correct or not. The second noise type relates to *i)* the variable clip lengths and *ii)* the weak nature of the clip-level labels. As mentioned in Section 3.3.2, a naive but common way of processing variable-length clips is to split them into several fixed-length patches, each inheriting the clip-level label (coined *false strong labeling* in Morfi & Stowell). This can generate false positives if the label is not active in a given patch. This type of label noise is conceptually similar to the label density noise of Shah et al. (2018).

The analysis of the noisy data revealed that roughly 40% of the analyzed labels are correct and complete, whereas roughly 60% of the labels show some type of label noise, whose distribution is listed in Table 5.1.

Table 5.1: Distribution of label noise types in a random 15% of the noisy data of FSDnoisy18k.

Label Noise Type	**Amount**
Overall	60%
Incorrect/OOV	38%
Incomplete/OOV	10%
Incorrect/IV	6%
Incomplete/IV	5%
Ambiguous labels	1%

The most frequent types of label noise correspond to out-of-vocabulary (OOV) problems, either in the form of incorrect labels (that generate false positives) or incomplete labels (which generate false negatives). Furthermore, we have observed that a few clips within the incorrect/OOV category are incorrectly labeled according to the semantic meaning of the class, and yet they are relatively similar (in terms of their acoustics) to the true label. For example, in *Clapping* there is a certain amount of applause sounds and claps generated by drum machines. We estimate that ≈10% of the clips analyzed shows this phenomenon, although it is highly subjective. This ≈10% is included in the 38% of incorrect/OOV labels. The label density noise is only relevant in few classes, especially *Slam*, and to a lesser extent *Fireworks* and *Fire*. This type of noise was quantified by counting the audio clips that present at least one segment of 2s (or more) where the observed label is not present (2s is the patch length used in the baseline system, see Section 5.2.4). The degree of total label noise per-class ranges from 20% to 80% roughly. A per-class description of the label noise similar to that of Table 5.1 is available at the dataset companion site in order to facilitate per-class analysis.[43]

5.2.3 Dataset Description

FSDnoisy18k contains 18,532 mono audio clips (42.5h) unequally distributed in the 20 aforementioned classes drawn from the AudioSet Ontology (Gemmeke et al., 2017). The audio clips are of variable length ranging from 300ms to 30s, and each clip has a single ground truth label (singly-labeled data). The dataset is split into a test set and a train set as seen in Figure 5.2.

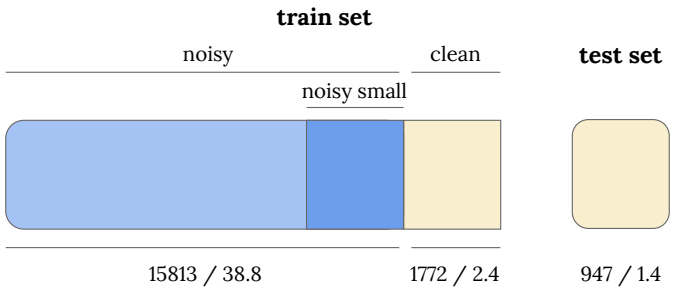

Figure 5.2: Data split in FSDnoisy18k, including number of clips / duration in hours. Blue = noisy data. Yellow = clean data.

[43] http://www.eduardofonseca.net/FSDnoisy18k/

The test set is drawn entirely from the clean portion, while the remainder of data forms the train set. The **train set** is composed of 17,585 clips (41.1h) unequally distributed among the 20 classes. The distribution of training clips across the 20 classes is shown in Figure 5.3. The total number of training audio clips per-class ranges from 316 to 1170. The train set features a clean subset and a noisy subset. In terms of number of clips their proportion is ≈10%/90%, whereas in terms of duration the proportion is slightly more extreme (≈6%/94%). The per-class percentage of clean data within the train set is also imbalanced, ranging from 6.1% to 22.4%. The number of audio clips per class ranges from 51 to 170, and from 250 to 1000 in the clean and noisy subsets, respectively. The limitation of 1000 clips per class in the noisy subset was imposed in order to mitigate data imbalance among classes.

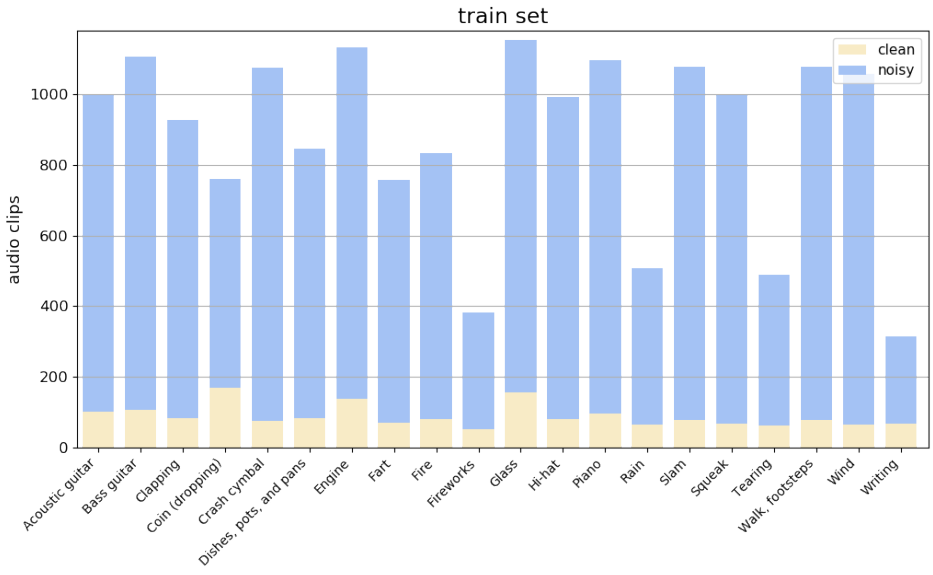

Figure 5.3: Per-class distribution of training clips in FSDnoisy18k. Blue = noisy data. Yellow = clean data.

Further, a *noisy_small* subset is defined (dark blue box in Figure 5.2), which includes an amount of (noisy) data comparable (in terms of duration) to that of the clean subset. The **test set** is composed of 947 clips (1.4h) that belong to the clean portion of the data. Its class distribution is similar to that of the clean subset of the train set. The number of per-class audio clips in the test set ranges from 30 to 72. The test set enables a multi-class classification problem. The dataset is openly available from its companion site,[43] along with the proposed data splits for reproducibility and a more detailed dataset

description. FSDnoisy18k is an expandable dataset that features a per-class varying degree of types and amount of label noise. The dataset opens the door to the evaluation of a variety of measures against label noise as well as other approaches, from semi-supervised learning, e.g., self-training (Elizalde et al., 2017) to learning with minimal supervision (Veit et al., 2017).

FSDnoisy18k has some overlap with FSD50K, especially in the clean portion of the dataset. At the same time, FSDnoisy18k contains audio data that are not present in FSD50K, especially in the noisy portion, given that this portion is not curated after its automatic retrieval.

5.2.4 Baseline System

In this Section we describe a baseline system for FSDnoisy18k, the main aspects of which are illustrated in Figure 5.4.

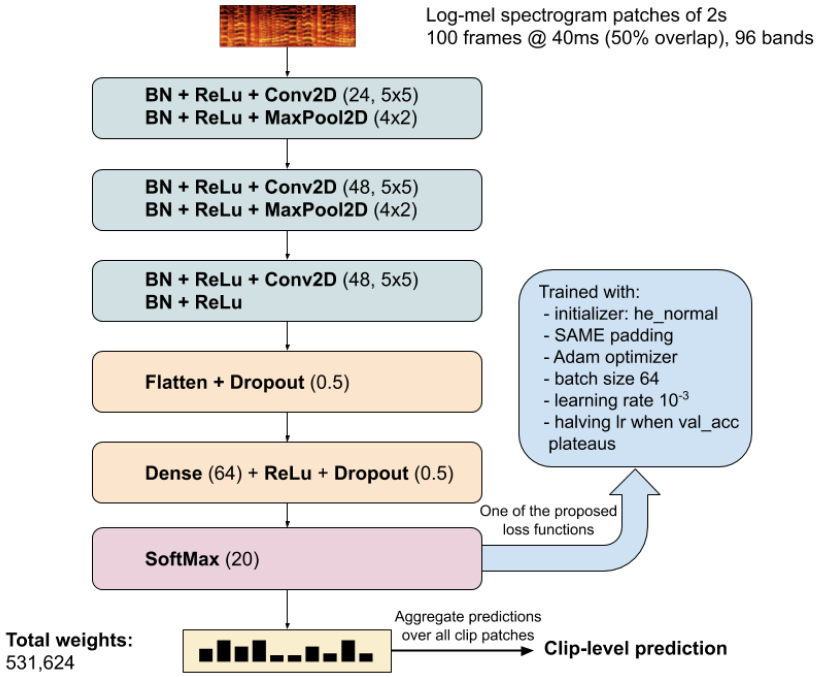

Figure 5.4: Baseline system for FSDnoisy18k.

Some of the techniques to combat label noise covered in subsequent Sections will be incorporated into this system. Incoming audio is transformed to 96-band, log-mel spectrogram as input representation. To deal with the variable-length clips, we use time-frequency patches of 2s (which is equivalent to 100

frames of 40ms with 50% overlap). Shorter clips are replicated while longer clips are sliced in several patches inheriting the clip-level label. The model used is a CNN featuring 3 convolutional layers and 1 dense layer following that of Salamon & Bello (2017), with two main changes. First, we include Batch Normalization (Ioffe & Szegedy, 2015) between each convolutional layer and ReLU non-linearity. Second, we use *pre-activation*, a technique initially devised in deep residual networks (He et al., 2016b) which essentially consists of applying Batch Normalization and ReLU as pre-activation before each convolutional layer. It was proved beneficial for acoustic scene classification in our previous work (Fonseca et al., 2018a), as well as in preliminary experiments with FSDnoisy18k. The model has ≈0.5M weights. The loss function is CCE, the batch size is 64, and we use Adam optimizer (Kingma & Ba, 2015) with initial learning rate of 0.001, which is halved whenever the validation accuracy plateaus for 5 epochs. Earlystopping is adopted with a patience of 15 epochs on the validation accuracy. To this end, a 15% validation set is split randomly from the training data of every class. The system is implemented in Keras and TensorFlow. The prediction for every clip is obtained by computing predictions at the patch level, and aggregating them with geometric mean to produce a clip-level prediction. The goal of the baseline is to give a sense of the classification accuracy that a well-known architecture can attain and not to maximize the performance. Extensive hyper-parameter tuning or additional model exploration was not conducted. Code for the training framework and a more detailed description of the baseline are available at the code release.[44]

5.2.4.1 Experiments

We present the experiments carried out with the baseline system. The results for different subsets of training are listed in Table 5.2.

Table 5.2: Average classification accuracy (%) and 95% confidence interval (across 7 runs) obtained by the baseline system using different subsets of FSDnoisy18k for training (see Figure 5.2); *all* = entire train set.

Approach	all	noisy	noisy_small	clean
baseline	71.6 ± 0.4	66.5 ± 0.6	44.4 ± 1.1	60.2 ± 0.5

From right to left, it can be seen that using the *clean* subset leads to an accuracy increase[45] of 15.8% with respect to using the *noisy_small* subset (consisting of roughly the same amount of data). However, curating the clean subset requires significant effort. Training with the *noisy* subset provides a

[44] https://github.com/edufonseca/icassp19

[45] Performance differences are expressed in terms of absolute accuracy.

boost of 6.3% over the performance obtained with the clean subset (despite the considerable amount of label noise present). Nevertheless, this improvement comes at the expense of using data that is an order of magnitude greater (see Section 5.2.3). Finally, using the entire train set, that is, adding a small amount of manually-curated data to the noisy subset, increases the accuracy by 5.1% over the performance obtained using only the noisy subset. The results suggest that large amounts of Freesound audio with the level of supervision provided by user-generated tags mapped to classes of the AudioSet Ontology can be a feasible option for training sound event classifiers. This can be useful in case of no labeling budget, as long as the computational resources can be accommodated. If only limited budget is available, curating a small portion of data to be combined with larger amounts of noisy data yields top performance.

5.3 Regularization Techniques to Handling Noisy Labels

Regularization aims to prevent overfitting and improve generalization, which can also be beneficial against label noise. Common regularization strategies include weight decay and dropout (Srivastava et al., 2014), which act on the weights or hidden units of the network. For example, dropout has been shown useful in reducing label noise memorization (Arpit et al., 2017). In this Section, we study some regularization techniques external to the model that can mitigate the effect of label noise. In particular, we consider LSR and *mixup*. The former operates on the ground truth labels, while the latter operates on both ground truth labels *and* input training examples, as illustrated in Figure 5.5. Noise-robust loss functions are covered in Section 5.4. We show that these simple regularization methods can be effective in mitigating the effect of label noise, providing up to 1.9% of accuracy boost when incorporated into a baseline CNN, while requiring minimal intervention and computational overhead.

As introduced in Section 2.3.1, we consider a dataset $\mathcal{D}$ with training examples $(\boldsymbol{x}, \boldsymbol{y})$ and C classes. Let $\boldsymbol{y} \in \{0,1\}^C$ be a vector representing the target label distribution for a given input example $\boldsymbol{x}$. In this Section, we consider a multi-class classification problem (i.e., only a single target label is available per input example). Hence, the ground truth distribution is a one-hot encoded vector $\boldsymbol{y} = \delta_{c,t}$, with $\delta_{c,t}$ being the Dirac delta, which equals 1 for the target class label, $c = t$, and 0 otherwise. Assuming CCE as default loss function, the training goal is to update the network weights in order to minimize CCE, which means maximizing the log-likelihood of the correct label.

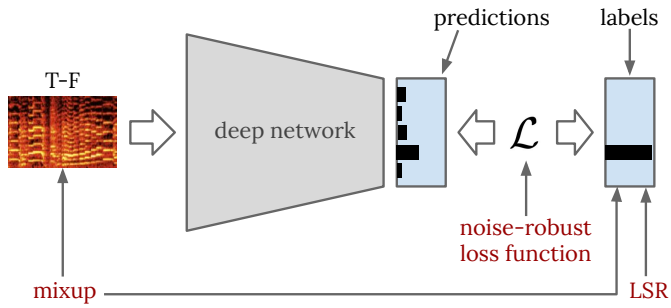

Figure 5.5: Sketch of the model-agnostic approaches against label noise considered in Sections 5.3 and 5.4, indicating the component(s) of the learning pipeline where they operate.

5.3.1 Label Smoothing Regularization

Maximizing the log-likelihood of the (potentially) correct label means encouraging the model to be confident about its predictions, which can be harmful in presence of training data with noisy labels. To address this issue, LSR models the noise on the labels using a smoothing parameter ε, as explained next (Szegedy et al., 2016). Given a training instance, the ground truth label distribution is changed from a one-hot encoded vector $\boldsymbol{y} = \delta_{c,t}$, to:

$$\boldsymbol{y}' = (1-\varepsilon)\delta_{c,t} + \frac{\varepsilon}{C}. \tag{5.1}$$

This is equivalent to the combination of the original distribution $\boldsymbol{y}$ weighted by $1-\varepsilon$, and another distribution (in this case, uniform, i.e., $1/C$) weighted by ε. Essentially, we *smooth* the label distribution by replacing the 1 and 0 hard classification targets with float soft targets (i.e., real-valued). Other distributions can be used to spread ε across the non-active labels, e.g., using prior knowledge of the dataset. By using LSR we are not only seeking to maximize the log-likelihood of training labels; we also aim at regularizing the model by promoting less confident output distributions without discouraging correct classification (Goodfellow et al., 2016). This property can make a classifier less vulnerable to label noise.

5.3.2 *mixup*

As mentioned in Section 2.3.4.1, mixup is typically understood as a data augmentation technique that acts as a regularizer by favoring linear behavior in-between training examples, encouraging networks to predict less confidently on linear interpolations of examples. Zhang et al. (2018) report that mixup

enhances the robustness of deep networks to corrupted labels for image recognition in an experiment inflicting artificial label noise to a clean dataset. Here, we are interested in evaluating whether mixup is helpful in a sound event classification setting with real-world label noise. mixup bears similarities with LSR in the sense that the external supervision is given in the form of multiple soft labels rather than with hard labels. However, mixup produces new examples with only two active labels (two-hot encoded vectors), whereas with LSR no label is strictly inactive due to the smoothing distribution of ε (see Equation 5.1). In addition, mixup modifies both target labels and input examples, unlike LSR which modifies only the target labels.

5.3.3 Experiments

The evaluation of the LSR and mixup in presence of label noise is conducted using the FSDnoisy18k dataset presented in Section 5.2. We use only the noisy set of FSDnoisy18k, composed of 15,813 audio clips (38.8h), and the test set, composed of 947 audio clips (1.4h) with correct labels. We evaluate these approaches by incorporating them to the baseline system of Section 5.2.4, following the experimental setup mentioned in that Section. Code for the experiments is available at the code release. [46]

5.3.3.1 Label Smoothing Regularization

First, the default version of LSR as described in Section 5.3.1 is evaluated for several ε, which implies a uniform distribution of ε across the non-active labels. This can also be seen from a probabilistic perspective, where the observed label is correct with probability $1-\varepsilon$ and, otherwise, any other label can be correct with equal probability (Goodfellow et al., 2016). Results in Table 5.3 indicate a small improvement over the baseline system. Larger values of ε do not lead to better scores in our experiments. In addition, we experiment with different smoothing strategies leveraging prior knowledge of FSDnoisy18k. A per-class estimation of the label noise is available at the dataset companion site. Based on this information, we group the audio categories in two groups, according to the estimated amount of noise (low/high). Specifically, the low-noise group is composed by the categories *Bass guitar*, *Clapping*, *Crash cymbal*, *Engine*, *Fire*, *Rain*, *Slam*, *Walk, footsteps* and *Wind*, while the high-noise group is complementary. Then, we assign a different ε to each group such that

$$\begin{aligned} \varepsilon_{low} &= \varepsilon - \Delta\varepsilon \\ \varepsilon_{high} &= \varepsilon + \Delta\varepsilon \end{aligned} \tag{5.2}$$

[46] https://github.com/edufonseca/waspaa19

Table 5.3: Average classification accuracy (%) and 95% confidence interval (across 7 runs) obtained by LSR and mixup approaches incorporated into the baseline system of Section 5.2.4.

Approach	**Accuracy**
Baseline (see Table 5.2)	66.5 ± 0.6
LSR ($\varepsilon = 0.1$)	66.8 ± 1.0
LSR ($\varepsilon = 0.15$)	67.1 ± 1.1
LSR ($\varepsilon = 0.15 \pm 0.05$)	68.1 ± 0.8
mixup ($\alpha = 0.1$)	67.1 ± 0.8
mixup ($\alpha = 0.2$)	66.6 ± 0.7
warm-up (10 epochs) & mixup ($\alpha = 0.3$)	68.4 ± 0.5

where we grid-search for $\Delta\varepsilon \in \{0.025, 0.05\}$ on the validation set, the latter providing best results. This simple way of encoding prior knowledge of label noise through a noise-dependent ε leads to the best LSR-based performance. However, a finer grouping of the categories in three levels of noise (low/mid/high) does not provide further gain.

We also experiment with non-uniform smoothing distributions in order to model per-class information, in particular: *i)* mapping each estimated amount of per-class noise to a per-class label energy within the target vector, and *ii)* using a distribution based on the number of per-class T-F patches in the dataset. However, adding this level of specificity leads to performance degradation in our experiments.

5.3.3.2 mixup

We conduct experiments with the default version as explained in Section 2.3.4.1 for several values of the interpolation strength α. Examples to be mixed up are log-mel patches drawn randomly from the training data. In particular, we try both intra- and inter-batch variants, that is, applying mixup to examples of the same batch after random permutation, or to examples of two different batches. No major differences are observed. In Zhang et al. (2018), the authors carry out experiments using mixup against memorization of corrupted labels and find out that larger values of α (e.g., $\{8, 32\}$) perform best. They hypothesize that increasing α creates virtual examples further away from the training distribution, thus hampering noise memorization. Surprisingly, in our experiments we find that larger values of α do not yield any improvement, the best accuracy being obtained with $\alpha = 0.1$, as seen in Table 5.3. The main

differences between our work and Zhang et al. (2018) are that FSDnoisy18k features real-world label noise (mainly of OOV type) in sound events, while Zhang et al. (2018) consider artificial label noise of IV type (i.e., randomly flipping labels of training examples) in images.

We also evaluate mixup by using a *warm-up* training period in which mixup is not applied. This is motivated by experiments conducted also in Zhang et al. (2018), in this case for speech recognition, although unrelated to label noise mitigation. We choose warm-up periods of 5 and 10 epochs, and we evaluate $\alpha \in \{0.1, 0.2, 0.3, 0.4, 1, 2\}$. Warm-up based mixup shows a significant improvement over its default version, as can be seen in Table 5.3, the highest accuracy being obtained with a warm-up period of 10 epochs and $\alpha = 0.3$. A possible explanation for the effectiveness of mixup is that continuously creating different virtual examples hinders label noise memorization. Also, it could contribute to reduce the overall exposure to label noise. For example, let us consider two training examples inputting mixup, of which only one is correctly labeled. Due to the properties of the beta distribution, the low range of α used means that one input example clearly dominates over the other in the new virtual example. Without mixup, we would be in a scenario where we learn from one correct label and another incorrect label. When mixup is applied, the scenario changes. We now learn either from one almost-correct label set (whenever the correctly labeled example dominates), or from another label set which is not entirely wrong (whenever the incorrectly labeled example dominates). Nonetheless, a more in-depth analysis would be required to better understand how mixup mitigates the effect of label noise.

5.4 Using Loss Functions to Mitigate the Effect of Label Noise

In this Section we first provide an empirical evaluation of noise-robust loss functions, originally proposed for image recognition. These loss functions are designed to provide some robustness against corrupted labels. To our knowledge, this is the first time that these loss functions have been used for SET. Then, we develop sample rejection strategies (also known as instance selection strategies) that use the loss values associated with training examples in order to detect and discard examples that are potentially noisy labeled. This is done without additional networks or data resources, unlike other works in the literature, e.g, Han et al. (2018). Evaluating the proposed methods, we show that noise-robust loss functions can be effective in improving performance in presence of corrupted labels. We also show that the proposed sample rejection strategies can be effective in mitigating the effect of label noise, providing up to 2.5% of accuracy boost when incorporated into two different CNNs, while

requiring minimal intervention and computational overhead.

5.4.1 Noise-Robust Loss Functions

As explained in Section 2.3.1, the training of a deep network is based on updating the network weights to minimize a loss function that expresses the divergence between the network predictions and the ground truth labels. If the ground truth labels are corrupted, the weights' update can be suboptimal thus hindering model convergence. In these cases, loss functions that are robust against label noise can be helpful. Next, we describe the noise-robust loss functions evaluated in this thesis and their underlying principles. All of them are modifications of the CCE loss commonly-used for multi-class classification. The CCE loss is given by Equation 2.2, which we include here again for convenience:

$$\mathcal{L}_{cce} = -\sum_{c=1}^{C} y_c \log(p_c), \tag{5.3}$$

where y_c is the c'th element of the target label (a one-hot encoded vector), p_c is the c'th element of the network predictions (the predicted class probabilities), and C is the number of classes in the vocabulary. Since $\boldsymbol{y}$ is a one-hot encoded vector, only one term of the summation in Equation 5.3 is different from zero, and therefore contributes to the loss.

The soft bootstrapping loss function, $\mathcal{L}_{soft}$, dynamically updates the target labels based on the current state of the model (Reed et al., 2015). More specifically, the updated target label is a convex combination of the current model's prediction and the (potentially noisy) target label. The idea is to pay less attention to the noisy labels, in favour of the model predictions, which are more reliable as the learning progresses. This approach is referred to as soft bootstrapping and the loss function $\mathcal{L}_{soft}$ is expressed by:

$$\mathcal{L}_{soft} = -\sum_{c=1}^{C} [\beta y_c + (1-\beta) p_c] \log(p_c), \quad \beta \in [0,1]. \tag{5.4}$$

The generalized cross-entropy loss, $\mathcal{L}_q$, is a generalization of CCE and Mean Absolute Error (MAE) proposed in Zhang & Sabuncu (2018). CCE is sensitive to label noise as it puts more emphasis on *hard* or *difficult* examples. Due to the logarithm in Equation 5.3, the examples for which the softmax predictions differ more from the target labels are also weighed more in the gradient update. This weighting property is beneficial when dealing with clean data, but it can be undesirable in the case of noisy labels. In contrast, MAE weighs all the predictions equally, a property that has been theoretically shown to be beneficial against corrupted labels (Ghosh et al., 2017). However, in preliminary experiments we obtained poor performance using MAE with FSDnoisy18k,

in accordance with Zhang & Sabuncu (2018), who reported difficulties when training with this loss function, leading to performance drop in other datasets.

To take advantage of the benefits of CCE (its weighting property) and MAE (its noise-robustness), a generalization of those functions has been recently proposed (Zhang & Sabuncu, 2018). This generalized cross-entropy loss is the negative Box-Cox transformation of the softmax predictions:

$$\mathcal{L}_q = \frac{1 - (\sum_{c=1}^{C} y_c p_c)^q}{q}, \quad q \in [0, 1]. \tag{5.5}$$

where q controls the closeness of $\mathcal{L}_q$ to CCE or MAE. When $q = 1$, Equation 5.5 reduces to the expression of MAE. When $q \to 0$, applying L'Hopital's rule to Equation 5.5 yields the expression of the CCE given by Equation 5.3. Figure 5.6 shows the curves for CCE loss as well as several instantiations of the generalized cross-entropy loss, for different values of q. It can be appreciated the different attenuation of loss contributions for the lowest predictions.

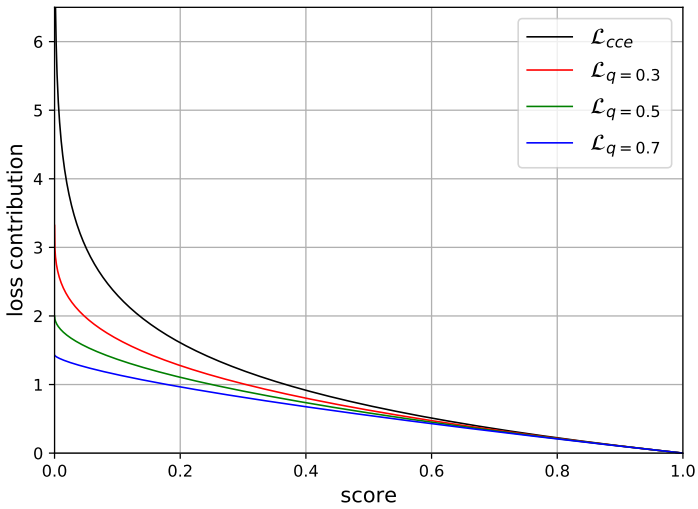

Figure 5.6: Generalized cross-entropy loss for different values of q (red, green, blue), and categorical cross-entropy loss (black), as a function of the predicted score.

5.4.2 Loss-based Instance Selection

Arpit et al. (2017) showed that deep neural networks in presence of label noise tend to learn first easy and general patterns from the underlying clean data, before fitting or memorizing the noise. In other words, the negative impact of label noise becomes more severe as learning progresses. This motivates us to view the learning process as a two-stage process.

In the *first stage*, we adopt $\mathcal{L}_q$ during a training period of n_1 epochs. While in the first epochs it is likely that label noise is not extremely critical (as long as there are some clean and simple instances in the dataset), it is not trivial to know when noise memorization kicks in as this depends on the data and the model; hence it is safer to adopt a noise-robust loss function.

After n_1 epochs, the model has converged to some extent and, therefore, it can be used for *sample rejection* or *instance selection* during training, a concept treated in the label noise literature (Han et al., 2018; Zhang & Sabuncu, 2018). Intuitively, when labels are corrupted, model predictions are likely to be less congruent with the noisy target labels, yielding artificially high losses (see left side of plot in Figure 5.6). By discarding them, we prevent data points that presumably feature corrupted labels from contributing to the total loss. This idea of using the loss associated with training instances as a proxy to discriminate between clean and noisy instances has been previously considered for image recognition (Han et al., 2018; Arazo et al., 2019). Thus, in the *second stage* of the process, we use the current state of the model to identify instances with large training loss. Assuming they correspond to noisy labeled examples, the goal is to reject them for the gradient update, thereby reducing noise memorization.

We experiment with two approaches to ignore large-loss instances in the second stage, as illustrated in Figure 5.7.

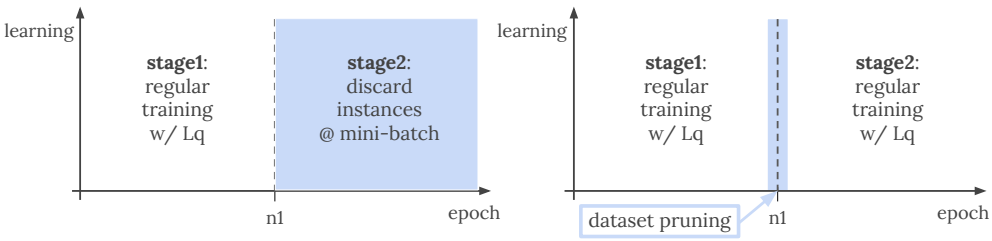

Figure 5.7: Approaches for loss-based instance selection. Left: discard instances from each mini-batch. Right: prune train set.

The first one consists of *discarding* large-loss instances from each mini-batch of data dynamically at every iteration. This is based on a time-dependent, noise-robust loss function that exhibits a change of behaviour as learning progresses, starting to discard potentially corrupted instances after n_1 epochs. The second approach consists of *pruning* the train set after n_1 epochs based on loss values, keeping only a subset of the train set to continue the learning process. To this end, we use the current model checkpoint to make predictions on the entire train set, and we compute the $\mathcal{L}_q$ losses associated to the softmax predictions.

Regardless of the approach (discarding from each mini-batch, or pruning the train set), the rejection of large-loss examples is as follows. We first compute an array of loss values $\mathcal{L}_q \in \mathbb{R}^{N\times 1}$, with the loss associated with every example in the mini-batch ($N = 64$) or the train set ($N = 13{,}441$ after keeping 15% for validation). Then, we define a threshold t_m such that elements in $\mathcal{L}_q$ greater than t_m are discarded for the computation of the total loss. We experiment with two simple ways of defining the threshold t_m: *i)* $t_m = m \cdot max(\mathcal{L}_q)$ with $m \in [0,1]$, and *ii)* $t_m = percentile(\mathcal{L}_q, l)$ where l is the percentile $\in [0,100]$.

5.4.3 Experiments

The evaluation of the loss-based approaches in presence of label noise is conducted using the FSDnoisy18k dataset presented in Section 5.2. Code for the experiments is available at the aforementioned code repositories.[44 and 46]

5.4.3.1 Noise-Robust Loss Functions

We present the experiments carried out with the proposed noise-robust loss functions, evaluated by replacing the CCE loss by each one of them in the baseline system of Section 5.2.4. Classification accuracy results for the different subsets of training data and loss functions are listed in Table 5.4.

Table 5.4: Average classification accuracy (%) and 95% confidence interval (across 7 runs) obtained by noise-robust loss functions using different subsets of FSDnoisy18k for training (see Figure 5.2); *all* = entire train set.

Approach	all	noisy	noisy_small	clean
baseline	71.6 ± 0.4	66.5 ± 0.6	44.4 ± 1.1	60.2 ± 0.5
$\mathcal{L}_{soft}, \beta = 0.3$	73.1 ± 0.6	66.8 ± 0.6	**46.0 ± 0.9**	–
$\mathcal{L}_{soft}, \beta = 0.7$	72.6 ± 0.6	67.6 ± 0.7	44.6 ± 1.0	–
$\mathcal{L}_q, q = 0.5$	73.4 ± 0.8	**68.4 ± 0.5**	45.0 ± 1.0	–
$\mathcal{L}_q, q = 0.7$	**74.3 ± 0.7**	66.7 ± 1.2	43.2 ± 1.2	–

We show results after fine-tuning the hyper-parameter of every loss function on the validation set. When training with the entire train set or the noisy subset, the top-performing loss function is consistently $\mathcal{L}_q$, followed by $\mathcal{L}_{soft}$. This means that $\mathcal{L}_{soft}$, and especially $\mathcal{L}_q$, originally proposed for image recognition, also work well for sound classification tasks. More specifically, $\mathcal{L}_q$ provides an accuracy increase over the baseline of 2.7% and 1.9% for the entire train set and noisy subset, respectively. The results confirm the insights in Zhang &

Sabuncu (2018), where it is shown that $\mathcal{L}_q$ works well with both OOV and IV noisy labels, which is the case of FSDnoisy18k.

When training with the entire train set, the noise-robust loss functions are applied selectively based on data origin, i.e., they are applied only to the data coming from the noisy subset, whereas for the clean subset the regular CCE loss is adopted. Specifically, this means: *i)* in $\mathcal{L}_{soft}$ the target labels are updated only for data points coming from the noisy subset; *ii)* when testing $\mathcal{L}_q$, only data points from the noisy subset contribute with $\mathcal{L}_q$ to the total loss. For $\mathcal{L}_{soft}$, and especially $\mathcal{L}_q$, this selective procedure leads to slightly better performance, in contrast to the naive way of mixing all the data and applying the noise-robust approaches indiscriminately. This suggests that $\mathcal{L}_q$ is more effective when a greater amount of label noise is present.

It is interesting to compare *i)* the accuracy boost obtained from adding manually-curated data to the noisy subset, versus *ii)* the boost resulting from using the noisy subset with the top-performing loss function. The baseline classification accuracy when training with the noisy subset is 66.5%. If we add a small amount of curated data, we obtain a 5.1% boost (see *all* column). Conversely, if we leverage the top performing $\mathcal{L}_q$ we obtain an increase of 1.9% (i.e., ≈37% of the boost by manual curation). Note that the manual curation requires a significant effort, while the latter approach requires very little engineering effort and adds minimal computational cost. Combining both approaches yields top performance.

5.4.3.2 Loss-based Instance Selection

We present the experiments carried out with the instance selection approaches. We evaluate these approaches by incorporating them to the baseline pipeline of Section 5.2.4. We use only the noisy set of FSDnoisy18k, composed of 15,813 audio clips (38.8h), and the test set, composed of 947 audio clips (1.4h) with correct labels. In addition to using the baseline model of Section 5.2.4, the proposed approaches are tested with a model of higher capacity illustrated in Figure 5.8. This model is a CNN based on Dense Convolutional Networks (DenseNet) (Huang et al., 2017), which has been shown to improve information flow in the network by connecting layers directly to all subsequent layers, combining their features by concatenation. In particular, we use four *dense* blocks composed of a bottleneck layer and a convolutional layer. In addition, we include Squeeze-and-Excitation (*SE*) blocks that calibrate the features extracted channel-wise by modelling channel interdependencies (Hu et al., 2018). These architectural blocks have been proven useful for sound event classification (Jeong & Lim, 2018). We refer to this model as *DenSE* due to its composition. DenSE is more accurate than the baseline (see Table 5.5) while using less weights (458k). Implementation details can be checked in the code

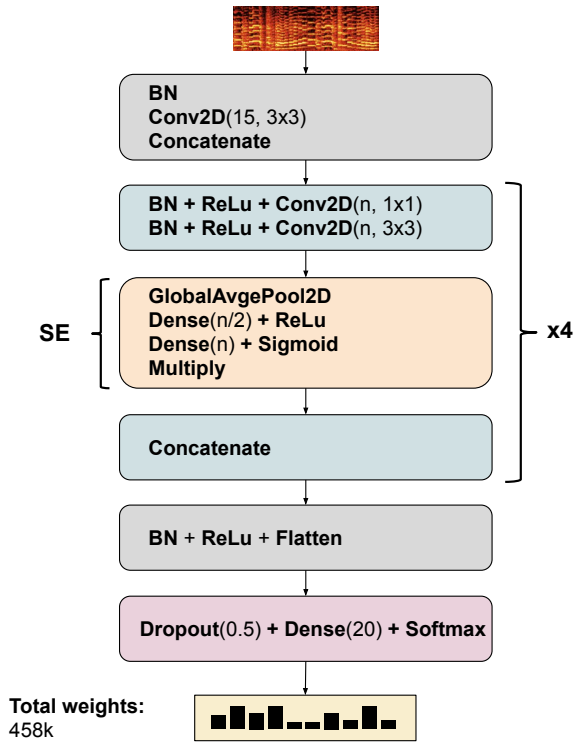

Figure 5.8: DenSE model based on DenseNet (Huang et al., 2017).

release.[46]

Table 5.5 lists the results for this evaluation. We include three baseline approaches (top section). Each of them follows a constant behaviour during the learning process, unlike the proposed approaches which feature two different learning stages (bottom section). The baselines are: *i)* using plain CCE loss ($\mathcal{L}_{cce}$); *ii)* adopting the *discard*-from-every-mini-batch approach using CCE loss since the beginning of the training ($\mathcal{L}_{cce,discard}$); *iii)* using plain generalized cross-entropy loss function ($\mathcal{L}_q$). The optimal values of every approach are determined through grid search on the validation set. Results suggest that discarding instances right from the scratch ($\mathcal{L}_{cce,discard}$) provides lifts with respect to not discarding anything at all ($\mathcal{L}_{cce}$). However, the accuracy obtained is still lower than using plain generalized cross-entropy loss ($\mathcal{L}_q$). A possible explanation is that using the model as an instance selector from the very beginning of training is suboptimal because the model needs some training period to gain proficiency. The better results obtained in the bottom section of Table 5.5 seem to back this hypothesis.

For the approach of *discarding* from every mini-batch after n_1 epoch, $\mathcal{L}_{q,discard}$,

we experiment with $n_1 \in \{10, 15, 20, 25\}$ epochs, $m \in \{0.93, 0.96, 0.99\}$, and using percentiles of loss values in order to discard $\{1, 3, 5\}$ T-F patches. Note that we are discarding T-F patches at every mini-batch, and not entire clips (as explained in Section 5.2.4). Results in Table 5.5 show accuracy boosts[47] of 0.4% and 0.6% over using plain $\mathcal{L}_q$, by discarding 5 patches/mini-batch and using $m = 0.93$ for the baseline and DenSE models, respectively.

Table 5.5: Average classification accuracy (%) and 95% confidence interval (across 7 runs) obtained by approaches for loss-based instance selection. Approaches in the top section follow a constant behaviour during the learning process. Approaches in the bottom section have two stages, as described in Section 5.4.2, where n_1 indicates number of epochs prior to the stage of instance selection, for (Baseline Model | DenSE).

Approach	**Baseline Model**	**DenSE**
$\mathcal{L}_{cce}$ (baseline in Table 5.2)	66.5 ± 0.6	67.9 ± 0.7
$\mathcal{L}_{cce,discard}$	67.7 ± 0.9	68.6 ± 0.7
$\mathcal{L}_q$	68.4 ± 0.5	69.2 ± 0.8
$\mathcal{L}_{q,discard}$ $(n_1 = 25 \vert n_1 = 10)$	68.8 ± 0.9	69.8 ± 0.7
$\mathcal{L}_{q,prune}$ $(n_1 = 20 \vert n_1 = 15)$	**69.0** ± 0.6	**70.2** ± 0.5

For the approach of *pruning* the train set, $\mathcal{L}_{q,prune}$, rejecting clips using percentiles yields better results. We explore pruning $\{100, 200, 300, 400, 500\}$ clips in the train set using the corresponding loss percentiles after $n_1 \in \{10, 15, 20\}$ epochs. In order to compute loss values at the clip-level, we aggregate the patch-level losses using arithmetic mean. Improvements of 0.6% and 1.0% can be seen over plain $\mathcal{L}_q$ by pruning 200 and 400 clips from the train set, for the baseline and DenSE models, respectively.

The approach based on $\mathcal{L}_{q,prune}$ slightly outperforms the one based on $\mathcal{L}_{q,discard}$ for the two models considered, although the differences lie within the confidence intervals. Additionally, we observe that results yielded by $\mathcal{L}_{q,discard}$ present more stochasticity than those of $\mathcal{L}_{q,prune}$. This could happen as the former discards patches from a tiny distribution of losses (64 instances in our case), compared to considering the entire train set loss distribution in the latter.

Regarding the models, DenSE attains higher accuracy boosts with respect to the best baseline (plain $\mathcal{L}_q$). Further, in general, results obtained with DenSE seem slightly more stable than with the baseline. We hypothesize this occurs due to the higher proficiency of DenSE, which allows better identification of the corrupted examples. Additionally, this model has less weights, which in principle makes it less prone to noise memorization. These aspects make DenSE more suitable for the proposed methods.

[47] Performance differences are expressed in terms of absolute accuracy.

5.5 Discussion

In this Section we provide a brief discussion of the approaches studied so far in Sections 5.3 and 5.4. One of the most important aspects, and the motivation for studying these approaches, is that they are easy to incorporate to existing pipelines. LSR and mixup can be added as simple functions within the data loader that feeds the network. $\mathcal{L}_{q,discard}$ can be implemented as a custom loss function by providing information of the current epoch. The approach based on $\mathcal{L}_{q,prune}$ can be easily added to any training procedure with few lines of code. Furthermore, all methods cause minimal computational overhead. The top-performing approaches on the baseline system are those based on loss-based instance selection, especially $\mathcal{L}_{q,prune}$, which provides an accuracy increase over the CCE baseline of up to 2.5%. It must be noted that while we prune the dataset only once, the pruning could be done several times in an iterative fashion until convergence, potentially improving performance. Also, this method can be used for dataset cleaning.

However, these approaches seem to be highly dependent on the period n_1 prior to instance selection and on the amount of rejected instances, which in turn can depend on the model used, the dataset and its type and amount of label noise. We also note that some of the reported accuracy scores feature not small confidence intervals. Beyond the non-deterministic nature of results obtained with GPU, we conjecture that some stochasticity is due to the noisiness of the labels. For instance, the validation set used to take decisions is composed of noisy labeled data. It is conceivable that the approach performing best on the validation set does not necessarily perform best on the cleanly-labeled test set, this being an inherent problem of evaluation with noisy labels.

5.6 Addressing Missing Labels in Multi-label Sound Event Classification

The label noise mitigation techniques studied so far address generic problems of labels, but not any specific pattern of label noise. However, labelling everyday sound data has inherent problems, some of which are described in Section 5.2 and also in Section 3.3.3 in the context of FSD50K. In this Section, we address the problem of *missing labels*, one of the big weaknesses of large audio datasets, and one of the most conspicuous issues for AudioSet (Gemmeke et al., 2017). Our contribution is two-fold. First, we propose a simple and model architecture-agnostic method based on a teacher-student framework to first identify the most critical potentially missing labels in AudioSet, and then ignore their contribution in the learning process through a loss masking approach. We then analyse the effect of the proposed method via a set of ex-

periments using two model architectures of different capacity and two train sets of different size. We find that a simple optimisation of the training label set can lead to a non-negligible improvement in recognition performance without additional compute. We also discover that most of the improvement comes from ignoring a tiny portion of the missing labels. The ultimate goal is to demonstrate how prior knowledge of a dataset can be leveraged to build simple, efficient, and model-agnostic solutions to improve recognition performance, which can complement other approaches focused on improving network architectures.

5.6.1 Missing Labels in AudioSet

We refer to *missing labels* as those labels that would be included in an ideal, exhaustive annotation but which are missing from the current set. The existence of missing labels in AudioSet is due to the dataset curation process. This process consisted of two steps: the compilation of a list of candidate labels per clip, and the human validation of the labels nominated in that list. This process is also adopted in other datasets, e.g., FSD50K. In the case of AudioSet, the list of candidate labels was compiled by means of a series of automatic methods, including the processing of the available metadata (e.g., video title and/or description) as well as a query-by-example method. These methods can be sub-optimal due to the high inter- and intra-class variation of sound events in the AudioSet Ontology (Gemmeke et al., 2017). In addition, the list of candidate labels was limited to a maximum of ten labels per clip. There are therefore several ways by which some existing sound events fail to be nominated by the system, or are nominated but ranked below the top ten, thus leading to missing labels. We call the nominated labels that have received human validation *explicit* labels (that can in turn be positive or negative, depending on the human rating being "Present" or "Not Present"). The remaining labels which are not proposed by the nomination system (the vast majority) are referred to as *implicit* negative labels, and have received no human validation. Hence, it is likely that some of the implicit negative labels are indeed missing (positive) labels.

AudioSet poses a multi-label problem, which is usually addressed by a deep network with an output layer composed by C binary classifiers, with C being the number of classes in the vocabulary, as mentioned in Section 2.3.1. In this setting, binary classification loss functions are typically adopted, composed by two terms, one accounting for the positive examples, and the other for the negative ones. The default option is binary cross-entropy, introduced in

Equation 2.3 which we show here again for convenience:

$$\mathcal{L}_{bce} = -\sum_{c=1}^{C} y_c \log(p_c) + (1 - y_c) \log(1 - p_c), \tag{5.6}$$

where p_c represents the network output prediction and y_c the ground truth label for class c. The implicit negative labels are considered negative examples (despite not having been rated), and are, therefore, covered by the second term of Equation (5.6). If a sound event is actually present, we want the model to emit a high score even if the "Present" label is missing. However, this virtuous behaviour will be penalized (with the penalty increasing for higher output predictions) due to the back-propagation of an artificially high loss contribution, which causes a misleading gradient update. We hypothesize this hinders the learning process to some extent.

5.6.2 Method

To address this issue, we propose a two-step strategy based on a teacher-student framework (Ba & Caruana, 2014) depicted in Figure 5.9.

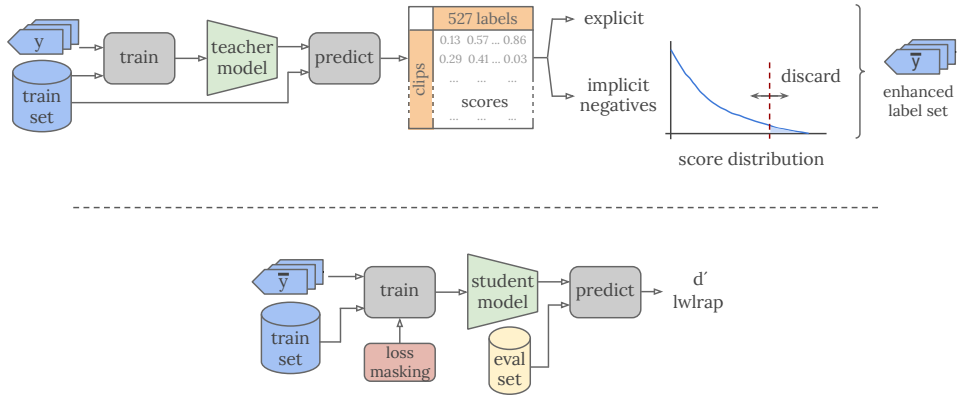

Figure 5.9: Proposed teacher-student framework. *Top*: Identification of potential missing labels per class using teacher's predictions and creation of enhanced label set. *Bottom*: Training a student model while ignoring missing labels through loss masking.

First, a teacher model is trained using the original AudioSet labels, $\boldsymbol{y}$, and the teacher's predictions are used to build a new enhanced label set, $\bar{\boldsymbol{y}}$, where the suspected missing labels are flagged. Second, $\bar{\boldsymbol{y}}$ is used to train a student model where the flagged labels are ignored through a loss masking approach. Next we explain the proposed method in detail.

The first step consists of identifying the potential missing labels per class. To do so, a *teacher* model is trained using the original AudioSet labels, $\boldsymbol{y}$. We use the trained teacher model to predict scores for the train set, leading to a set of $\mathbb{R}^{C\times 1}$ scores per audio clip. The teacher's predicted scores are used to take decisions on labels' veracity. We focus on the predictions associated with the implicit negative labels as explained in Section 5.6.1. Our hypothesis is that the top-scored implicit negative labels (henceforth, *top-scored negatives*) are likely to correspond mostly to missing "Present" labels, i.e., false negatives. Under this hypothesis, we rank implicit negative labels based on the teacher's predictions and we create a new label set, $\bar{\boldsymbol{y}}$, by flagging a given percentage of the top-scored negatives per class, with the intention of ignoring them in the student's learning. Note that, unlike other teacher-student pipelines where teacher's predictions are used as ground truth to train a student (e.g., via soft labels (Ba & Caruana, 2014; Li et al., 2014)), our case features a *skeptical* teacher whose supervision is used to highlight flaws in the current ground truth, estimating potentially missing labels and flagging them in a new label set. The outcome of this first step is an enhanced training label set, $\bar{\boldsymbol{y}}$, where the label information is encoded as multi-hot target vectors with three states (positive, negative, and to-be-ignored labels).

The second step consists of training a *student* model using the label set optimised through the teacher's predictions. The goal here is to ignore the loss contributions of the previously flagged labels in the loss function computation. This is done through a simple loss masking approach, where we minimally modify the student's learning pipeline so as to create a binary mask of size $C\times 1$ per input example, using the information encoded in the new target label vector $\bar{\boldsymbol{y}}$. Each element of the binary mask, M_c, is defined as (5.7)

$$M_c = \begin{cases} 0, & \text{if label is implicit negative and score} > t_c \\ 1, & \text{otherwise,} \end{cases} \tag{5.7}$$

where t_c is a per-class threshold computed as a given percentile of the per-class scores distribution. In practice, we compute the loss function $\mathcal{L}$ following (5.6) as usual, and M_c is applied to the negative term of $\mathcal{L}$ in order to discard the loss contributions of potentially missing labels, as follows

$$\mathcal{L}'_{bce} = -\sum_{c=1}^{C} y_c \log(p_c) + M_c(1-y_c)\log(1-p_c), \tag{5.8}$$

A similar masking approach to ignore false negatives in SER was recently used in Kim & Pardo (2019).

Another way to introduce this method is from the perspective of knowledge distillation (Hinton et al., 2015). A typical formulation of distillation is $\mathcal{L}(p_{teacher}, p_{student})$, whereas our method could be formulated as

$\mathcal{L}(f(p_{teacher}), p_{student})$. For f identity, standard distillation is recovered. We define one instantiation of f as a nonlinear transform applied to the teacher scores for the implicit negatives—our *skeptical* teacher. Similarly, other classes of f might also be relevant to accommodate label noise.

5.6.3 Experiments

5.6.3.1 Experimental Setup

We evaluate the proposed method using an internal version of AudioSet (Gemmeke et al., 2017), including information about which labels are explicit/implicit.[48] In order to study the impact of missing labels as a function of training data size, we use two train sets with similar class distributions but one roughly five times larger than the other, as specified in Table 5.6.

Table 5.6: Train sets and architectures used in our experiments.

Train set	**Clips**	**Hours**
tr_small	506,721	1407
tr_large	2,467,357	6853
Architecture	**Weights**	**Mult-Adds**
ResNet-50	30M	1860M
MobileNetV1	3.7M	69.2M

One of the motivations to choose the size of *tr_small* as roughly 0.5M clips is that this volume of data is closer to what could be gathered from other online repositories. For example, Freesound contains slightly over 0.5M audio clips at the time of writing (see Section 3.2.3). For evaluation, we use an internal *eval set* of 47,132 audio clips. Incoming audio clips are transformed to log-mel spectrograms using a 25ms Hann window with 10ms hop, and $F = 64$ mel log-energy bands. The network is presented with T-F patches of $T = 96$ frames (corresponding to a duration of 0.96s) with 50% overlap.

In order to assess the impact of missing labels on models of different capacity, we employ two CNNs as students: ResNet-50 (He et al., 2016a) and MobileNetV1 (Howard et al., 2017). Both are taken from the computer vision literature and have also proven successful in audio recognition research (see Hershey et al. (2017) and the recently released YAMNet[49]). ResNet-50

[48] The work presented in Section 5.6 was carried out during an internship at Google Research.

[49] https://github.com/tensorflow/models/tree/master/research/audioset/yamnet

is based on residual units, with shortcut connections that perform identity mappings added to the outputs of a small group of stacked layers. This allows information to pass through while leaving additional residual mappings to be learned, and was found to support the training of substantially deeper networks, thus yielding accuracy gains (He et al., 2016a). MobileNetV1 is based on depthwise-separable convolutions, in which a standard convolution (that filters and combines inputs into outputs in the same step) is factorized into *i)* a depthwise convolution that does the filtering and *ii)* a 1x1 convolution to combine the results into a set of outputs. This decomposition reduces computation and model size significantly Howard et al. (2017). Table 5.6 shows model size and Mult-Adds for both architectures. For training we use Adam optimizer (Kingma & Ba, 2015) and a fixed learning rate of 1e-5. We use random weight initialization with a standard deviation of 0.001.

For evaluation, we pass each 0.96s evaluation patch through the model to compute classifier output scores, which are then averaged per-class across all patches in a clip to obtain clip-level predictions, as done in Gemmeke et al. (2017). As evaluation metrics we use primarily d' and *lwlrap*, which we introduced in Section 2.3.7. In order to avoid the impact of potential missing positive labels in the evaluation set, only samples with explicit labels for a given class (both positive and negative) are used in the calculation of d' for that class. Because this excludes many "easy" samples, the resulting d' values are substantially lower than including all non-positive samples as negatives.) Both metrics are computed on a per-class basis, then averaged with equal weight across all classes to yield the overall performance shown in Table 5.7 and Figure 5.10.

5.6.3.2 Overall Performance vs. Discarded Negatives

As explained in Section 5.6.2, we first train a teacher model with the unmodified labels and use it to predict scores in the train set. We used an internal ResNet-50 model for the teacher which had been trained using several tweaks to improve performance, similar to those used in the publicly-released YAMNet model.[49] Based on the teacher's predicted scores, we generate a total of 18 new label sets, each of them using a different threshold t_c, i.e., discarding a different proportion of top-scored negatives in the train set. Finally, for every enhanced label set, we train a student model on the train set, and predict on the evaluation set, reporting the best performance obtained. While choosing parameters based on the test set introduces overfitting, our experience with data at this scale (i.e., validation and test sets in the range of hundreds of hours) is that results obtained by this suspect methodology are in practice similar to those from a more rigorous separation of tuning and evaluation sets.

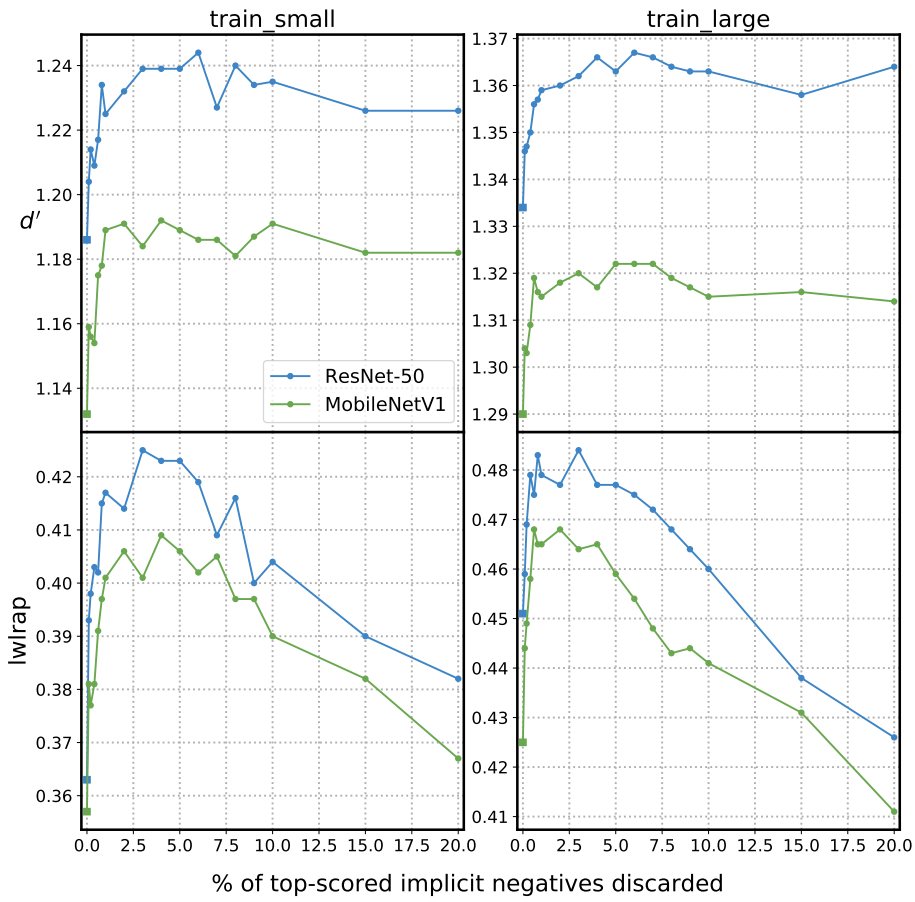

Figure 5.10: Classification performance as a function of the proportion of top-scored negative labels that are discarded. Each point in the lines corresponds to one operating point. The leftmost point in each curve, marked with a square, corresponds to using all negative labels.

The results in Figure 5.10 illustrate the impact of missing labels by plotting performance (d' and *lwlrap*) as a function of the amount of top-scored negatives discarded, similar to the treatment of noisy ImageNet labels in Northcutt et al. (2021). We experiment with progressively discarding $t_c \in \{0, 0.1, 0.2, 0.4, 0.6, 0.8, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20\}$ % of top-scored negatives for the two train sets and architectures mentioned. Each point in the lines is the result of one experiment trial using one label set with a given amount of discarded negatives. The leftmost point (at $x = 0.0\%$, marked with a square) corresponds to *normal training*, i.e., no labels ignored and all false negative labels included. We use it as our baseline.

Common to all the curves of Figure 5.10, we observe three regions from left to right: a steep increase at the beginning of the curve, followed by a sweet-spot, and a final decay that is more severe in *lwlrap* than in d'. A possible interpretation of this behaviour is as follows. We conjecture that the top-scored negatives correspond either to missing "Present" labels (i.e., false negatives (FNs)), *or* they are "decoys", difficult (and thus informative) true negatives (TNs), perhaps from similar classes, and especially useful in learning. First, we remove some critical FNs that damage the learning process, hence the sudden performance increase at the left of the curves. As we continue discarding more top-scored negatives, we keep removing FNs, but we also start to remove some TNs. Therefore, performance increases more slowly, until a sweet-spot is reached where both effects cancel out. Finally, if we keep ignoring more top-scored negatives, performance is degraded. As to why the decay in d' is much less pronounced than in *lwlrap*, a possible explanation lies in the way d' works. d' characterizes the separation of the positive and negative score distributions as the distance between their *means*. It may be that removing the high scoring tail changes the mean of the negative distribution (hence d' increases suddenly), but as we remove more labels with much more frequent scores the mean of the negative distribution barely changes (and consequently so d'). By contrast, *lwlrap* does not suffer from this issue, its curves showing a decay as expected. Based on this intuition, we consider the right end of the d' curves less reliable.

Table 5.7 lists the performances for baselines and best operating points for all the train sets and architectures considered. Based on the results of Figure 5.10 and Table 5.7, next we make a number of observations.

Table 5.7: Classification performance for baselines and best operating points for architectures and train sets considered.

Model	**Train set**	d'		***lwlrap***	
		baseline	best	baseline	best
ResNet-50	*tr_small*	1.186	1.244	0.363	0.425
	tr_large	1.334	1.367	0.451	0.484
MobileNetV1	*tr_small*	1.132	1.192	0.357	0.409
	tr_large	1.290	1.322	0.425	0.468

Effect of Ignoring the Highest Ranked Top-scored Negatives. The proposed method yields performance improvements in all cases considered. The best operating points are usually between 3 and 6% discarded for d', and between 1 and 4% for *lwlrap*. We believe this result is relevant as AudioSet

training examples are often treated as if they had complete labels. However, the most important pattern we observe in all cases is the consistent steep increase at the beginning of the curves. In all cases, most of the improvement comes from removing just ≈1% of the top-scored negatives. Further, in most cases, just by removing a tiny percentage (<=0.2%) of (potentially) missing labels, approximately half of the total boost is already attained. Two observations can be made from these findings. First, this indicates that a tiny portion of labels is troublesome and it is moderately affecting classifier performance, a concept which is basis for disciplines like instance selection, where it is assumed that not all training examples are equally informative, some of them being redundant and some being harmful (Liu & Motoda, 2002). Second, these findings become interesting as they contrast with the common trend of acquiring more and more training data to improve recognition performance, even if noisily labeled (Sun et al., 2017) (something we also find useful in our experiments in general).

Effect of Train Set Size. Table 5.7 shows improvements with respect to the baseline of ≈0.060 for d' when training with *tr_small* for both architectures, whereas when using *tr_large*, improvements are almost half of that (≈0.033). This relationship also holds for *lwlrap* when using ResNet-50,[50] whereas when using MobileNetV1, the performance difference between training with *tr_small* and *tr_large* is smaller. These results seem to indicate that the damage done by missing labels, and consequently the performance boost obtained by discarding them, can be higher when the dataset is smaller. A possible explanation is that larger amounts of data help to mitigate the effect of these errors in the label space, which accords with Sun et al. (2017). However, even when training with massive amounts of audio (almost 7000h, see Table 5.6), the impact of these labelling errors can still be observed. The d' sweet spot occurs roughly in the same region for both train sets. The *lwlrap* sweet spot seems to move slightly to minimal discards when training with larger amounts of data.

Effect of Model Architectures. The proposed method is effective for both model architectures considered despite having different underlying principles and significantly different numbers of parameters, in a proportion of around 8:1. The overall trend of the curves in Figure 5.10 is similar for both architectures. As can be seen in Table 5.7, in terms of d', both architectures show very similar improvements with respect to their corresponding baselines. In terms of *lwlrap*, however, results are inconsistent, with ResNet-50 providing a greater improvement than MobileNetV1 when training on *tr_small*, and vice versa when training on *tr_large*. We do not observe consistently larger improvements

[50] By chance, absolute improvements for both metrics are numerically similar in this case, despite the metrics are conceptually different and their numeric range is also different.

using ResNet-50, even though its much larger number of parameters might lead one to expect it to overfit labeling errors more readily. As an aside, regardless of missing labels, when comparing baselines, ResNet-50 outperforms MobileNetV1 as expected, but not by a particularly large margin considering the huge difference in number of parameters between the architectures.

Effect on Evaluation Metrics. By looking at Table 5.7, it can be seen that d' improvements reach up to relative 5.3% (MobileNetV1) and *lwlrap* improvements reach up to relative 17.1% (ResNet-50), both cases occurring when using *tr_small* ($\approx$ half a million clips), where improvements are more evident.

Finally, we carried out a small informal listening test in which we inspected some of the clips associated with the discarded top-scored negatives for a few classes. As expected, most clips were missing "Present" labels, some of them being flagrant labelling errors, but difficult to detect considering the train set size. These findings indicate that the proposed method, while simple, is effective in identifying missing labels in a human annotated dataset like AudioSet, and it is able to improve training over unnoticed missing labels. Additionally, it can be useful for dataset cleaning or labeling refinement. Re-labelling a small amount of flagged top-scored negatives may lead to better results than the proposed method. While the presented results are specific to AudioSet, we believe the insight and impact found can also apply to other large-scale audio datasets, especially those annotated via human validation of sub-optimally nominated candidates.

5.6.3.3 An Example of Applying the Method

For easier assimilation, we provide an example of application of the proposed method. Table 5.8 illustrates the details of the operating point of 0.1% discard in *tr_small* for the *Ambulance (siren)* and *Speech* classes. The total number of labels is the number of clips in the train set (506,721). The number of explicit labels (i.e., human rated, which are both positive and negative) is usually a tiny portion, in the range of a few hundreds or thousands (as in *Ambulance (siren)*), except for a few high prior classes such as *Speech*. Implicit labels (all negative) form the remainder of the clips. Note that *Ambulance (siren)* represents the typical case that holds for the vast majority of classes, while *Speech* represents an extreme case relevant to only a handful of classes. In this operating point, we ignore the top-scored 0.1% of the implicit negatives, which is usually around 500 labels per class, except for the few high prior classes, in which it is less.

Table 5.8: Label counts for two example classes at one operating point of Figure 5.10 (*tr_small* and discarding 0.1% of top-scored negatives)

Class	Total	Explicit	Implicit	To Ignore
Ambulance (siren)	506,721	1657	505,064	504
Speech	506,721	464,262	42,459	42

5.6.3.4 Per-class *lwlrap* Analysis

In this Section, we provide a brief per-class analysis to see how the proposed method affects the classes as a function of their prior in the dataset. As a case study we focus on *lwlrap* since the improvements are observed more easily, and we compare the baseline with the best operating point of ResNet-50 on *tr_small*. Figure 5.11 shows the scatter plot of per-class *lwlrap* values for baseline (i.e., no label rejection) vs. those of the best operating point (3% discard); the diagonal line divides the space into classes improved (above the line) or worsened (below the line) by discarding.

We divide the 527 AudioSet classes into three groups according to their prior: *i)* 15 largest classes with prior $\rho_c > 0.01$ (red), *ii)* 474 smallest classes with a prior $\rho_c < 0.00325$, corresponding, approximately, to the subset of 474 leaf nodes in the hierarchy of the AudioSet Ontology (Gemmeke et al., 2017) (blue), and *iii)* remaining 38 classes of medium size (green). Table 5.9 lists the number of classes in which performance improves, along with the average improvement, for every group of classes.

Table 5.9: Number of classes with improvement and average improvement for the three groups of classes in Figure 5.11.

Group	Classes	Classes w/ Improvement	Avg *lwlrap* Improvement
large	15	2 (13.3%)	0.082
medium	38	27 (71.1%)	0.086
small	474	359 (75.7%)	0.106

In light of Figure 5.11 and Table 5.9, we see the following. Classes with high prior tend to get slightly worse. While the performance changes observed are relatively small, this is is somewhat surprising as the number of labels ignored is even smaller in these cases—a possible explanation is that most of the labels being discarded correspond to informative TNs. On the contrary, groups of classes with medium and small priors present a similar percentage of classes showing improvement, being slightly larger in the group of small classes. In

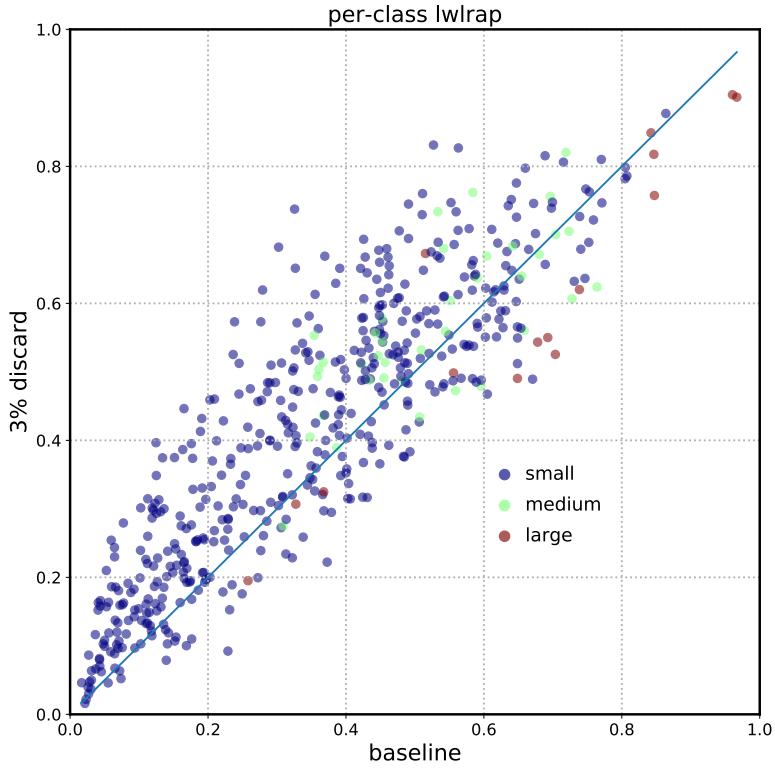

Figure 5.11: Per-class *lwlrap* for baseline (no label rejection) vs. best operating point (3% discard) for ResNet-50 on *tr_small*.

addition, the average improvement is also higher in the group of small classes, with an absolute difference of 0.02. While a more in-depth study is needed before making stronger claims, results seem to indicate that the impact of missing labels (and of the proposed method) is greater on classes with low prior, which goes in line with findings in Section 5.6.3.2 about the effect of train set size. Classes that benefit the most out of this process are: *Waterfall*, *Fusillade*, *Sizzle* and *Babbling*, featuring improvements greater than 0.3. The procedure carried out can be useful to detect classes with labelling errors, applicable in dataset cleaning or labeling refinement.

5.7 Summary and Conclusion

In this Chapter, we first have presented *FSDnoisy18k*, an openly-available dataset that facilitates the investigation of label noise in multi-class sound event classification. The dataset is singly-labeled and consists of a small

amount of manually-labelled data and a large amount of noisy data, featuring a per-class varying degree of types and amount of real label noise. An empirical characterization of the label noise in the noisy subset reveals that ≈60% of the labels show some type of label noise, mostly corresponding to OOV noisy labels. Experimental results with a CNN baseline system suggest that large amounts of Freesound audio with the level of supervision given by the user-provided tags can be a feasible option for training sound event classifiers.

Then, we explored simple and efficient approaches in order to improve the performance of sound event classifiers trained in presence of noisy labels. We strived for proposing approaches that are agnostic to network architectures and learning settings, hence they can be easily incorporated into existing learning pipelines. The simple regularization methods studied can be effective in mitigating the effect of label noise, providing up to 1.9% of accuracy boost when incorporated into a baseline CNN in our experiments. In particular, we found out that small variations of the original methods are beneficial in our task, namely, encoding prior knowledge of label noise through a noise-dependent $\boldsymbol{\varepsilon}$ in LSR, and using a warm-up training period before mixup. We observe a similar behaviour with the considered noise-robust loss functions, which achieve on par performance boosts. We also find that the noise-robust loss functions considered tend to be more effective when a greater amount of label noise is present. Both regularization mechanisms and noise-robust loss functions require very minimal engineering effort.

Going one step further, our results suggest that rejecting the contribution of noisy samples during training (or the contribution of noisy labels associated to some samples) can be more effective than previous approaches based on *accepting* the noisy labels and mitigating their effect—at least for the regularization methods and robust loss functions considered. In particular, when incorporated into the training of two different CNNs, the considered loss-based sample rejection methods provide absolute accuracy boosts of up to 2.5% when training on the noisy subset of FSDnoisy18k. Specifically, our top approach uses the model being trained as an instance selector to prune the train set during the learning process. Remarkably, this is done without additional networks or data resources, unlike other works in the literature, e.g, Han et al. (2018).

In the case of addressing missing labels for multi-label AudioSet classification, the first important takeaway is that we have identified missing labels as a pathology in the labelling of AudioSet, which is often ignored. By using a teacher-student framework with loss masking to identify and ignore the most critical potentially missing labels, we obtained multiple relevant findings: *i)* most of the improvement comes from filtering out a tiny portion (<1%) of the most critical estimated missing labels, showing a moderate impact on performance; *ii)* the damage done by missing labels (and the performance boost obtained by discarding them) becomes higher as the train set gets smaller—

however, even when training with massive amounts of audio, the impact of these labelling errors can still be observed; *iii)* when applied to two CNN architectures of different nature and size, the proposed method is effective in identifying missing labels and it is able to improve training over unnoticed missing labels, behaving similarly in both cases. It is conceivable that these insights will also apply to large-scale audio datasets beyond AudioSet, since the problem of missing labels is endemic. The main shortcoming of sample/label rejection is the risk of mistakenly discarding legit valuable information due to flaws of the detection mechanisms, which may lead to performance degradation. A clear direction for future work consists of improving noise detection mechanisms and making them less dependent on external decisions and tuning.

Overall, one of the main advantages of the proposed methods is that they can be easily plugged into existing deep learning pipelines, requiring only minimal intervention of the learning pipeline, negligible computational overhead, and no extra resources such as clean data or auxiliary classifiers. By showing the aforementioned improvements with purposely-chosen simple methods, we have demonstrated that label noise is indeed a problem in SET, and addressing it can bring benefits. In particular, we have identified missing labels as a pathology in the labelling of AudioSet (and possibly in everyday sound labelling in general). Increasing robustness against label noise is a promising research direction that may lead to performance boosts when audio labels are unreliable, and that can relax the dependency on careful and expensive manual annotations. Considering their simplicity and efficiency, the proposed approaches show promise in increasing robustness against noisy labels. However, in absolute terms, the performance boosts achieved are not very large. We believe our work puts some grounds and paves the way for the investigation of more complex approaches.

Importantly, as some of our experiments indicate, in order to obtain better performance, substantially larger amounts of noisy data may be needed to compensate supervision's quality with quantity, compared to the scenario of training with clean labels. The success of each methodology (using noisy or clean labels) will depend on factors such as the ratio of noisy labeled data vs. clean labeled data, as well as the type and amount of label noise. While collecting noisy labeled data will be generally cheaper than carefully annotating audio, larger volumes of data may require also larger computational resources, which may not be available to everyone. Another factor that must be considered is the source(s) for the clean and noisy labeled data. For example, in the experiments of Sections 5.2 to 5.4, both the noisy data and the clean data used for training come from Freesound, same as the data used for evaluation. However, in other possible scenarios, the noisy labeled data may come from a different source than the target data—for example, in the case where the noisy labeled data for training are gathered from some web audio repository not par-

ticularly related with the target task under consideration. In this case, issues of domain mismatch between training and evaluation data could occur, which may represent an obstacle for models' generalization. Lastly, as explained in Section 2.6.2, we believe that the label noise work carried out in this thesis has contributed to the progressive consolidation of this research direction in the SER community.

CHAPTER 6

Self-Supervised Learning of Sound Event Representations

6.1 Introduction

So far in this thesis, we have assumed that textual labels accompanying the audio data are always available, either provided by humans or automatically derived from audio metadata. Either way, the presented supervised learning algorithms depend on this external supervision, which carries important limitations. The construction of human-labeled audio datasets for SER is notoriously time-consuming and subjective, imposing limitations on dataset size and label quality. Likewise, automatically generating labels from metadata can produce substantial levels of label noise, which can hinder successful supervised training of classifiers.

As introduced in Section 1.3.2, the alternative to the supervised learning paradigm consists of using learning algorithms that do not depend on any external supervision, but that have the ability to extract supervision from the audio data. These *self-supervised learning* methods aim at *learning representations* without the need for external supervision. Absent explicit labels, the success of these methods relies on the design of *proxy* learning tasks in which pseudo-labels are generated from patterns in the data. By solving proxy tasks on unlabeled data, these methods learn mappings from input examples to low-dimensional representations, that can then be used for downstream tasks such as sound event classification. The main advantage of the self-supervision paradigm is that potentially unlimited data could be exploited, without need for prior manual labelling or metadata.

This Chapter explores multiple strategies to learn general-purpose audio representations from unlabeled data. We focus on self-supervised *contrastive* audio representation learning, where representations are learned by comparing pairs

of examples selected by some semantically-correlated notion of similarity (Le-Khac et al., 2020). Specifically, comparisons are made between positive pairs of "similar" and negative pairs of "dissimilar" examples, with the goal of learning a representation that pulls together positive pairs and thus reflects semantic structure, as introduced in Section 2.7. In order to generate pairs of positive examples, we create different versions or *views* of the original input examples. These views are created via compositions of data augmentation methods, which are one of the key elements in our learning pipelines.

The rest of the Chapter is organised as follows. Section 6.2 introduces the data augmentation methods that will be used in the subsequent Sections to create differently-augmented views of sound events.

In Section 6.3, we propose the proxy task of similarity maximization to learn representations by contrasting differently-augmented views of sound events. The views are computed primarily via temporal proximity sampling, followed by mixing of training examples with unrelated backgrounds, and other data augmentations. We first analyze the main components of the proposed pipeline via ablation experiments. Then, we evaluate the learned representations using a linear evaluation protocol, and in two in-domain downstream sound event classification tasks, namely, using limited manually labeled data, and using noisy labeled data.

In Section 6.4, we explore the use of unsupervised automatic sound separation to decompose sound scenes into multiple semantically-linked views for use in contrastive learning. In addition, we go one step further and propose to jointly optimize the proxy tasks of similarity maximization and coincidence prediction. We evaluate the learned representations as features for a query-by-example retrieval task, as well as on the established downstream shallow-model AudioSet classification task, where our unsupervised audio representation rivals state-of-the-art alternatives. Section 6.5 concludes the Chapter with a summary of the key results and a discussion about our main findings.

6.2 Data Augmentation to Create Different Example Views

In this Section, we introduce the data augmentation methods that will be used in this Chapter to create differently-augmented views of sound events. We seek to learn sound event representations by contrasting differently-augmented views of sound events. To this end, a number of semantics-preserving augmentations are explored in order to create different example views. In order to come up with good views for contrastive learning, Tian et al. (2020) argue that their mutual information must be reduced while the downstream semantically-

relevant information between the views is retained. Another important observation is that, since contrastive learning pulls together representations of positive views, the proxy task attempts to ignore the transformations applied to create them. Consequently, how the pairs of views are generated determines the invariant properties promoted in the learned representation.

Using a single transformation to generate data views has been shown as inferior to the composition of several augmentations, which is essential to obtain effective representations for vision tasks (Chen et al., 2020b). By composing multiple augmentations, the goal is to define a more challenging learning task so that higher-quality representations can emerge. However, not all compositions are necessarily valid; rather, the elements in the composition must adequately complement each other (Chen et al., 2020b). In this Chapter, we use a variety of data augmentation methods:

- temporal proximity sampling,
- a method based on example mixing (*mix-back*),
- a variety of other simple augmentations such as random resized cropping or SpecAugment (Park et al., 2019), and
- automatic sound separation.

To our knowledge, this is the first time that some of these augmentations are used to define proxy tasks for sound event representation learning, such as automatic sound separation, random resized cropping or SpecAugment. Our proposed example mixing augmentation (*mix-back*) is introduced in Section 6.3.1.1 along with other simple augmentations. The usage of automatic sound separation as data augmentation for representation learning is argued in Section 6.4.1. Here, we describe temporal proximity sampling as it is the one augmentation that we always use in our learning pipelines.

Temporal proximity sampling consists of randomly selecting two audio snippets as basic units to construct pairs of examples, instead of leveraging entire audio clips. These audio snippets are typically short—in our case, around 1 second. When randomly sampling audio snippets within a prescribed temporal proximity, we are likely to sample *i)* the same sound sources emitting somewhat different acoustic patterns as they evolve over time; or *ii)* different sources that are related semantically or casually with the initial ones. Thus, the temporal coherence among neighboring audio snippets implies a natural form of data augmentation. This simple method has been proven effective in our experiments of Sections 6.3.3.1 and 6.4.4.1 (Fonseca et al., 2021c,b), as well as in many other contrastive audio representation learning works (Jansen et al., 2018, 2020; Saeed et al., 2021; Wang & van den Oord, 2020), analogous to the common practice of random cropping with images (Chen et al., 2020b,c).

6.3 Similarity Maximization for Sound Event Representation Learning

In this Section, we propose to learn sound event representations using the proxy task of *similarity maximization*, where we contrast differently-augmented views of sound events. The different views are computed via temporal proximity sampling, mixing of training examples with unrelated background examples, and a composition of other data augmentations. We first provide an empirical evaluation of the different components of the proposed method through an ablation study. Then, we evaluate the learned representations using the linear evaluation protocol, and in two downstream sound event classification tasks, namely, using limited manually annotated data, and using noisy labeled data. Our results suggest that unsupervised contrastive pre-training can mitigate the impact of data scarcity and increase robustness against noisy labels. To our knowledge, the work included in this Section is the first one conducting contrastive sound event representation learning by maximizing the similarity between differently-augmented views created with multiple augmentations. Code for the experiments is available.[51]

It is interesting to note that other approaches similar to what we propose in this Section—published in Fonseca et al. (2021c)—were also proposed by other researchers at a similar time (Saeed et al., 2021; Wang & van den Oord, 2020). Both Fonseca et al. (2021c) and Saeed et al. (2021) were presented at the 2021 IEEE "International Conference on Acoustics, Speech and Signal Processing", and were uploaded to arXiv within weeks of difference in October and November of 2020.[52] The work by Wang & van den Oord (2020) was presented in December of 2020 at the NeurIPS 2020 "Self-Supervised Learning for Speech and Audio Processing Workshop".

Specifically, Saeed et al. (2021) propose the *COntrastive Learning for Audio* (COLA) approach. The main differences of COLA with respect to our approach are: *i)* the only augmentation used is temporal proximity sampling; *ii)* the contrastive loss function has a different scoring function, as we will see later in Section 6.3.1.2. The main difference of the approach of Wang & van den Oord (2020) with respect to ours is the format of the data used as starting point in the learning pipeline. In order to compute different views of the same input example, their starting point is the raw waveform of examples *and* its corresponding log-mel spectrogram. This multi-format strategy is shown to provide significant gains compared to the single-format counterpart (typically, using only log-mel spectrograms, as we do in our experiments). However, performance gain comes at the expense of needing two different encoder networks

[51] https://github.com/edufonseca/uclser20

[52] https://arxiv.org/abs/2011.07616 and https://arxiv.org/abs/2010.10915

(one per audio format) instead of one (see Figure 6.1).

6.3.1 Learning Framework

We seek learning sound event representations from unlabeled data via contrastive learning using the proxy task of similarity maximization. This proxy task consists of maximizing the agreement between differently-augmented views of the same audio example. To do so, we first create pairs of correlated views (denoted as *positive examples*) via different augmentations of a single sound event example. Then, their corresponding embedding representations are compared using a contrastive loss that pulls together representations of positive examples, while pushing apart those of negative ones (i.e., unrelated examples) (Chen et al., 2020b; Nandan & Vepa, 2020). In other words, this type of loss attempts to co-locate the representations of two positive examples in the same spot of the embedding space, thus promoting invariance to the transformations applied to generate the views. This task is based on recent work on visual representation learning (Chen et al., 2020b), commonly referred to as *SimCLR*. Our hypothesis is that discriminative sound event representations can emerge by solving this task. Figure 6.1 illustrates the main components of the proposed method, which we explain next. Implementation details and hyperparameter choices can be inspected in the released code.[51]

A pair of *positive* examples is constructed by selecting two audio snippets within the *same* audio clip. Analogously, a pair of *negative* examples is constructed by selecting two snippets from *different* clips. This is based on the assumption that, generally, two snippets within a given temporal proximity are much more likely to be semantically related than two snippets from independent recordings.

6.3.1.1 Data Augmentation Front-end

Temporal Proximity Sampling. The incoming training examples to our framework are log-mel spectrograms of audio clips. From each training example, $\mathcal{X}$, we sample two views, which we call T-F *patches*. These patches, $x_i \in \mathcal{X}$ and $x_j \in \mathcal{X}$ are selected randomly over the length of the clip spectrogram. However, there are other ways to select $\{x_i, x_j\} \in \mathcal{X}$. Section 6.3.3.1 analyzes the benefits of sampling $\{x_i, x_j\}$ at random over other alternatives.

Mix-back. The first operation that we apply to each incoming patch is what we call *mix-back*. It consists in *mix*ing the incoming patch x_i with a *back*ground patch, b_i, as follows:

$$x_i^m = (1-\lambda)\, x_i + \lambda \left[E(x_i)/E(b_i)\right] b_i, \tag{6.1}$$

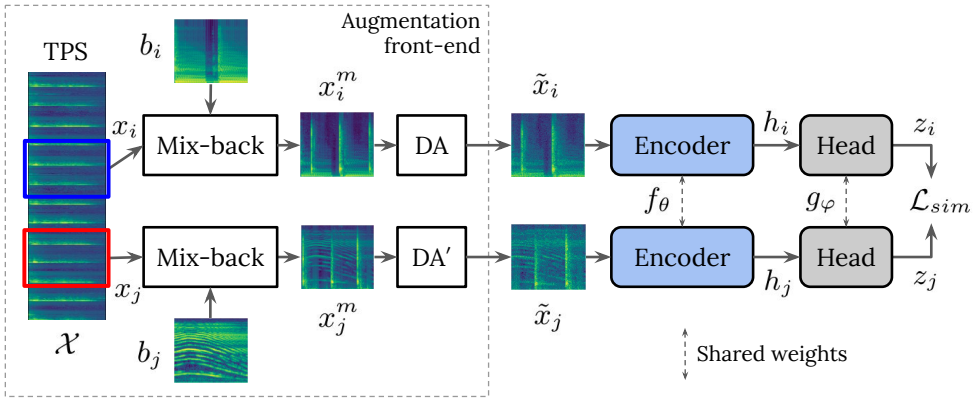

Figure 6.1: Framework for contrastive learning of audio representations based on similarity maximization. The framework is composed of an augmentation front-end, a common encoder f_θ, and a common projection head g_φ. Dashed lines between networks denote shared weights. The augmentation front-end is composed of temporal proximity sampling (TPS), mix-back and other data augmentations. Primes in the data augmentation (DA) blocks illustrate that each block is a different instance of the same augmentation policy. The Figure illustrates the creation of pairs of positive examples—the pairs of negatives are constructed from different clips.

where $\lambda \sim \mathcal{U}(0, \alpha)$, $\mathcal{U}$ is a uniform distribution, $\alpha \in [0, 1]$ is the mixing hyper-parameter (typically a small value), and $E(\cdot)$ denotes the energy of a given patch. A similar approach is used in Jansen et al. (2018) to create examples for triplet loss-based training. The energy adjustment of Equation 6.1 ensures that x_i is always dominant over b_i, even if $E(b_i) >> E(x_i)$, thereby preventing aggressive transformations that may make the proxy task too difficult. Before Equation 6.1, patches are transformed to linear scale (inversion of the log in the log-mel) to allow energy-wise compensation, after which mix-back is applied, and then the output, x_i^m, is transformed back to log scale. Background patches b are randomly drawn from the training set (excluding the input clip $\mathcal{X}$), hence they are out-of-batch in the vast majority of cases. As mentioned in Section 6.2, recent work shows that useful representations arise if data views share as little information as possible, while preserving relevant semantic information that keeps the predictive power for related downstream tasks (Tian et al., 2020). That is our motivation to use mix-back: *i)* shared information across positives is decreased by mixing x_i and x_j with different backgrounds, and *ii)* semantic information is preserved due to sound transparency (i.e., a mixture of two sound events inherits the classes of the constituents) and the fact that the positive patch is always predominant in the mixture. Mix-back can be understood as a data augmentation method, but we separate it from the others as it involves two input patches.

Other Stochastic Data Augmentation Methods. After temporal proximity sampling and mix-back, we finally adopt other data augmentation methods. We choose data augmentation methods that are directly computable over T-F patches (rather than waveforms). One of the main criteria is that the augmentations are simple and suitable for on-the-fly computation. Therefore, we prioritize speed rather than acoustical or mathematical correctness. We consider data augmentation methods both from the computer vision and audio literature:

- **Random resized cropping**. This is a common augmentation in the computer vision literature, used for example for ImageNet training (Russakovsky et al., 2015). It consists of making a random sized crop of the incoming patch and resizing it to the original patch dimensions. We follow the Inception-style random cropping (Szegedy et al., 2015), also adopted in Chen et al. (2020b) for visual representation learning. The size of the crop is distributed uniformly within a pre-specified range of the incoming patch size. While, a priori, this image augmentation might seem not appropriate for spectrograms, we are interested in knowing whether there is any benefit from the stretching effect that it might inflict over the time and frequency dimensions.

- **Random time/frequency shift**. In the time dimension, this is a time delay. The hyperparameter n_f determines the maximum number of delay frames. The actual delay is drawn from a uniform distribution $\mathcal{U}(1, n_f)$. In the frequency dimension, we shift the input patch by a number of mel bands upwards in frequency. The highest bands that are shifted beyond the patch dimensions are placed in the lowest bands of the new transformed patch. The hyperparameter n_b determines the maximum number of shifting bands. The actual shift is drawn from a uniform distribution $\mathcal{U}(1, n_b)$.

- **Compression**. This augmentation consists of a simple compression of the log-mel energies obtained by multiplying them with a scaling factor drawn from a uniform distribution $\mathcal{U}(\alpha_c, 1)$, where $\alpha_c \in (0, 1)$ is a hyperparameter.

- **specAugment**. specAugment (Park et al., 2019) is described in Section 2.3.4.2. In these experiments, we apply two frequency masks and two time masks. Time warping is not applied. The width of the frequency masks and time masks is drawn from uniform distributions, $\mathcal{U}(0, F_m)$ and $\mathcal{U}(0, T_m)$, where F_m and T_m are hyperparameters representing the maximum number of consecutive bands or time frames being masked by one mask, respectively.

- **Gaussian noise addition**. This augmentation consists of addition of Gaussian noise directly on the log-mel energies. The noise is given by a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with mean 0 and standard deviation σ. In turn, σ is drawn from a uniform distribution $\mathcal{U}(0, \alpha_g)$, where α_g is a hyperparameter.

Implementation for the above augmentation methods is available in the released code.[51] Each augmentation method has one or more hyperparameter(s) that must be manually tuned and that are kept constant for a given experiment. As seen above, each of these manually-tuned hyperparameter(s) determines a uniform distribution from which the actual parameter determining the specific augmentation is drawn. This process is done at the patch level, that is, for every patch a new parameter is drawn from each distribution, leading to a different instantiation of the augmentation method. In this way, the data augmentation operations specifically applied over each view of a given example are in fact two different instantiations of the same family of transformations. This is indicated by DA and DA′ in Figure 6.1.

The final augmentation policy adopted consists of sequentially applying random resized cropping, compression and Gaussian noise addition. Nonetheless, as we discuss in Section 6.3.3.3, other augmentation policies may also be valid and possibly lead to better performance. These augmentation methods transform x_i^m into the input patch $\tilde{x}_i$ for the encoder network.

6.3.1.2 Similarity Maximization

Encoder Network. We use a CNN-based encoder network f_θ with parameters θ to extract the embedding $h_i = f_\theta(\tilde{x}_i)$ from the previously augmented patch $\tilde{x}_i$, where h_i is the embedding right before the final fully-connected classification layer. Once the contrastive learning process is over and the encoder is trained, the representation h_i can be used for downstream tasks. In Section 6.3.3.4 we experiment with different encoder networks.

Similarity Projection Head. Following Chen et al. (2020b), a simple projection network g_φ with parameters φ maps h_i to the final L2-normalized low-dimensional representation z_i where the contrastive loss is applied. Our head consists of a Multi-Layer Perceptron (MLP) with one hidden layer, batch-normalization, and a ReLU non-linearity. Note that the projection head is only used during contrastive learning, i.e., once the training is over, only the trained encoder is used for downstream tasks. It must also be noted that the embeddings z_i and z_j are obtained with a single instantiation of the encoder network and the projection head (see *shared weights* in Figure 6.1).

Contrastive Loss. To compare a positive pair of examples, x_i and x_j, we adopt the normalized temperature-scaled cross-entropy (*NT-Xent*) loss given by (Le-Khac et al., 2020; Chen et al., 2020b):

$$\mathcal{L}_{sim_{i,j}} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{v=1}^{2N} \mathbb{1}_{v \neq i} \exp(\text{sim}(z_i, z_v)/\tau)}, \tag{6.2}$$

where z_i and z_j are the metric embeddings corresponding to the patches x_i and x_j, $\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v}/\|\boldsymbol{u}\|\|\boldsymbol{v}\|$ represents cosine similarity whose sensitivity is adjusted by a temperature value $\tau \in (0, 1]$, $\mathbb{1}_{v \neq i} \in \{0, 1\}$ is an indicator function that returns 1 when $v \neq i$, and N is the batch size. Since two views are generated from each incoming audio clip, the batch size is extended from N to $2N$ elements. This allows for one pair of positive examples and $2(2N-2)$ pairs of negative examples for every input clip—the overall loss includes both $\mathcal{L}_{sim_{i,j}}$ and $\mathcal{L}_{sim_{j,i}}$. By minimizing the objective in Equation 6.2 during training, parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ are adjusted to maximize the numerator (i.e., the agreement between embeddings of positives, assigning them to neighboring representations) while simultaneously minimizing the denominator (i.e., the similarity between embeddings of negatives, forcing them to distant spots in the embedding space).

The COLA approach utilizes bilinear similarity instead of cosine similarity as scoring function in Equation 6.2, reporting performance boosts (Saeed et al., 2021).

6.3.2 Experimental Setup

6.3.2.1 Dataset

The experiments conducted with our proposed framework are based on the FSDnoisy18k dataset presented in Section 5.2, whose main characteristics are outlined next. FSDnoisy18k contains 42.5h of Freesound audio distributed across 20 classes drawn from the AudioSet Ontology (Gemmeke et al., 2017). The dataset includes a small *clean* training set (1772 clips / 2.4h), a larger *noisy* train set (15,813 clips / 38.8h), and a test set (947 clips/ 1.4h). Labels in the clean and test sets are manually-labelled whereas labels in the noisy set are inferred automatically from metadata, hence featuring real-world label noise. The dataset is singly- and weakly-labeled, and clips are of variable-length in range [0.3, 30]s.

We decided to avoid large-scale datasets, e.g., AudioSet (Gemmeke et al., 2017), due to the computationally intensive contrastive learning experiments that we set out to conduct. Utilizing AudioSet (over 5000h of audio) would have been intractable under our compute resources at the time. FSDnoisy18k

is much smaller than AudioSet while it has a much more reduced vocabulary (20 classes vs 527 in AudioSet). As mentioned in Section 5.2.3, the number of clips per class in the noisy set of FSDnoisy18k ranges from 250 to 1000. This represents a relatively large amount of per-class training data when compared to other popular related datasets (Salamon et al., 2014; Piczak, 2015b; Fonseca et al., 2018b; Cartwright et al., 2019c), making FSDnoisy18k a reasonable candidate for this kind of study. The main shortcoming is that FSDnoisy18k's limited vocabulary implies that the learned representations are not transferable to unrelated datasets or downstream tasks. Therefore, we conduct an *in-domain* evaluation, that is, we evaluate the learned representations using also FSDnoisy18k. Specifically, FSDnoisy18k allows evaluation of learned representations on two real-world scenarios: using a small clean set for training, and a larger noisy set for training.

The FSD50K dataset proposed in Chapter 3 would have been a natural candidate for this study. Its vocabulary of 200 classes seems reasonably large to be able to learn general-purpose audio representations that can then be transferred to other downstream tasks or datasets, as done in a recent work by Niizumi et al. (2021). Unfortunately, FSD50K was not finished by the time these experiments started.

6.3.2.2 Learning Pipeline

Our learning methodology consists of two stages. First, we carry out unsupervised contrastive learning of a low-dimensional representation. Then, we conduct the evaluation of the learned audio representation using supervised tasks. In both stages, incoming audio is transformed to 96-band log-mel spectrograms, and to deal with variable-length clips we use T-F patches of 1s (equivalent to 101 frames of 30ms with 10ms hop). Shorter clips are replicated; longer clips are trimmed in several patches. Thus, the input to all models is of shape TxF=101x96, as in the pre-processing of FSD50K (Section 3.4.1). We use three CNN architectures in our study. On the one hand, we use a ResNet-18 (He et al., 2016a) as this network is widely used among visual representation learning works (Chen et al., 2020b,c). On the other hand, we also adopt a VGG-like and a CRNN very similar to those evaluated with FSD50K in Section 3.4.1. As mentioned in that Section, these networks are commonly-used for SER tasks. Models are always trained using SGD with momentum 0.9 and weight decay 10^{-4}, using a batch size of 128 and shuffling examples between epochs.

For the **unsupervised contrastive learning experiments** (covered in Section 6.3.3), we follow the approach described in Section 6.3.1. In this stage, we always train on the noisy set and validate on the clean set of FSDnoisy18k. Models are trained for 500 epochs, with initial learning rate of 0.03, divided

by 10 in epochs 325 and 425.

For the **supervised learning experiments** assessing the learned representations (covered in Section 6.3.4), we adopt two downstream classification tasks: one training on the noisy set and validating on the clean set of FSDnoisy18k, and another training on the clean set, after separating 15% for validation purposes (see Section 6.3.4.2). Models are trained for 200 epochs to minimize categorical cross-entropy, reducing the learning rate in epochs 80 and 160. The initial learning rate is 0.1 when training from scratch, and 0.01 in case of initializing with unsupervised pre-trained weights in order to constrain the learning process.

6.3.2.3 Evaluation Methodology

To quantify the quality of the learned audio representations, we follow three approaches: *i)* a variation of the standard ***k*-Nearest Neighbour (kNN) evaluation** in Wu et al. (2018), *ii)* the widely used **linear evaluation** protocol (Oord et al., 2018; Chen et al., 2020b), and *iii)* **fine-tuning a model end-to-end**.

kNN Evaluation. For the kNN evaluation, we estimate the representation z for each validation patch and compare it against every other patch in the given set via cosine similarity. The prediction for every patch is, then, obtained by majority voting across the k neighbouring labels, where $k = 200$ as in Wu et al. (2018). In turn, clip-level predictions are obtained by majority voting of patch-level ones. This evaluation is used for fast contrastive learning experimentation as no additional training is required.

Unlike kNN evaluation, linear evaluation protocol and end-to-end fine-tuning involve further training and passing patches through an entire model to produce prediction probabilities.

Linear Evaluation Protocol. The linear evaluation protocol involves training an additional linear classifier on top of the pre-trained unsupervised embedding on a given downstream task. In other words, this means training an additional linear classifier on top of the frozen base network. In this setting, we use test accuracy as a proxy for the quality of the learned representation, as in Chen et al. (2020b).

End-to-end Fine-tuning. This evaluation approach consists of fine-tuning a model on a given downstream task after initializing it with the pre-trained

contrastive weights. Unlike in the linear evaluation protocol, here *all* the weights in the network are updated on the downstream task, hence the term *end-to-end*.

In linear evaluation and end-to-end fine-tuning, patch-level predictions are averaged per-class across all patches in a clip to obtain clip-level predictions. Common to the three evaluation methods in this Section, once clip-level predictions are gathered for a given set, overall accuracies are computed on a per-class basis, then averaged with equal weight across all classes to yield the performance reported in Sections 6.3.3 and 6.3.4. For linear evaluation and end-to-end fine-tuning, we report test accuracy provided by the best validation accuracy model. However, learning curves for the contrastive learning experiments using kNN evaluation were found to be relatively noisy, such that *best* accuracy is not always representative of the overall quality of the training process. Consequently, we decided to use the *average* validation accuracy across the last 50 epochs, as top performing model checkpoints appear at the end of the training. Finally, results by the three above evaluation methods are compared against supervised baselines, where metrics are computed following an identical procedure to that of end-to-end fine-tuning.

6.3.3 Ablation Study

In this Section, we report on ablation experiments carried out in order to study the main components of our proposed method of Figure 6.1. Results reported in this Section are always using **kNN validation accuracy** as a result of the kNN evaluation method described in the previous Section. In the analysis of each component, we always start from the best configuration found, then evaluate other alternatives.

6.3.3.1 Temporal Proximity Sampling

Table 6.1 shows the results for different implementations of the temporal proximity sampling, that is, different ways of sampling T-F patches within the incoming audio clip. In particular, we compare *i)* sampling patches at random within the clip, and *ii)* deterministic sampling of patches in which they are separated by a sampling distance of d time frames (where each frame corresponds to a time shift of 10ms, as per the hop size adopted when framing the audio signal). It can be seen that stochastically sampling patches provides superior performance. We study the impact of progressively increasing d between a

Table 6.1: kNN validation accuracy for several mechanisms of temporal proximity sampling. d is the distance between patches, in number of time frames. Each frame corresponds to a time shift of 10ms, given by the hop size adopted when framing audio.

Sampling Method	**kNN**
Sampling at random	**70.1**
$d = 0$ (same T-F patch)	51.1
$d = 25$	61.5
$d = 75$	65.1
$d = 125$	67.9
$d = 200$	69.9
$d = 300$	68.5
$d = 400$	69.7

first randomly sampled patch x_i and a second patch x_j.[53] The overall trend observed is that the higher the distance, the better the representation learned. As the patch length is 101 frames, when $d < 101$ both sampled patches overlap (middle section of Table 6.1). Results show that the cases featuring overlapping patches ($d < 101$) underperform no overlapping ones ($d > 101$, bottom section of Table 6.1), where performance saturates. This observed behaviour aligns to some extent with recent results in computer vision, where increasing the distance between image crops is beneficial only up to some values, after which performance decreases due to little semantic content shared between the views (Tian et al., 2020).

6.3.3.2 Mix-back

Table 6.2 shows results for mix-back for best mixing hyperparameter $\alpha \in \{0.02, 0.05, 0.1, 0.2\}$. It can be seen that using mix-back helps considerably, and adjusting the energy is also beneficial. The latter means that the foreground patch is always dominant over the background patch, thus preventing potentially aggressive transformations. The optimal values for the mixing hyperparameter α are small, which indicates that a light mixture is preferred (see Section 6.3.1.1 for details). Interestingly, we observe that lightly adding backgrounds from other patches (i.e., mix-back, Table 6.2) is more beneficial than adding artificial Gaussian noise (see first and second rows from Table 6.3). These results suggest that mixing with natural backgrounds from real audio signals is suitable for contrastive audio representation learning.

[53] The distance d between patches might be bounded due to clip length, which lies in the range [0.3, 30] seconds.

Table 6.2: kNN validation accuracy for several mix-back settings. $\alpha \in [0, 1]$ is the mixing hyperparameter.

Mix-back Setting (α)	kNN
Mix-back w/ E adjustment (0.05)	**70.1**
Mix-back w/o E adjustment (0.02)	66.2
w/o mix-back	63.3

6.3.3.3 Other Stochastic Data Augmentation Methods

Table 6.3 lists the main results for several data augmentation policies. Each row represents the best result after non-exhaustively sweeping the corresponding data augmentation hyperparameters. We started by exploring data augmentation methods individually (from bottom to top in Table 6.3). The top data augmentation applied individually is random resized cropping. Through visual inspection of a number of T-F patches, we found out that the optimal random resized cropping applies a mild cropping (instead of a severe one), which can be seen as a small stretch in time and frequency that also involves a small frequency transposition. In particular, the original Inception-style implementation of random resized cropping considers crops whose size is distributed uniformly between 8% and 100% of the incoming image size (Szegedy et al., 2015). Here, after a non-exhaustive sweep of the crop size, the best validation kNN accuracies were found when applying a much less aggressive cropping, specifically, using crop sizes distributed uniformly between 80% and 100% of the incoming T-F patch size. This is the cropping configuration in which the cropped T-F patches most resemble the original patch, among the configurations considered. This result suggests that more severe croppings (i.e., smaller, such as those applied with images in computer vision) provide transformations that are too aggressive and thereby unsuitable for our task. This random resized cropping slightly outperforms specAugment (Park et al., 2019), which has been successfully used for contrastive learning of speech embeddings (Nandan & Vepa, 2020), and which also works well for sound events.

Then, we explored compositions of data augmentation methods based on the mentioned random resized cropping. We found out that, to a lesser extent, compression and Gaussian noise addition also improve the learned representation. We adopt this data augmentation policy (top row of Table 6.3) for all experiments reported in Sections 6.3.3 and 6.3.4. The configuration of best hyperparameters after non-exhaustive tuning consists of: crop sizes distributed uniformly between 80% and 100% of the incoming T-F patch size, compression with $\alpha_c = 0.75$, and Gaussian noise addition with $\alpha_g = 0.01$.

Table 6.3: kNN validation accuracy for several data augmentation (DA) settings. RRC = random resized cropping.

DA Policy	kNN
RRC + compression + Gaussian noise	**70.1**
RRC + compression	69.6
RRC + specAugment	70.0
RRC	69.0
specAugment (Park et al., 2019)	68.0
w/o DA	60.1

However, a subsequent more thorough exploration of the augmentation methods revealed promising results by composing random resized cropping and specAugment, yielding almost top results (70.0). It is conceivable that a more exhaustive (and costly) exploration of the data augmentation compositions may lead to better results, e.g., complementing random resized cropping and specAugment with other softer augmentations such as compression or Gaussian noise addition. Nonetheless, this should be done carefully. For example, in our experiments we have seen that the ordering of the augmentations matter, and joining individually-tuned augmentations can be suboptimal as different augmentations in a composition affect each other.

6.3.3.4 Encoder and Temperature

For encoder architectures, we explore ResNet-18, VGG-like and CRNN, obtaining kNN validation accuracies of 70.1, 67.7 and 67.1, respectively. These results may suggest that networks with higher capacity (ResNet-18 in our case) are better suited for contrastive audio representation learning, which accords with previous insight in the visual domain (Chen et al., 2020b). We will see more results supporting this hypothesis in Section 6.3.4. We also experiment with the temperature τ of the NT-Xent loss function (Equation 6.2), which has been found to be a relevant parameter in Chen et al. (2020c). In particular we sweep $\tau \in \{0.1, 0.2, 0.3, 0.4\}$, obtaining the kNN validation accuracies of 68.9, 70.1, 68.9 and 67, respectively. Results show the framework's sensitivity to τ, which we also find to be dependent on the projection head configuration.

6.3.3.5 Discussion

In general, we observe that the proposed framework of Figure 6.1 is sensitive to hyperparameter changes, and that the various settings of each component affect each other, thus requiring extensive experimentation for appropriate

tuning. At the same time, computation runtimes for unsupervised contrastive pre-training experiments are longer than those for supervised classification experiments. This is due to *i)* the higher number of training epochs used, which we found useful in accordance with Chen et al. (2020b), and *ii)* a more computationally expensive learning algorithm, which includes pairwise comparisons of embeddings after the forward pass as well as several data augmentation methods. As a consequence, the results reported here are the outcome of a non-exhaustive exploration of the components in the data augmentation front-end of Section 6.3.1.1. It is possible that a more in-depth exploration of the augmentation methods and their interaction leads to further performance boosts.

From Sections 6.3.3.1 and 6.3.3.2 we observe that *i)* sampling *overlapped* T-F patches when drawing positive examples is detrimental, and *ii)* lightly mixing the positive examples with natural backgrounds is beneficial. This could indicate that the original positive examples sometimes share time-frequency patterns that can be used to lower the NT-Xent loss function of Equation 6.2, but that hinder the learning of useful audio representations. These undesired patterns are denoted as *shortcuts* in the computer vision literature (Minderer et al., 2020). Examples of shortcuts in self-supervised audio representation learning may include recording equipment, room acoustics or background noise. FSDnoisy18k is based on Freesound audio, which in turn is composed of audio contributed by uploaders. As we discussed in Sections 3.2.6 and 3.4.2, clips coming from the same uploader are likely to share some of these patterns, which has been shown to have an impact in supervised sound event tagging. We conjecture that this could be a source of shortcuts in our setting, which is being mitigated by the stochastic sampling of positive T-F patches and mix-back. This interesting topic warrants further investigation.

Finally, negative examples are drawn from clips within the current batch at every iteration, as done in Jansen et al. (2020); Chen et al. (2020b). Some previous works report advantages of using larger batch sizes as a way to provide more negative examples, which facilitates training convergence. For example, batch sizes up to 8k are used in Chen et al. (2020b). Our compute resources do not support such large batch sizes, hence all our experiments are conducted with a batch size of 128, commonly used for supervised learning experiments. Based on insights from Chen et al. (2020b); Wang & van den Oord (2020), it is conceivable that using larger batches yields better results than those reported here.

Table 6.4: Test accuracy for linear evaluation protocol (second column), and for two downstream sound event classification tasks: a larger noisy set and a small clean set for training. *This column also corresponds to the supervised baselines to be compared with the linear evaluation. p-t = pre-trained.

Model	**Linear**	**Larger noisy set**		**Small clean set**	
(weights in M)	-	random*	p-t	random	p-t
ResNet-18 (11)	**74.3**	65.4	**78.2**	56.5	**77.9**
VGG-like (0.3)	70.0	70.6	**72.8**	61.1	**72.3**
CRNN (1)	64.4	72.0	**74.2**	58.7	**69.1**

6.3.4 Evaluation of Learned Representations

6.3.4.1 Baseline Systems and Linear Evaluation

Table 6.4 presents the test accuracies for the linear evaluation protocol and the corresponding supervised baselines (second and third columns, respectively). In the third column, we observe that the supervised CRNN and VGG-like perform similarly (72.0 and 70.6), while ResNet-18 performs worse (65.4). This trend accords with the results obtained for the baseline systems of FSD50K in Section 3.4.1 and Table 3.6. In the current setting, this could be due to the capacity of ResNet-18 (the largest by far), which may lead to overfitting of the smaller dataset (and the noisy labels to some degree). In linear evaluation of the contrastive weights, however, ResNet-18 is the top performing system (74.3). This finding accords with the kNN evaluation of Section 6.3.3.4 and with findings in Chen et al. (2020b). Thus, with ResNet-18 the supervised baseline is exceeded by a considerable margin, whereas with VGG-like and CRNN most of the supervised performance is recovered (99% and 89%, respectively).

6.3.4.2 In-Domain Downstream Prediction Tasks

As mentioned in Section 6.3.2.1, we conduct an in-domain evaluation of the learned audio representations, similar to that of Cartwright et al. (2019a) in the context of urban sounds. In particular, we utilize the two downstream sound event classification tasks posed by FSDnoisy18k: *i)* training on the larger set of noisy labels (as done always so far for unsupervised contrastive learning), and *ii)* training on the small set of clean data (in this case, keeping 15% of the clean set for validation purposes). For each task, we compare a supervised baseline trained from scratch, with fine-tuning the network initialized with unsupervised pre-trained weights. These correspond respectively to the columns

random and *p-t* (pre-trained) in Table 6.4. This set of experiments aims at measuring the benefits of unsupervised contrastive pre-training with respect to training from scratch in noisy- and small-data regimes.

Table 6.4 shows that unsupervised contrastive pre-training brings great benefits, achieving better results than training from scratch in both tasks and across all network architectures considered. For both tasks, using the ResNet-18 architecture as encoder yields top accuracy in the pre-trained setup, and the lowest accuracy when trained from scratch. This suggests that the performance attainable with ResNet-18 supervised from scratch is limited, potentially by limited data and/or label quality. In contrast, unsupervised contrastive pre-training seems to alleviate these problems, leveraging ResNet-18's capacity and yielding superior performance. Greater pre-trained vs. random improvements are observed in the "smaller clean" task. Also, interestingly, the pre-trained performance in the "smaller clean" task shows little degradation with respect to that of the "larger noisy" task, despite having far fewer training examples. As established in Section 6.3.2.1, the noisy set has 15,813 clips whereas the clean set has only 1772 clips—a proportion of almost 9:1. Specifically, for ResNet-18, the pre-trained performance decreases from 78.2 to 77.9, whereas training from scratch yields a substantial accuracy drop (from 65.4 to 56.5). A possible explanation for the similar pre-trained performance across downstream tasks may be that, in the "smaller clean" task, the pre-trained model is fine-tuned with unseen clean data (albeit a small amount). However, in the "larger noisy" task, the model is fine-tuned with the *same* data previously used for unsupervised contrastive learning, and the supervision that is now available for fine-tuning is affected by label noise. Yet, even when using the same data, unsupervised contrastive pre-training still outperforms training from sractch.

6.4 Self-Supervised Representation Learning from Automatically Separated Sound Scenes

In the previous Section, we have seen that the simultaneous use of a diversity of augmentation methods is critical to the success of the similarity maximization task, which is consistent with the findings reported in Chen et al. (2020b). For audio modeling, commonly-adopted augmentation strategies include mainly those explored in the previous Section, which have also been used by recent works: temporal proximity sampling (Jansen et al., 2018, 2020; Saeed et al., 2021; Wang & van den Oord, 2020), artificial example mixing (Jansen et al., 2018; Wang & van den Oord, 2020), time/frequency masking (Nandan & Vepa, 2020; Wang & van den Oord, 2020), time/frequency shifts (Jansen et al., 2018; Wang & van den Oord, 2020), Gaussian noise addition (Jansen et al., 2018) and random resized cropping. In most cases, these augmentations introduce arti-

ficial, handcrafted transformations with hyperparameters that must be tuned to lie within a semantics-preserving range. However, typical sound scene recordings already tend to be quite complex, involving mixtures of several sound sources at varying levels and unexpected channel distortions. Therefore, these artificial augmentation techniques risk introducing an unrealistic domain shift that can hinder generalization to real-world applications.

Real-world sound scenes consist of time-varying collections of sound sources, each generating characteristic sound events that are mixed together in audio recordings. The association of these constituent sound events with their mixture and each other is semantically constrained: the sound scene contains the union of source classes and not all classes co-occur naturally. With this motivation, in this Section we explore the use of unsupervised automatic sound separation to decompose unlabeled sound scenes into multiple semantically-linked views for use in self-supervised contrastive learning. This provides a sort of inverse to traditional example mixing augmentation: instead of constructing artificial mixtures, we decompose a sound scene into a collection of simpler channels that share semantic aspects with the original recording and each other. In contrast to the previous approaches, this automatic separation approach is data-driven and input-dependent, producing ecologically valid views that eliminate the need for parameter tuning for the given dataset.

We find that learning to associate sound mixtures with their constituent separated channels elicits semantic structure in the learned representation and yields stronger representations than past approaches that use the mixtures alone. We also show that learning to associate pairs of mixtures and their separated channels is complementary to some of the commonly-used data augmentations from the previous Section. Furthermore, we pair this augmentation procedure with a multitask objective that includes the proxy tasks of similarity maximization and coincidence prediction, which exhibit complementary behavior for different downstream representation use cases. Finally, we discover that a wide range of separation model competencies enable useful (and complementary) augmentations, suggesting that optimal sound separation performance is not essential for representation learning. The best representation learned with our sound-separation informed framework achieves an mAP of 0.326 on the downstream shallow-model AudioSet classification task. This exceeds previous results on this benchmark by a large margin under the same evaluation protocol (Jansen et al., 2018, 2020), and is on par with the state-of-the-art under comparable evaluation settings (Wang & van den Oord, 2020).

6.4.1 Sound Separation as Data Augmentation

Sound separation has been studied as preprocessing to improve supervised sound event detection (Turpault et al., 2020b, 2021). Here, we propose sound separation as an *augmentation* to generate pairs of positive examples for contrastive learning. In Section 6.2, we highlighted two observations: First, suitable views for contrastive learning are those such that their mutual information is reduced while the semantically-relevant information between them is retained (Tian et al., 2020). Second, the proxy learning task attempts to ignore the transformations applied to create the views, that is, to learn representations invariant to such transformations.

We propose to decompose an incoming audio clip, which in general is a *mixture* of multiple sound events, into its constituent sources. We then use the mixture and these separated channels to form positive pairs for contrastive learning. In particular, the comparison of the input mixture and one of the separated channels should meet the requirements established above: *i)* the mutual information is reduced as, in principle, there is at least one input sound source that is no longer in the separated channel; *ii)* some relevant semantics are preserved as the sound source(s) present in the separated channel is also present in the input mixture. Therefore, in theory, this comparison would be well suited for contrastive learning. Further, with this comparison we are promoting the learning of representations that are invariant to combinations of naturally co-occurring or overlapping sources—a valuable property for general-purpose audio recognition applications. We use this *mixture vs channel* comparison as the default contrastive setup for the majority of our experiments, as illustrated in the proposed learning framework depicted in Figure 6.2. By contrast, the comparison between two separated channels would not be appropriate for the similarity maximization task as, in principle, each channel would contain different sources, thereby violating the semantic preservation requirement. However, this *channel vs channel* comparison could still be useful for the coincidence prediction task, where the semantic equality demand is relaxed to require only some consistency between the examples in order to support their coincidence prediction. We evaluate experimentally these hypotheses in Section 6.4.4, uncovering several nuances.

6.4.1.1 MixIT for Unsupervised Sound Separation

For a sound separation system we use a model trained with Mixture Invariant Training (MixIT) (Wisdom et al., 2020). MixIT is an unsupervised method in which training examples are constructed by mixing existing audio clips, and the model is tasked to separate the resulting mixtures into a number of latent sources, such that an optimal remix of the separated sources best ap-

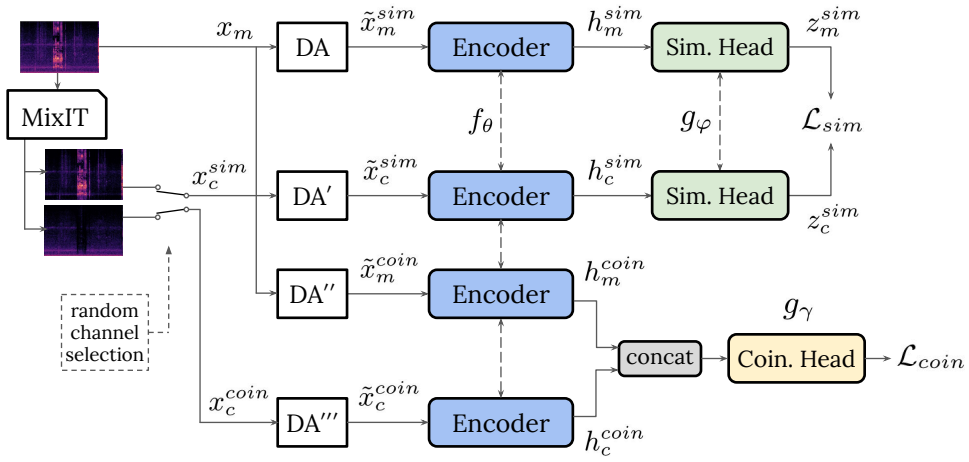

Figure 6.2: Sound-separation informed framework for contrastive learning of audio representations. It is composed of an unsupervised sound separation and augmentation front-end, a common encoder f_θ, and two task-specific heads, g_φ and g_γ, for the similarity maximization and coincidence prediction tasks respectively. Dashed lines between networks denote shared weights. Each separated channel feeding each proxy task (x_c^{sim} for similarity maximization or x_c^{coin} for coincidence prediction) is selected randomly between the two output channels from the MixIT separation model. The concat block stacks the latent representations for each view to define the input to the coincidence prediction head. Primes in the data augmentation (DA) blocks illustrate that each block is a different instance of the same augmentation policy, combining Temporal Proximity and SpecAugment. Note that the front-end illustrates the creation of pairs of positive examples—the pairs of negatives are constructed from different clips.

proximates the original mixtures. The main advantage of MixIT compared to previous methods is that it does not require knowledge of ground truth source signals, which allows leveraging large amounts of unlabeled data. In addition, MixIT has shown great promise in the task of *universal sound separation*, that is, separating arbitrary sounds instead of specializing to, for example, speech (Kavalerov et al., 2019b; Wisdom et al., 2021).

Our separation models were trained on AudioSet (Gemmeke et al., 2017) while ignoring all available labels and using previously proposed training settings (Wisdom et al., 2020). All MixIT separation models were provided by Scott Wisdom of Google Research. The separation model architecture used is based on an improved time-domain convolutional network (TDCN++) (Kavalerov et al., 2019b), which is similar to a Conv-TasNet (Luo & Mesgarani, 2019). This model consists first of an encoder that maps short frames of the input waveform to a latent space. Then, separation is done in the latent space where M masks are predicted for the target sources. Finally, M separated waveforms

are reconstructed through a decoder from the masked features. Separated sources are constrained to add up to the input mixture via a consistency projection layer (Wisdom et al., 2019). One of our goals is to assess the benefits of MixIT separation pre-processing coupled with a contrastive learning back-end.

6.4.1.2 Composition of Augmentations

As discussed in Section 6.2, composing several augmentations is important to obtain effective representations. This has been shown for vision tasks (Chen et al., 2020b), and it is also suggested by the results of Section 6.3.3. By composing multiple augmentations, we define a more challenging learning task so that higher-quality representations can emerge.

Here, in order to construct a more challenging proxy task, we combine sound separation with temporal proximity sampling and SpecAugment (Park et al., 2019) (in this order). Temporal proximity sampling was introduced in Section 6.2, and was proven effective in the experiments of Section 6.3.3.1. Here, it consists of randomly selecting two audio snippets of 0.96s as basic units to construct pairs of examples, instead of leveraging entire AudioSet clips of typically 10s.

Then, we apply SpecAugment on the log-mel spectrograms of the selected audio snippets, with time warping and time/frequency masking (Park et al., 2019). SpecAugment has gained popularity as data augmentation in supervised classification, and it has also been used to generate views for contrastive learning of speech (Nandan & Vepa, 2020). By using it, we also observed competitive results in the experiments of Section 6.3.3.3.

The combination of temporal proximity sampling and SpecAugment is represented by the data augmentation (DA) blocks in Figure 6.2. Together with the preceding unsupervised sound separation stage, they form the front-end for the two proxy tasks. In Section 6.4.4.3 we further extend these compositions including different convergence states of the separation model which provide distinct transformations to the incoming audio.

6.4.2 Proxy Learning Tasks

This Section describes the two proxy tasks used in our framework: a similarity maximization task, and a coincidence prediction task. For both, a pair of *positive* examples is constructed by selecting two audio snippets within the *same* 10s AudioSet clip, either from the input mixture or from the resulting separated channels. Analogously, a pair of *negative* examples is constructed by selecting two snippets from *different* clips (mixtures or separated channels). This is based on the assumption that two snippets within a given temporal

proximity are much more likely to be semantically related than two snippets from independent recordings.

6.4.2.1 Similarity Maximization

As introduced in 6.3.1, the similarity maximization task consists of maximizing the agreement between differently-augmented views of the same audio example. This loss attempts to co-locate the two representations in the same spot of the embedding space, thus promoting invariance to the transformations applied to generate the views. Its block diagram is depicted in the top half of Figure 6.2.

Front-end. The pipeline starts with one input mixture x_m (i.e., one AudioSet clip). Using the separation model, every incoming mixture x_m is separated into two output channels, from which one is randomly selected for this proxy task, x_c^{sim}. Next, x_m and x_c^{sim} undergo temporal proximity sampling and SpecAugment transformations (see Section 6.4.1.2). Note that each example x_m or x_c^{sim} undergoes a different instance of the same transformation policy (indicated by DA and DA$'$ in Figure 6.2).

Encoder Network. The outputs from the DA blocks, $\tilde{x}_m^{sim}$ and $\tilde{x}_c^{sim}$, feed a convolutional encoder f_θ in order to extract low-dimensional embeddings h. Specifically, for the top branch we obtain $h_m^{sim} = f_\theta\left(\tilde{x}_m^{sim}\right)$, where h_m^{sim} is the representation after a d-dimensional embedding layer and θ are the encoder's parameters. Once the training is over and the encoder has converged, the representation h is evaluated on downstream tasks.

Similarity Projection Head. We use a simple MLP, g_φ with parameters φ, to map the representation h to the final metric embedding z, the domain in which the contrastive loss is applied. This head is used to allow the representation h to back away from the representation at the training objective. Previous work reports better downstream performance using h instead of z (Chen et al., 2020b), something we also observed in preliminary experiments.

Contrastive Loss. To compare a positive pair of examples, x_m and x_c^{sim}, we adopt the NT-Xent loss (Le-Khac et al., 2020; Chen et al., 2020b) introduced in Equation 6.2, which we also include next for convenience:

$$\mathcal{L}_{sim_{i,j}} = -\log \frac{\exp\left(\text{sim}(z_i, z_j)/\tau\right)}{\sum_{v=1}^{2N} \mathbb{1}_{v \neq i} \exp\left(\text{sim}(z_i, z_v)/\tau\right)}, \tag{6.3}$$

where z_i and z_j are the metric embeddings corresponding to x_m and x_c^{sim}, $\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$ represents cosine similarity whose sensitivity is adjusted by a temperature value $\tau \in (0, 1]$, $\mathbb{1}_{v \neq i} \in \{0, 1\}$ is a function that returns 1 when $v \neq i$, and N is the batch size. Since two views are generated from each incoming audio clip, the batch size is extended from N to $2N$ elements, allowing for one pair of positive examples and $2(2N-2)$ pairs of negative examples for every input mixture—the overall loss includes both $\mathcal{L}_{sim_{i,j}}$ and $\mathcal{L}_{sim_{j,i}}$.

6.4.2.2 Coincidence Prediction

The coincidence prediction task relies on the *slowness prior* in representation learning (Wiskott & Sejnowski, 2002). Audio waveforms of sound sources can vary quickly, whereas the corresponding perceived semantics typically change at a much slower rate. Consequently, there should be a relatively stable latent representation in order to explain the semantic perception of the sound events. This representation would ideally support the prediction of whether a pair of examples are coinciding within a given temporal proximity. This task is a generalization of the correspondence prediction task proposed for audio-visual multimodal learning (Arandjelovic & Zisserman, 2017), where the task is to predict time *correspondence* between audio and video frames. Here, we relax the time scale requirement to predict *coincidence* within a prescribed temporal proximity, specifically within the (maximum) 10s of AudioSet clips. The task diagram is depicted in the bottom half of Figure 6.2.

Front-end and Encoder Network. In Jansen et al. (2020), this proxy task has been applied directly to audio snippets drawn from the same/different temporal proximity. Here, we adopt the augmentation front-end described in Section 6.4.1, which is the same as for the similarity maximization task. The processing carried out by the augmentation front-end yields the augmented examples $\tilde{x}_m^{coin}$ and $\tilde{x}_c^{coin}$ from x_m and x_c^{coin}. Then, we use a convolutional encoder to extract d-dimensional embedding representations h. Note that the same encoder network is shared across both proxy tasks, f_θ, as can be seen in Figure 6.2.

Coincidence Projection Head. Once the embedding representations for a pair of examples are obtained, we use a coincidence network, g_γ with parameters γ, tasked to predict the (non)-coincidence between the pair. More specifically, we feed g_γ with the concatenation of the two embeddings $[h_m^{coin}, h_c^{coin}] \in \mathbb{R}^{2d}$. The coincidence head consists of an MLP with one output unit, mapping the concatenated embedding representation to the probability that the input pair is coinciding—a binary classification task.

Loss. In a generic batch of N pairs of within-clip coinciding examples (i.e., positive pairs), $X = \left\{(x_1^i, x_2^i)\right\}_{i=1}^{N}$, we define $N-1$ pairs of negative examples per each pair of positives. This is done by pairing the non-coinciding examples (x_1^i, x_2^j) for $i \neq j$. In this setting, for a given batch X and focusing on our goal of optimizing the representation h, the coincidence loss function follows the class-balanced binary cross entropy expression (Jansen et al., 2020):

$$\begin{aligned}\mathcal{L}_{coin}(X) = &-\frac{1}{N}\sum_{i=1}^{N}\log g_\gamma([h_m^{coin,i}, h_c^{coin,i}]) \\ &-\frac{1}{N(N-1)}\sum_{\substack{1\leq i,j\leq N \\ j\neq i}}\log\left[1 - g_\gamma([h_m^{coin,i}, h_c^{coin,j}])\right].\end{aligned} \tag{6.4}$$

6.4.2.3 Joint Optimization

We conjecture that jointly optimizing the two objectives above in a multi-task setting can favor learning complementary information for semantic representation learning. Both proxy tasks share the ultimate goal of contrastive learning, that is, supporting relationships between pairs of positives and pairs of negatives so as to force a semantically structured embedding space. However, each task pursues this goal in a slightly different way, in terms of underlying principle and implementation.

Underlying Principle. The similarity maximization task essentially aims to co-locate the representations of both examples in a positive pair at the same point in the embedding space. Therefore, for successful representation learning, it is usually required that some semantic relationship is preserved between the two examples, e.g., the two examples share some class label(s). By contrast, the coincidence prediction task is based on a weaker condition. Instead of co-locating representations, the goal is to assign a representation that supports coincidence prediction, establishing a clear relationship between the representations for both examples, but not necessarily requiring their co-location.

Implementation The NT-Xent loss of Equation 6.3 follows a canonical version of contrastive loss, explicitly measuring similarity of embeddings as the scoring function (Le-Khac et al., 2020). By contrast, the binary cross entropy loss of Equation 6.4 is not a contrastive loss per se, but rather a loss typically used for classification, fed with probabilities. In this case, one could argue that the coincidence head serves as a learned similarity measure between two points in the embedding space, conceptually analogous to the handcrafted scor-

ing functions typically present in the canonical contrastive losses (e.g., cosine similarity in the NT-Xent loss).

6.4.3 Experimental Setup

In Section 6.3.2 we discussed how our limited computational resources influenced the choice of dataset and some design choices for the experiments with the framework of Figure 6.1. In contrast, the experiments described in the current and following Sections with the framework of Figure 6.2 were conducted during an internship at Google Research. This allowed leveraging larger amounts of data and computational resources.

6.4.3.1 Evaluation Methodology

By optimizing the training objectives of Equations 6.3 and 6.4, the goal is to learn semantically discriminative audio representations. We train our framework using a superset of the AudioSet training set consisting of around 3M audio clips, while ignoring all available labels. To evaluate the learned representation h, we use the trained encoder f_{θ} as a feature extractor following the next two evaluation methodologies. We adopt these evaluation methodologies to allow direct comparison with past work by Jansen et al. (2018, 2020).

Query by Example (QbE) Retrieval. Given a small subset of AudioSet with around 100 examples per class, cosine distance is computed between *i)* all the within-class target pairs, and *ii)* all the (present, not-present) pairs as non-target trials. Then, we sort the resulting distances in ascending order and compute per-class AP of ranking target over non-target trials. Averaging per-class AP leads to the reported *QbE* mAP. This is a direct measurement of the representation semantic consistency without requiring further training.

Downstream Classification with Shallow Model. This is a supervised classification task carried out by training and evaluating a shallow architecture on top of the fixed embeddings previously learned. In particular, we use an MLP with one 512-unit hidden layer and ReLU activation, followed by a 527-way classification layer with sigmoid activation. For this purpose, we use the entire AudioSet training set version and report *classification* mAP. This measures the usefulness of the learned representation for a large-vocabulary downstream supervised classification task. This evaluation method is conceptually analogous to the linear evaluation protocol of Section 6.3.2.3. Here, however, we use an MLP instead of the linear classifier used in linear evaluation.

For every experiment, we train our framework until QbE convergence, which typically occurs between 400k and 600k steps, after which QbE mAP plateaus. We select an encoder checkpoint from this plateau and report the QbE mAP. Then, we use this checkpoint to extract features for the entire AudioSet and conduct the shallow classifier evaluation. After L2-normalizing the embeddings, we train on the AudioSet training set, allowing 5% for validation where we optimize mAP, then report classification mAP on the evaluation set.

6.4.3.2 Implementation and Training Details

Sound Separation. A critical parameter in the proposed framework is the number of output waveforms in the separation model, M, which must be defined at train time. Upon inspection of a few AudioSet clips selected randomly, we realize that many clips contain one or two dominant sources (i.e., in the foreground, lasting long within the clip), sometimes accompanied by additional sources (either in the foreground but very short, e.g., impact sounds, or in the background).[54] We therefore ran preliminary experiments with $M = \{2,4\}$ and saw that results using $M = 4$ were slightly worse for both proxy tasks. We attribute this to the fact that when $M = 4$ it is not uncommon to find near-empty output channels, filled with mild background noise, or with sound sources active only in a very short period of time. We hypothesize that using these channels to create positive pairs can be problematic. To confirm our hypothesis, we designed simple heuristics (based on energy and cosine similarity) to detect these near-empty channels, in order to allow discarding the "worst" channel in every contrastive setup, thus keeping only the other 3 channels from where to pool positive examples. This led to a small but consistent performance improvement, confirming our initial hypothesis, yet still underperforming results with $M = 2$. While further optimizations to allow using $M = 4$ could be pursued, for simplicity we decided to adopt $M = 2$ in our experiments, which is the minimal separation possible. In some cases, the two output waveforms coming out of the separation model will contain one source each, whereas in other cases they will contain several sources each. Consequently, we use the term separated *channels* (and not *sources*) as it is deemed more appropriate. We believe $M = 2$ is sufficient to evaluate our hypothesis of sound separation serving as a valid transformation for view generation in contrastive learning. Note that sound separation is only used during the learning of the representation—in our downstream tasks no separation is applied.

Temporal Proximity Sampling and SpecAugment. The DA blocks in Figure 6.2 consist first of temporal proximity sampling, i.e., random selection of

[54] The main exception to this rule is music segments.

0.96 s waveform snippets within the (maximum) 10s AudioSet clips. Snippets are transformed to log-mel spectrogram patches using a 25ms Hann window with 10ms hop, and 64 log-mel energy bands, leading to T-F patches of $T \times F = 96 \times 64$. SpecAugment is then applied using *i)* two frequency masks and two time masks, with a max width of 10 bands or frames, respectively; and *ii)* time warping with 8 frames as maximum warp (Park et al., 2019).

Networks. For the encoder we use a convolutional network based on CNN14 from Kong et al. (2020a). Our modifications from the original CNN14 include removing Batch Normalization (Ioffe & Szegedy, 2015) and Dropout (Srivastava et al., 2014), which was not found to be beneficial in our experiments. In addition, we substitute the classifier layer and the preceding fully-connected layer by an embedding convolutional layer with d filters, followed by a global max pooling operation to produce the d-dimensional representation h, which is used for downstream tasks. We use $d = 128$ unless stated otherwise. The resulting encoder network has 76M weights. The similarity head consists of an MLP with one hidden layer of 256 units and ReLU non-linearity, followed by an output layer with 128 units, which is the dimension for the metric embeddings z feeding the NT-Xent loss. The coincidence head consists of an MLP with two hidden layers of 512 units and ReLU nonlinearities, followed by an output layer with one single unit to produce coincidence predictions feeding the class-balanced binary cross entropy loss.

Training Details. Experiments are carried out considering each proxy task individually as well as the full framework trained jointly. When both tasks are trained jointly, the two objectives are optimized from scratch and equally weighted obtaining a joint loss $\mathcal{L}_{joint} = \mathcal{L}_{sim} + \mathcal{L}_{coin}$, using one optimizer to update all the networks. We use the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 1e-4 whenever the coincidence prediction task is involved, or 3e-4 when only the similarity maximization task is considered. The temperature parameter in Equation 6.3 is set to $\tau = 0.3$. Learning rates and τ are tuned by optimizing QbE mAP on a validation set different from that used to report results.

Batch Size. The framework is trained on Google Cloud TPUs of 32 cores with a global batch size of 2048, which means local batches of 64 examples per core. Loss contributions and gradients are computed locally in each replica, then aggregated across replicas before applying the gradient update. Contrastive learning approaches typically benefit from comparison with multiple negative examples. In our framework, negative examples are drawn from clips within the current batch at every iteration, as done in Section 6.3.3 and also in

previous works (Jansen et al., 2020; Chen et al., 2020b; Wang & van den Oord, 2020). This approach is more practical than relying on a memory bank (Wu et al., 2018), a memory queue (He et al., 2020), or negative mining techniques to find suitable negatives (Jansen et al., 2018). However, with this simple approach the quality and diversity of negatives are limited by the batch size (in our case, the local batch size). Recent works show how increasing batch sizes provide solid improvements in visual (Chen et al., 2020b) and audio (Wang & van den Oord, 2020) contrastive representation learning. For example, Wang & van den Oord (2020) utilize batch sizes of up to 32k examples. Here, we do not explore this avenue and evaluate our proposed approach using the aforementioned more usual batch size. Based on previous literature (Le-Khac et al., 2020; Chen et al., 2020b; Wang & van den Oord, 2020), if our approach shows promise using the batch size selected for our experiments, it is expected to provide better performance under more favorable conditions given by larger batches.

6.4.4 Experiments

This Section describes the experiments run using the framework of Figure 6.2, or portions of it. For simplicity, in the following Tables the similarity maximization task and the coincidence prediction task are sometimes referred to as *SimCLR* and *CP*, respectively.

6.4.4.1 Baseline Experiments

Table 6.5 lists the performance when sound separation is ablated from the front-end in Figure 6.2, which is equivalent to all DA blocks being fed by the input mixture x_m.

Table 6.5: mAP without sound separation in the front-end (i.e., using only the input mixture). SA = SpecAugment, TP = Temporal Proximity, CP = Coincidence Prediction.

Representation	QbE mAP	Classification mAP
Log-Mel Spectrogram (baseline)	0.423	0.065
simCLR & SA	0.551	0.196
simCLR & TP	0.591	0.248
simCLR & TP & SA	**0.613**	0.265
CP & TP & SA	0.599	**0.286**

We use log-mel spectrogram as a baseline handcrafted representation. As expected, both SpecAugment and temporal proximity with the similarity maximization task as back-end substantially outperform the naive log-mel spectrogram. The effectiveness of temporal proximity is noteworthy considering its simplicity. Initially proposed in Jansen et al. (2018), it has been widely adopted in contrastive learning works (Jansen et al., 2020; Wang & van den Oord, 2020; Saeed et al., 2021), some of which use it as the sole augmentation (Jansen et al., 2020; Saeed et al., 2021). Note that the approach simCLR & TP is conceptually comparable to the recent COLA (Saeed et al., 2021). Combining temporal proximity and SpecAugment outperforms either one alone, thus validating the composition in the DA blocks of the front-end. Finally, results indicate different tendencies for the two proxy tasks, with the similarity maximization providing better QbE mAP, and the coincidence prediction attaining better classification mAP.

6.4.4.2 Sound Separation for Contrastive Representation Learning

We now report the experiments including the unsupervised sound separation block in the front-end, as depicted in Figure 6.2. We assess various comparisons enabled by sound separation preprocessing, namely: *i)* comparing the input mixture with one of the separated channels (*mix vs chan*); *ii)* comparing the two separated channels (*chan vs chan*); or *iii)* comparing the input mixture with anything else, i.e., either with the input mixture or with one of the separated channels (*mix vs any*). Table 6.6 shows the performance with the similarity maximization (*SimCLR*) task as back-end.

Table 6.6: mAP using sound separation (SSep) in the front-end and the *SimCLR* back-end. Temporal proximity sampling is always applied; SpecAugment (SA) is applied as specified.

Comparison	SSep	SA	QbE mAP	Classification mAP
Mix vs mix (baseline)	-	-	0.591	0.248
Mix vs mix (baseline)	-	✓	0.613	0.265
Mix vs chan	✓	-	0.631	0.272
Mix vs chan	✓	✓	**0.640**	**0.282**
Mix vs any	✓	✓	0.638	0.279
Chan vs chan	✓	✓	0.611	0.254

By looking at the first rows of Table 6.6, we can benchmark SpecAugment and sound separation. We see that sound separation preprocessing (third row) provides a bigger boost in both metrics compared with SpecAugment (second

row), yet the best performance is obtained from their composition (fourth row). This trend for the *mix vs chan* comparison also holds for the other contrastive setups. Results indicate that comparing the input mixture with the separated channels provides substantially better representations than the baseline approach of leveraging only the input mixture. This confirms the usefulness of sound separation preprocessing for contrastive learning of audio representations. Allowing the input mixture to be compared with itself in addition to the separated channels (*mix vs any*) does not lead to performance boosts. Generally, the performance of *mix vs any* and *mix vs chan* were very similar across the experiments we ran. Hence, we adopt *mix vs chan* as best setup in order to focus on the effect of sound separation.

Finally, comparing both separated channels (*chan vs chan*) performs significantly worse, on par with the non-separated baseline (for QbE mAP) or even worse (for classification mAP). If we assume the separation has successfully isolated independent sources in each output, this comparison violates the semantic preservation principle (thus hindering the learning of semantic representations), so we might have expected decrements even larger than the ≈0.03 mAP with respect to *mix vs chan* for both metrics. After inspection of a few dozen separation examples, we identify two potential explanations for this observation. First, the result of the separation algorithm is not always perfect. This depends on the complexity of the input mixture—this is to be expected considering the great diversity of AudioSet clips. When this happens, the same source can be present in both separated channels. Second, even when the separation is satisfactory, there are some classes that retain a semantic relationship, e.g., two different instruments from the same family, or two different vocalizations from the same or similar animals. When used as a pair of positives, their relationship may still provide a useful learning supervisory signal compared to pairs of unrelated negative examples.

Table 6.7 shows the performance with the coincidence prediction task as back-end. Similar to the *SimCLR* back-end (Table 6.6), we again observe that

Table 6.7: mAP using sound separation in the front-end and the coincidence prediction back-end. Temporal proximity sampling and SpecAugment are applied.

Comparison	QbE mAP	Classification mAP
Mix vs mix (baseline)	0.599	0.286
Mix vs chan	**0.619**	**0.293**
Chan vs chan	0.590	0.283

the *mix vs chan* comparison yields top performance, outperforming the no-separation baseline. Comparing both separated channels (*chan vs chan*) again

leads to the worst results, in this case underperforming the baseline for QbE mAP, while being on par in terms of classification mAP. In addition to corroborating the utility of sound separation in the front-end, these results also show that coincidence prediction benefits from composing augmentations, which was not explored in the previous work of Jansen et al. (2020), where only temporal proximity is used.

Comparing performance across both proxy tasks, we notice that *SimCLR* always yields the best QbE mAP while coincidence prediction produces top classification mAP. This could be due to a better alignment between *SimCLR*'s underlying principle (maximizing/minimizing the cosine similarity between positives/negatives) and the QbE retrieval mAP (computed by ranking pairwise cosine distances). Finally, regarding the performance using the *chan vs chan* comparison, coincidence prediction shows substantially better classification mAP than similarity maximization (specifically, a mAP on par with the latter's best case). This accords with our intuition that coincidence prediction is tolerant of semantic differences between positives due to its weaker assumptions. However, for QbE mAP, the opposite behaviour is observed, presumably because this tolerance does not help the QbE objective.

6.4.4.3 Separation Processing at Different Convergence States

In Section 6.4.4.2 we show that sound separation is beneficial for our tasks, even when the separation is less than perfect as can occur when input mixtures are difficult to separate. This leads us to ask whether the processing provided by a separation model *before* convergence can also be a valid form of augmentation for contrastive representation learning. To answer this, we experiment with separation examples generated by multiple training checkpoints of a single separation network. We view the separation checkpoints as *audio processors* that implement complex modifications on the incoming audio. A qualitative assessment of output streams as learning progresses indicates four types of processors corresponding to four convergence states, and we empirically characterize their behavior as follows.[55] Figure 6.3 shows example spectrograms of the separated channels for each of these processors given the same input mixture.

1. **Separation after full convergence (S2, 1.7M steps)**. This is the separation model used for experiments in Section 6.4.4.2.
2. **Separation before convergence (S1, 5k steps)**. Separation performance is more limited.

[55]Note that the description of every *processor* is approximate and somewhat dependent on the input's complexity. For example the S1 model could provide a good separation when fed with an easy mixture.

3. **Filtering with early training model (F, 500 steps)**. Outputs are produced by the separation model very early during training. After ∼500 steps, sources are not separated, and the output channels are differently filtered versions of the input. Most sources in the input are present in all outputs, but often with different levels/spectral content, such as different spectro-temporal modulations.

4. **Noise with untrained model (N, 0 step)**. Outputs are produced by the separation model untrained. They feature a clearly audible, wide-band structured noise, correlated with the input signal. Audio artifacts are sometimes present. Both output channels are very similar.

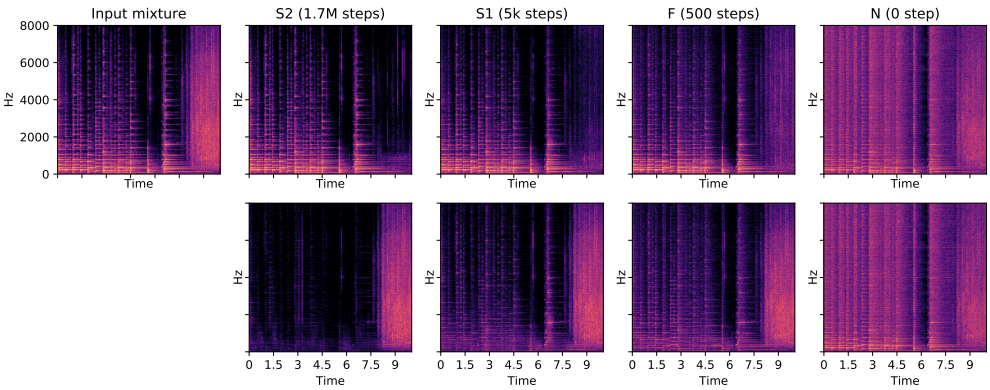

Figure 6.3: Spectrograms of the two separated channels obtained with four checkpoints (S2, S1, F, N) of the same separation model, given one input mixture (top left). The input mixture contains a guitar melody (up to ≈8s) followed by applause. For illustration purposes, this is a simple case where the separation is purely temporal (i.e., sources do not overlap). The general case features overlapping sources.

In Table 6.8 we show the results of substituting S2 with the other identified processors, while keeping the rest of the framework as in Figure 6.2. By looking at the top left section of Table 6.8, two observations can be made. First, the four processors all provide valid forms of augmentation to generate positive views for contrastive learning. While sound separation is beneficial (S2), a poorer separation is also valuable (S1) and the earlier checkpoints of the separation network (which do not actually provide separation) are also useful (F and N). Second, an untrained, quasi-random TDCN++ provides structured noise that surprisingly yields the best single-checkpoint performance (N).

Following the common practice of composing augmentations to achieve more powerful representations (Chen et al., 2020b), we investigate combining the processors. The bottom left section of Table 6.8 shows the best results obtained when combining processors using the *OR* rule, that is, applying only

Table 6.8: mAP using different checkpoints of the separation model as learning progresses (top), as well as some combinations (bottom). As back-end, the *SimCLR* task is used (left), as well as the two proxy tasks trained jointly (right). Temporal proximity sampling and SpecAugment are applied. Comparison is always *mix vs chan*. CP = Coincidence Prediction.

Models	SimCLR		SimCLR & CP	
	QbE	Classification	QbE	Classification
S2 (1.7M)	0.640	0.282	0.649	0.289
S1 (5k)	0.639	0.283	0.651	0.293
F (500)	0.651	0.280	0.659	0.297
N (0)	**0.659**	**0.286**	**0.663**	**0.301**
S2 ∨ F	0.653	0.283	0.658	0.300
S2 ∨ N	0.660	**0.297**	0.671	0.306
S2 ∨ F ∨ N	**0.667**	0.285	**0.672**	**0.310**

one randomly selected processor at a time. It can be seen that sound separation and the quasi-random TDCN++ noise turn out to be complementary augmentations, resulting in a more beneficial composition. Adding the F processor seems to provide lift for QbE mAP, but not for classification mAP. Applying two processors in cascade to every example does not improve performance.

6.4.4.4 Joint Learning Framework

Lastly, the right side of Table 6.8 lists the results when training the entire framework of Figure 6.2, jointly optimizing both proxy tasks. We observe trends similar to using the *SimCLR* back-end alone (left side of Table 6.8), but with increased performance. When compared to the coincidence prediction back-end alone (i.e., S2 in Table 6.8 vs second row of Table 6.7), QbE mAP is improved by a large margin whereas classification mAP is on par. Overall, while the boost from jointly optimizing both tasks is sometimes not very large, it is consistent across almost all cases considered, both for individual processors as well as their combinations. We also note that the changes needed in the framework to accommodate a second task are minimal—only an additional MLP head and corresponding loss function—and the training setup carries no modifications—both tasks are trained jointly from scratch using one optimizer. Adopting a curriculum learning instead could enhance performance, as done in Jansen et al. (2020).

Results suggest that the key ingredient is not the quality of the sound separation process, but rather the combination of diverse processing provided by the

separation model as its learning progresses. While training a separation model requires a certain effort, once it is done several non-parametric augmentation generators become available, facilitating the generation of useful positive examples. While we choose a MixIT-based TDCN++, in principle, any source separation methodology could be used (and there may be additional benefit to using supervised systems).

6.4.4.5 Comparison with Previous Work

In Table 6.9 we compare our best setup with previous work on the downstream supervised classification task defined in Section 6.4.3.1. Works are grouped by ascending embedding dimensionality, d.

Table 6.9: Comparison with previous work using the downstream supervised classification with shallow model on AudioSet. mAP reported is classification mAP. MM = Multimodal approach.

Method	d	**MM**	**mAP**
Unsupervised triplet (Jansen et al., 2018)	128	-	0.244
C^3 (Jansen et al., 2020)	128	✓	0.285
Separation-based framework (**ours**)	128	-	0.310
CPC (Wang et al., 2020)	512	-	0.277
Separation-based framework (**ours**)	1024	-	0.326
MMV (Alayrac et al., 2020)	2048	✓	0.309
Multi-format (Wang & van den Oord, 2020)	2048	-	**0.329**
L^3 (Arandjelovic & Zisserman, 2017)	6144	✓	0.249
AudioSet baseline (Gemmeke et al., 2017)	-	-	0.314
Supervised PANN (Kong et al., 2020a)	-	-	0.439
Supervised PSLA (Gong et al., 2021b)	-	-	0.474

Results are strictly comparable only in the top section as those works are the only ones using the same training data, evaluation protocol and downstream embedding dimensionality, $d = 128$. Note that C^3 is based on audio-video multimodality for representation learning (Jansen et al., 2020), while our proposed framework outperforms it using only audio.

We also compare our system with works that use somewhat different evaluation settings in terms of, e.g., training data, embedding dimensionality or shallow classifier type, thus hindering a fair comparison. For example, most previous works use larger d values, ranging from 512 to 6144—we expect performance to

improve to some extent as d increases (Kong et al., 2020a). This could be due to the fact that the embedding representation contains more information that can be leveraged by the shallow model in the downstream task. We confirm this by increasing our d from 128 to 1024, which yields an absolute increase of 0.016 mAP. Some of these works leverage multimodal data for training such as audio-video (L^3 (Arandjelovic & Zisserman, 2017)) or audio-video-text (MMV (Alayrac et al., 2020)), while reporting worse performance than our lower-d audio-only framework.

The current unsupervised state-of-the-art on this task is achieved by Wang & van den Oord (2020), who propose a contrastive learning setup that maximizes the agreement between raw audio and its spectral representation. Among several variants proposed by the authors, we select the one that is more comparable to our proposed framework (i.e., using only log-mel as input and one encoder). Our reported performance is on par with this approach (0.326 vs 0.329) despite it leveraging higher d (2048 vs our 1024) and a much larger batch size (32768 vs our 64), potentially having an impact on performance as discussed in Section 6.4.3.2. Better results are reported in Wang & van den Oord (2020) by combining two different encoders (one per audio format) and concatenating their output representations into a $d = 4096$ embedding.

Finally, for reference, we include the current supervised state-of-the-art on this task. PANN is based on data balancing and augmentation (Kong et al., 2020a), whereas PSLA makes use of a collection of training techniques to boost performance (ImageNet pretraining, data balancing and augmentation, label enhancement, weight averaging and model aggregation) (Gong et al., 2021b). Gemmeke et al. (2017) provide a baseline for AudioSet based on a shallow fully-connected neural network classifier for the 485 categories available at the time.

6.5 Summary and Conclusion

In this Section, we summarize the main results and takeaways of this Chapter and we discuss several aspects of self-supervised audio representation learning.

6.5.1 Similarity Maximization for Sound Event Representation Learning

In Section 6.3, we first presented a framework for unsupervised contrastive learning of sound event representations, based on maximizing the similarity between differently-augmented views of the same log-mel spectrogram. Via ablation experiments, we showed that the simultaneous use of a diversity of augmentation methods is important for the success of the proposed sys-

tem, which is consistent with the findings reported in Chen et al. (2020b) for visual representation learning. Specifically, appropriately tuning the composition of positive patch sampling, mix-back, and other augmentations leads to successful representation learning. The evaluation on in-domain sound event classification tasks with FSDnoisy18k suggests that unsupervised contrastive pre-training can mitigate the impact of labeled data scarcity, and increase robustness against noisy labels, as recently found in supervised image classification (Hendrycks et al., 2019a). Specifically, when training a linear classifier on top of the pre-trained embeddings, we recover most of the corresponding supervised performance, and even surpass it using a ResNet-18 architecture. In addition, we show that fine-tuning several models initialized with pre-trained weights outperforms corresponding supervised baselines trained from scratch at our scale of data.

In particular, the results attained with the ResNet-18 architecture are remarkable. In Section 6.3, for the downstream tasks of fine-tuning on a small amount of clean labels (2.4h), and on a somewhat larger amount of noisy labels (38.8h), training a ResNet-18 from scratch yields the lowest accuracy compared to other models (a CRNN and a VGG-like network). The number of weights of these models are 11M, 1M and 0.3M, respectively. The performance of ResNet-18 in these tasks can be limited potentially by data scarcity and/or label quality. Similarly, in Section 3.4.1 we saw that training a ResNet-18 from scratch on FSD50K (80.4h) performs substantially worse than using other models. The poor results of ResNet-18 trained from scratch with tens of hours contrast with the good results of doing so with AudioSet (over 5000h of training audio) (Kong et al., 2020a). While the aforementioned Freesound-based datasets are different from AudioSet in a number of aspects (e.g., see Section 3.3.5), the most objective difference is the data size. A possible explanation for these different results is that the ResNet-18 architecture needs a certain amount of regularization in order to show its superior performance. Using large amounts of training data (as in AudioSet) can provide sufficient regularization, but the aforementioned Freesound-based datasets may fall short for training ResNet-18 from scratch. Remarkably, fine-tuning a ResNet-18 after unsupervised contrastive pre-training in the two downstream tasks of Section 6.3 yields top accuracy among several models (and the best results in the experiments of that Section). These findings suggest that unsupervised contrastive pre-training allows to alleviate the aforementioned problems of overfitting or lack of regularization, and to take advantage of ResNet-18's higher capacity at our scale of data.

Further, results of Section 6.3 suggest that the proposed method is able to learn useful audio representations in a scenario with relatively limited resources in terms of data and compute (training with less than 40h of audio). This contrasts with the general trend of using AudioSet for unsupervised contrastive pre-training (Jansen et al., 2018, 2020; Tagliasacchi et al., 2020; Wang &

van den Oord, 2020; Saeed et al., 2021), which we also did in this Chapter (Fonseca et al., 2021b).

6.5.2 Self-Supervised Representation Learning from Automatically Separated Sound Scenes

In Section 6.4, we presented a sound separation-based contrastive learning framework for unsupervised audio representation learning. We showed that sound separation can be seen as a valid augmentation to generate positive views for contrastive learning, and that learning to associate sound mixtures with their constituent separated channels elicits semantic structure in the learned representation, outperforming comparable systems without separation. We demonstrated that sound separation can be successfully combined with other commonly-used augmentations, such as temporal proximity sampling and SpecAugment, in order to define more challenging proxy tasks. We discovered that the transformations provided by different checkpoints of the same separation model as learning progresses are valid (and sometimes complementary) forms of augmentation for generating positives. This finding suggests that optimal sound separation performance is not essential for representation learning. In addition, we showed the benefit of jointly training the proxy tasks of similarity maximization and coincidence prediction.

By appropriately combining several separation processors followed by the joint optimization of the two aforementioned proxy tasks, we obtain downstream AudioSet classification results competitive with the state-of-the-art in unsupervised representations under comparable evaluation settings, and outperforming several multimodal approaches.

It is important to reflect on the implications of this and other achievements in the unsupervised learning literature. Our best learned representation achieves an mAP of 0.326 on the downstream shallow-model AudioSet classification task. In particular, this consists of first unsupervised representation learning, followed by training a shallow MLP of ≈336k weights on the learned representation, now with AudioSet labels. At the other side, one of the top works on supervised AudioSet classification achieves an mAP of 0.439 by training a model of 81M weights from scratch, along with data balancing and augmentation (Kong et al., 2020a).[56] While the state-of-the-art unsupervised performance still lags behind the supervised one, the gap is being progressively reduced. For instance, at the beginning of 2020, the state-of-the-art unsupervised mAP on the downstream AudioSet task was 0.285 (Jansen et al., 2020). Also, having

[56] We use Kong et al. (2020a) as a state-of-the-art supervised reference because the better results of Gong et al. (2021b) are obtained via a large collection of training techniques, which can distort the comparison.

a downstream task with far fewer labeled data—a more common real-world case—would have a more severe impact on the supervised learning paradigm, further reducing the gap. Therefore, in our view, self-supervised representation learning is a promising direction able to bridge the current gap between unsupervised and supervised learning, and potentially minimize the reliance on costly annotation efforts.

6.5.3 Discussion

In general, we observe that contrastive learning experiments require longer runtimes than supervised classification experiments. Reasons include a greater number of training steps required for convergence as well as a more computationally expensive learning algorithm, which includes pairwise comparisons of embeddings after the forward pass and several data augmentation methods. For example, the handling of multiple channels in the separation front-end of Figure 6.2 was found particularly expensive. Longer training times slow down the pace of experimentation, which can prevent in-depth exploration of data augmentation methods and other components in the pipeline.

Overall, we have seen how unsupervised contrastive pre-training yields promising results in three downstream sound event classification tasks that pose different real-world challenges: *i)* training on a small amount of clean labels of FSDnoisy18k (2.4h), *ii)* training on a somewhat larger amount of noisy labels of FSDnoisy18k (38.8h), and *iii)* training on a very large amount of large-vocabulary labels using an internal version of AudioSet (over 6500h). We have obtained very positive results in the two first tasks, which may be regarded as *small-data regime* (Bornschein et al., 2020). Here, unsupervised contrastive pre-training has led to performance boosts that exceed supervised baselines trained from scratch. In the large-scale AudioSet task, we have obtained competitive results, outperforming a shallow fully-connected classifier trained from scratch, but still far from the supervised state-of-the-art. We hypothesize that the usefulness of unsupervised contrastive pre-training depends on the amount of available labeled data in the downstream supervised classification task. In the small-data regime, the effects of data scarcity make supervised baseline classifiers less powerful, hence unsupervised contrastive pre-training is more effective. On the contrary, in the large-data regime, supervised classifiers are more proficient, hence the impact of unsupervised contrastive pre-training becomes more limited.

The above discussion suggests two main usages of unsupervised contrastive pre-training for sound event classification. One usage consists of applying unsupervised contrastive pre-training as a warm-up for a supervised classifier. This means training first in unsupervised fashion on a given labeled dataset, then using the unsupervised pre-trained weights as network initialisation for

fine-tuning on the same data, now using the labels. In this case, the expected performance boost is likely to be higher in the small-data regime, as our results suggest.

The other usage is to conduct unsupervised contrastive pre-training on a large piece of data (in terms of size and coverage) in order to learn general-purpose audio representations, with the goal of transferring them to different supervised downstream tasks to improve classification. We did not follow this approach for several reasons. In Section 6.3, we used FSDnoisy18k for unsupervised training due to compute limitations. This dataset has a vocabulary limited to 20 classes, hence the learned representations are not suitable for transferring knowledge to other tasks. In Section 6.4, we opted for the downstream AudioSet classification task as *i)* it has been used in previous works with which we wanted to compare our proposed system, and *ii)* given the large vocabulary of the downstream task, it is a representative benchmark despite the fact that the audio data are the same as in the unsupervised pre-training. The alternative of using external datasets for evaluation of general-purpose representations is more popular in visual (Chen et al., 2020b) and speech (Kawakami et al., 2020) representation learning. For general-purpose audio representations, recent works have also proposed this type of evaluation—e.g., Saeed et al. (2021) with 9 datasets and Niizumi et al. (2021) with 6 datasets, two of which differ from those used in Saeed et al. (2021). This type of *out-of-domain* evaluation is suitable and desirable for audio representation learning. A necessary milestone is for the community to agree on a solid benchmark formed by a common list of representative sound classification datasets.

CHAPTER 7

Summary and Future Perspectives

7.1 Introduction

In this thesis, we have investigated several strands of dataset creation as well as supervised and unsupervised learning in order to train large-vocabulary sound event classifiers, creating new data resources and using them as well as other existing ones in novel and alternative ways. In particular, we have advanced the state-of-the-art of sound event classification by proposing new large-vocabulary datasets, novel CNN architectural improvements, and multiple algorithms for learning with noisy labels and from unlabeled data. The proposed learning algorithms contribute to the transition from traditional supervised learning using clean labels to other learning strategies less dependent on annotation effort such as supervised learning using noisy labels and self-supervised representation learning. For the sake of open research and reproducibility, we strived to make openly available all datasets created and almost all source code used for this dissertation.

We started with an introduction to sound event classification and the different learning paradigms that we discuss in this dissertation (Chapter 1). We continued by providing background on the research field of sound event recognition, describing a sound event classification pipeline based on supervised learning, and summarising the existing literature on datasets, deep networks, and learning approaches using noisy labels and unlabeled data (Chapter 2). Then, we introduced the FSD50K dataset, including a description of the creation process, a characterization of the dataset, and a set of experiments to provide baseline systems as well as further insights on the data (Chapter 3). We advanced the state-of-the-art of sound event classification by proposing novel architectural changes that increase shift invariance in a well-known CNN, making it more robust against input time/frequency shifts (Chapter 4). We next introduced the

FSDnoisy18k dataset to support the investigation of label noise in sound event classification, and explored efficient and network-agnostic approaches to mitigate the effect of label noise while training sound event classifiers (Chapter 5). Finally, we proposed multiple strategies to learn audio representations from unlabeled data using self-supervised contrastive learning based on compositions of data augmentation methods (Chapter 6). In addition, we have significantly contributed to enabling and fostering sound event classification research on the multiple learning paradigms considered. This was achieved not only by the release of multiple open datasets and source code repositories, but also through the co-organization of two DCASE Challenge Tasks in 2018 and 2019 (Appendix A).

In this final Chapter, we enumerate our main contributions from a global point of view (Section 7.2), summarize the main conclusions from the previous Chapters (Section 7.3), and discuss the impact of this thesis (Section 7.4). Finally, we end this dissertation with a discussion about future perspectives on the topic of sound event classification using different types of supervision (Section 7.5).

7.2 Summary of Contributions

This thesis contributes to the advancement of the state-of-the-art in sound event classification using different types of supervision. Through the generated data and code resources and the proposed learning methodologies gathered in multiple publications, as well as the co-organized DCASE Challenges, the author hopes that this thesis paves the way for future machine listening research. The main contributions of this thesis can be summarized as follows:[57]

7.2.1 Technical Contributions

- A comprehensive review of the field of sound event recognition discussing its main characteristics and challenges, and a review of the literature on datasets and convolutional neural networks for sound event recognition, as well as approaches for learning with noisy labels and self-supervised learning (Chapter 2).

- Development of FSD50K, which contains 51,197 Freesound audio clips, totalling 108.3h of audio manually labeled using 200 classes drawn from the AudioSet Ontology. To our knowledge, this is the largest fully-open

[57] Links to code repositories as well as dataset download websites, companion websites, and other relevant websites can be found in Appendix A, C and D. Code is available for all works listed in this Section, except for those conducted at Google Research.

dataset of human-labeled sound events, and the second largest after AudioSet. Further, the large-vocabulary, stable and exhaustively labeled evaluation set is unprecedented. We provide a detailed description of the FSD50K creation process tailored to the particularities of Freesound data, including challenges encountered and solutions adopted (Section 3.2). The dataset release includes metadata used during the creation process as well as Freesound metadata for the clips forming the dataset.

- A comprehensive characterization of FSD50K along with a discussion of limitations and key factors to allow its audio-informed usage. This characterization includes classification experiments to provide baseline systems as well as critical insight on the main factors to consider when splitting Freesound audio data for machine listening tasks (Sections 3.3 and 3.4). Code and pre-trained models are available (Appendix D).

- Novel architectural modifications to increase shift invariance in CNNs based on low-pass filtering and adaptive sampling of feature maps. These modifications make the network exhibit higher robustness to time/frequency shifts in the input spectrograms. By addressing this often-taken-for-granted issue in CNNs along with data augmentation, we achieve new state-of-the-art classification performance on FSD50K (Chapter 4). Code and pre-trained models are available (Appendix D).

- Development of FSDnoisy18k, which contains 18,532 Freesound audio clips, totalling 42.5h of audio distributed in 20 classes drawn from the AudioSet Ontology. FSDnoisy18k consists of a small amount of manually-labeled clean data, and a much larger amount of noisy data containing a substantial amount of real-world label noise. To our knowledge, this is the first dataset to specifically provide for the investigation of real label noise in sound event classification, including an empirical characterization of the noise and a CNN baseline system (Section 5.2). The companion site includes a detailed characterization of the dataset as well as code for baseline experiments.

- Exploration and development of a series of techniques and learning strategies to mitigate the effect of label noise during the training of sound event classifiers. These approaches include regularization techniques, noise-robust loss functions, and sample rejection strategies to identify and discard potential noisy labeled examples during the learning process. These model-agnostic approaches can be easily incorporated into existing deep learning pipelines, requiring minimal intervention and computational overhead (Sections 5.3 and 5.4). Code is available (Appendix D).

- Development of a model-agnostic method to address the problem of missing labels in large sound event datasets, using AudioSet as a use case. The method is based on a teacher-student framework with loss masking to first identify the most critical missing label candidates, and then ignore their contribution during the learning process (Section 5.6).

- Development of a novel self-supervised contrastive learning framework for learning audio representations from unlabeled data, based on the proxy task of similarity maximization. This task consists of maximizing the agreement between differently-augmented views of sound events (Section 6.3). Code and pre-trained models are available (Appendix D).

- Development of a novel self-supervised contrastive learning framework for learning audio representations from unlabeled data, based on the joint optimization of similarity maximization and coincidence prediction tasks. These tasks are optimized across example views constructed via compositions of unsupervised sound separation models and other augmentations. To our knowledge, this is the first time that automatic sound separation is used as augmentation to create views for sound event representation learning. The result is an unsupervised audio representation that rivals state-of-the-art alternatives on the shallow AudioSet classification benchmark (Section 6.4).

7.2.2 Other Academic Contributions

- Technical Program Co-Chair of DCASE Workshop 2021.

- Co-organizer of the DCASE Challenge Task 2 in 2018 and 2019, where the topic of label noise was included for the first time as a research problem in the community.

7.2.3 Publications

The research carried out in this thesis has been published in the form of several papers in top international conferences and journals. The content of Chapter 3 is currently under review in IEEE/ACM Transactions on Audio, Speech, and Language Processing (Fonseca et al., 2020a). Similarly, parts of the research presented in Chapter 4 are currently under review in IEEE Signal Processing Letters (Fonseca et al., 2021a). The FSDnoisy18k along with the evaluation of noise-robust loss functions presented in Chapter 5 has been published in ICASSP (Fonseca et al., 2019b). Also in Chapter 5, the parts of the research related with regularization techniques and sample rejection strategies have been published in WASPAA (Fonseca et al., 2019a), and those related with

addressing the problem of missing labels using a teacher-student framework have been published in IEEE Signal Processing Letters (Fonseca et al., 2020b). The first part of Chapter 6 focused on optimizing a similarity maximization task has been published in ICASSP (Fonseca et al., 2021c). The second part of Chapter 6 focused on a multitask objective and sound separation as augmentation has been published in WASPAA (Fonseca et al., 2021b). Finally, the content of Appendix A related to the organization of the DCASE Challenge Tasks has been published in the DCASE Workshop (Fonseca et al., 2018b, 2019c). It must be noted that this dissertation focuses only on a subset of all the papers published during the thesis period. The full list of the author's publications during this thesis, either as a first author (12 papers) or through various collaborations (7 papers), is provided in Appendix B.

7.3 Summary of Conclusions

At the end of each Chapter, we included a Section summarising the most relevant results and conclusions of the corresponding work. For easier consumption, here we summarize the main takeaways of each Chapter.

While creating FSD50K, we experienced how human labeling using a large vocabulary of everyday sounds is a laborious and complex task, as we discuss in Chapter 3. Special emphasis was put on the careful curation of the evaluation set content and labels, so that it can serve as a reliable evaluation benchmark, which is unpredecented given its large-vocabulary, stability, exhaustive labeling and size. We showed that it is important to acquire solid knowledge of the specifics of the source data—in our case, Freesound audio and metadata, and the AudioSet Ontology—and identify data challenges and limitations, so that the dataset creation process can be adapted to these particularities, and pitfalls in the creation can be avoided. Through classification experiments, we showed that smaller models with basic tuning and audio-informed design choices can outperform larger off-the-shelf computer vision architectures. Finally, we showed that within-class data contamination must be considered and minimized when splitting Freesound audio for machine learning tasks as it can have a considerable effect on the evaluation of sound event classifiers. In contrast, our results suggest that between-class contamination has a lesser impact.

In Chapter 4, we evaluated two pooling methods to improve shift invariance in CNNs using the FSD50K sound event classification task. Results show that the VGG variants evaluated indeed present a problem of only-partial shift invariance. Inserting the proposed pooling methods into these architectures makes the networks exhibit higher robustness to time/frequency shifts and small perturbations in the input spectrograms, and leads to recognition boosts. These

facts suggest that reinforcing shift invariance in the models evaluated is beneficial for sound event classification, which could also apply to other CNN architectures. The proposed architectural changes yield consistent recognition improvements with minimal additional computation, which makes them an appealing alternative to conventional pooling layers. Our best system incorporates these architectural changes and simple mixup augmentation in order to achieve a new state-of-the-art mAP on FSD50K.

Chapter 5 focused on training sound event classifiers in presence of noisy labels. We explored simple and efficient approaches that are agnostic to network architectures, hence they can be easily incorporated into existing learning pipelines, requiring minimal intervention and computational overhead. Our results suggest that rejecting the contribution of noisy samples during training can be more effective than approaches based on *accepting* the noisy labels and mitigating their effect, such as regularization methods and noise-robust loss functions. In particular, our top approach on FSDnoisy18k is a loss-based sample rejection method that uses the model being trained as an instance selector to prune the train set during the learning process. Further, we identified missing labels as a pathology in the labelling of AudioSet. By using a teacher-student framework with loss masking to identify and ignore the most critical potentially missing labels, we found out that most of the improvement comes from filtering out a tiny portion (<1%) of the most critical estimated missing labels. We also showed that the damage done by missing labels becomes higher as the train set gets smaller—however, even when training with massive amounts of audio, the impact of these labelling errors can still be observed.

Chapter 6 focused on learning audio representations from unlabeled data using self-supervised contrastive learning. We showed that the simultaneous use of a diversity of augmentation methods is important for successful audio representation learning. Our evaluation with FSDnoisy18k suggests that unsupervised contrastive pre-training can mitigate the impact of labeled data scarcity, and increase robustness against noisy labels at this scale of data. In addition, we showed that sound separation is a valid augmentation to generate positive views for contrastive learning, and that learning to associate sound mixtures with their constituent separated channels elicits semantic structure in the learned representation, outperforming comparable systems without separation. We discovered that the transformations provided by different checkpoints of the same separation model as learning progresses are valid (and sometimes complementary) forms of augmentation for generating positives. This finding suggests that optimal sound separation performance is not essential for representation learning. By appropriately combining several separation processors followed by the joint optimization of the proxy tasks of similarity maximization and coincidence prediction, we obtain downstream AudioSet classification results competitive with the state-of-the-art in unsupervised representations

under comparable evaluation settings, and outperforming several multimodal approaches.

7.3.1 Discussion

Throughout this dissertation, we have discussed the advantages and shortcomings of three learning methodologies: supervised learning using clean labels, supervised learning using noisy labels, and self-supervised learning from unlabeled data. The success of each of these methodologies, and how proficient the resulting sound classifiers are, depends on a number of factors. Using clean labels clearly provides the strongest supervision for training sound event classifiers. However, this costly supervision is typically constrained by annotation budget, which often limits the amount of exhaustively labeled data that can be gathered. The other two learning methodologies considered reduce the dependence on careful and costly manual annotation processes to some degree, making these methodologies more sustainable in terms of annotation effort. In fact, one of the main advantages of these methodologies is that collecting training data without careful annotations (or any annotations at all) is cheaper, thereby downplaying or sidestepping annotation budget limitations, and opening the door to using much larger amounts of training data.

Results in Chapter 5 suggest that the combination of larger amounts of noisy labeled data and techniques to mitigate the effect of label noise can compensate for the deficiencies of the supervision to some extent. In this way, this combination of more data with weaker supervision can become a substitute for smaller amounts of clean labels. However, the success of each methodology (using noisy or clean labels) will depend on factors such as the ratio of noisy labeled data vs. clean labeled data, as well as the type and amount of label noise. Another factor that must be considered is the source(s) for the clean and noisy labeled data. In a scenario where the noisy labeled data comes from a different source than the target data, issues of domain mismatch can occur—for example, in the case where noisy labeled data for training is gathered from some web audio repository not particularly related with the task under consideration. Consequently, a domain mismatch between training and evaluation data could occur, which may represent an obstacle for models' generalization.

Results in Chapter 6 show the promise of using self-supervised contrastive representation learning to learn unsupervised audio representations that can then be used for supervised downstream tasks, thus reducing the reliance on external supervision. In particular, we discussed how a classifier warmed up with unsupervised contrastive pre-training, or trained on unsupervised learned representations, can be competitive with, or even outperform, baseline classifiers trained from scratch under certain conditions. Nonetheless, it must be noted that the baseline supervised classifiers used for comparison are relatively

simple. Their performance could be improved by adding enhancements to the traditional supervised learning setup, for example by using data augmentation methods. Once again, these results and conclusions can vary depending on factors such as ratio of unlabeled pre-training data vs. labeled fine-tuning data, and also the diversity of the unlabeled pre-training data and potential domain mismatch issues.

The aforementioned critical factors may vary substantially among different use cases. Hence, they must be carefully considered when deciding what methodology to adopt, if multiple choices are feasible. Nonetheless, there are three conditions that are likely to apply for the majority of use cases:

- A certain amount of clean labels is needed, even when opting for learning with noisy labels or self-supervised learning, at least for evaluation purposes. In addition, when learning with noisy labels, it is also recommended to use clean labels for validation purposes—tuning and optimizing models on noisy validation sets can lead to misleading results, as mentioned in Sections 3.4.1 and 5.5. Likewise, for self-supervised representation learning, clean labels are required for the fine-tuning of unsupervised pre-trained models or shallow models trained on unsupervised representations.

- To achieve a given classification performance via learning with noisy labels or self-supervised learning, typically larger amounts of training data are likely to be needed, compared to the amount needed to reach such performance in a conventional supervised learning fashion. This is to compensate for the weaker supervision provided by the noisy labels or the proxy tasks in self-supervised learning. The critical non-trivial question of how much additional data is needed will depend on the aforementioned factors and on how powerful the learning algorithms are.

- The processing of larger amounts of training data will unavoidably require more computational resources. Thus, learning with noisy labels and self-supervised learning are typically associated with longer runtimes, especially the latter as self-supervised learning algorithms tend to be more computationally intensive, as discussed in Section 6.5.3. In our view, this can carry serious implications for the future of machine listening, which we further discuss in Section 7.5.2.

7.4 Impact of this Work

The contributions listed in the previous Section have already had an impact on the research community. To measure this research impact we utilize easy-to-access metrics, namely, citations count, code repositories *star* count, and number of dataset downloads at the time of writing.[58] These metrics are chosen based on their accessibility, transparency and wide adoption. However, we acknowledge that these metrics are not necessarily rigorous indicators of research impact, and should be understood only as rough indicators.

- The work carried out during this thesis has received 516 citations as per Google Scholar.
- The released code repositories have received 225 *stars*.
- The released datasets have been downloaded multiple times. In particular, the number of unique downloads as per Zenodo records for FSD50K and FSDnoisy18k are 6693 and 4079, respectively. In addition, the two datasets created for the DCASE Challenge Tasks, FSDKaggle2018 and FSDKaggle2019, have been downloaded 7427 and 2472 times from Zenodo, and 6189 and 5110 times from the Kaggle platform, respectively.

One of the outcomes of this thesis with highest impact is FSD50K, which has already enabled a series of research directions, from sound classification to sound separation or representation learning. In addition, it has already been used for multiple machine learning competitions and for the creation of smaller datasets to enable further research in various tasks. A detailed list of both current and potential applications can be found in Section 3.3.4. Further, the work of this thesis has contributed to the consolidation of the research avenues of learning with noisy labels and self-supervised representation learning in the context of everyday sounds. As discussed in Sections 2.6.2 and 2.7.3, research on these topics prior to this thesis was scarce. We hope this thesis' work serves as a stepping stone to the proliferation of further research in these relevant directions, as it has already begun to happen in the topic of learning with noisy labels, with multiple publications building upon the work of this thesis and the organized DCASE Challenge Tasks (see Section 2.6.2 for examples).

7.4.1 Broader Impact

The author hopes that the research carried out in this thesis also has an impact beyond the context of sound event classification, or even beyond machine

[58] Data from September 28th, 2021.

listening. Next, we list some works from other research fields that have built upon the work of this thesis. Audio data from FSD50K (Fonseca et al., 2020a) has already been used for source separation (Wisdom et al., 2020), speech enhancement (Tzinis et al., 2021), or COVID-19 detection from audio signals (Ponomarchuk et al., 2021), and can serve as stimuli for listening experiments in cognitive sciences. One of our work on learning with noisy labels (Fonseca et al., 2019b) has been used in the fields of speech emotion recognition (Zhong et al., 2020), text classification (Tayal et al., 2020), and time-series classification for electrical signals (Castellani et al., 2021). Our work on unsupervised contrastive learning (Fonseca et al., 2021c) has been used in a work dealing with contrastive learning of seismic signal representations collected with sensor networks (Meyer et al., 2021). Our work on acoustic scene classification (Fonseca et al., 2017a, 2018a), which has not been covered in this dissertation but was conducted at the beginning of this thesis, has been cited in works related to pathological voice detection (Narendra & Alku, 2020), biosignals-based anxiety detection (Petrescu et al., 2020), or agricultural water management (Fan et al., 2019).

Finally, another goal that we wanted to achieve is to improve the utility of Freesound for the research community. With the work of this thesis, especially the audio datasets and DCASE Challenge Tasks, the usage of Freesound audio for research has become more popular. This can be noticed in the increasing number of papers citing Freesound in the last years compared to the years prior to the beginning of this thesis,[59] and also in the increasing interest from both academia and industry in doing research with FSD50K.

7.5 Future Perspectives

In this thesis, we have conducted research on dataset creation and learning algorithms for sound classification using alternative types of supervision. All the topics covered are bound to be relevant for the future of sound event classification, as we witness a transition to larger amounts of training data with weaker supervision. In this Section, we first discuss a number of future directions for these research topics, that can be addressed by leveraging the findings and resources outcome of this thesis. Then, we identify challenges that, in our view, must be considered for the future of machine listening research.

[59] A list of papers that mention Freesound or use Freesound data for research every year can be found at http://labs.freesound.org/papers/.

7.5.1 Future Methodologies

A More Sustainable Dataset Creation Methodology. The creation of FSD50K included tasks that required a considerable human effort, such as exhaustively labeling the evaluation set in order to obtain a solid benchmark. Further, we tried to obtain a set of labels as reliable as possible that could serve as a favorable starting point for future dataset extensions. These potential extensions could rely on (semi-) automatic methods to scale up more efficiently at the expense of certain amount of label noise. Fortunately, nowadays there are a few available resources in terms of data and pre-trained models that can aid dataset creation approaches. For example, more data for FSD50K could be curated via semi-automatic methods by leveraging existing pre-trained models such as the models proposed in Chapter 4, the recently released YAMNet,[60] or the models in Kong et al. (2020a). A possible semi-automatic approach for dataset creation could rely on active learning, where the informativeness of unlabeled data is estimated, and the most informative examples can be selected for human annotation. In particular, an strategy that seems appropriate is *query-by-committee* active learning, where several models with different underlying principles are used to estimate example informativeness (Han et al., 2016). Audio clips for which the models disagree the most could be regarded as more informative; hence, these clips can be prioritized for human annotation. On the contrary, models' agreement will often mean that the audio clip can be automatically labeled. Hence, we can use the models as a supervision mechanism, producing machine-generated annotations that substitutes the analogous human rating. In this way, dataset creation moves away from the dependence on manual labour in favour of automatic means to optimize human labeling.

A Unified Learning Framework for Sound Event Classification. In this dissertation, we have analyzed the paradigms of learning with noisy labels and unsupervised learning separately. However, there is nothing stopping us from combining them in a single unified learning framework for sound event classification. A possible instantiation of such framework is as follows. An initial stage of unsupervised audio representation learning would serve two purposes. On the one hand, it would serve as a warm-up model initialization for subsequent fine tuning on labeled data. On the other hand, the structured embedding space learned could be used for label noise detection. This could be done in several ways. A simple approach consists of detecting inconsistent labels in clusters of similar representations in order to prune the train set before or during the supervised fine-tuning stage. In this way, the detection of noisy labels could be based on a learned audio representation, instead of—or in combination with—loss values or score values associated with training

[60] https://github.com/tensorflow/models/tree/master/research/audioset/yamnet

examples, as done in Sections 5.4 and 5.6, respectively. This approach could be useful when dealing with noisy labeled data. Further, it could also be useful when dealing with clean *weak* labels whenever the processing is done by segmenting audio clips in short T-F patches, as done in this thesis or in Hershey et al. (2017, 2021). In these cases, inheriting the clip-level labels for every patch creates false positive examples that could be detected with the proposed approach.

Multimodal Self-Supervised Audio Representation Learning. As the gap between unsupervised and supervised learning is reduced, self-supervised representation learning is a promising direction for the future of sound event classification, potentially able to minimize the reliance on costly annotation efforts. In this thesis, we have focused on audio-only representation learning, that is, learning from unlabeled *audio*. However, web repositories such as YouTube or Flickr are sources of *audiovisual* data. Using audio-video multimodality for audio representation learning opens the door to new possibilities based on the design of novel proxy learning tasks to exploit the *natural* supervision between both modalities. Recent works following this multimodal methodology report promising results outperforming audio-only approaches (Wang et al., 2021). The main downside, however, is that the computational requirements to process video signals can grow substantially.

7.5.2 Future Challenges

As mentioned earlier, the adoption of the paradigms of learning with noisy labels and self-supervised learning are often associated with larger amounts of training data to compensate for the weaker supervision. While these paradigms are more sustainable in terms of annotation effort, they are less sustainable in terms of computational requirements for the development of sound event classifiers.

As the amounts of training data keep growing, there is a point when not everyone may have enough computational resources to process such amounts of data in reasonable runtimes, thus restricting the access to these exciting research avenues. In particular, some of the experiments carried out in this thesis are a luxury that not everyone can afford. The author had the opportunity to intern at Google Research, with access to large amounts of data and computational resources. Thanks to that, experiments in Sections 5.6 and 6.4 using AudioSet were possible. However, doing this kind of experiments using university infrastructure would have slowed down experimentation substantially, especially for the experiments based on self-supervised learning. This would have translated into suboptimal analysis and characterization of the proposed methods. Thus, in our view, access to computational resources can influence

how representative and generalizable experimental results are, as well as the impact of the research work. This can lead to an unfortunate segregation of the research community according to the available computational resources, as not all research institutions will be able to afford the processing of certain amounts of data, or certain type of experiments. A possible measure to alleviate this problem consists of fostering the inclusion of a set of experiments using smaller portions of data, in addition to other experiments with full large-scale datasets. The ultimate goal is to lower the barrier for those who can only afford limited compute, allowing system development in a more affordable research track.

In addition, intensive experimentation with large amounts of data can pose a challenge from the environmental standpoint. To our knowledge, this is an unexplored area in sound event classification; however, recent work analyzes the environmental impact of developing speech recognizers by estimating the amount of CO2 emitted during the training process (Parcollet & Ravanelli, 2021). The authors find out that the carbon footprint of training speech recognizers is not negligible, although they depend on factors such as geographical location, type of GPU or hyperparameter tuning. Noticeably, it is shown that the evolution of CO2 emissions vs. performance exhibits an exponential behaviour, where the final tiny performance improvements tend to come at a very high carbon cost. This raises the question of whether being competitive with state-of-the-art performance justifies the extra carbon footprint. Solutions to this problem are yet unclear, but a first step is to raise awareness and stimulate the debate on the environmental implications of doing research with very large amounts of data.

Relatedly, in various passages of this dissertation we have made references to the state-of-the-art performance in several tasks. This performance is based on evaluation metrics computed over evaluation sets. An evaluation set aims to be a faithful and comprehensive sample of *reality*, that is, of the real-world challenges and particularities of a given task. However, this ideal state is difficult to reach. For example, in the context of sound event classification, it is reasonable to expect that not absolutely all of the textual labels in an evaluation set will be correct. Further, it is difficult to create an evaluation set with a selection of audio material and acoustic conditions that fully accounts for the diversity of sounds that we all encounter everyday. While sound event classification research is largely based on objective evaluation performance, another evaluation from the perspective of *user experience* may provide complementary insight. For example, it is unclear how much objective performance boost is needed for it to be noticed from a user experience perspective in a given task. Analysing these *just-noticeable differences* might be relevant for the selection of sound event classifiers for real-world applications. Similarly, it can also be relevant to know how much objective performance is necessary for a classifier to provide a satisfactory user experience in a particular task. A natural next

step of this thesis consists of conducting subjective evaluations of proposed sound event classifiers and audio representations in the context of Freesound. This would imply the deployment of the proposed technology in the Freesound infrastructure followed by a subjective evaluation with the help of users from the Freesound community. Certainly, this endeavour is far from trivial, for example in terms of design and implementation of the evaluation protocols. Yet, it may provide additional insight to that of a purely objective evaluation, and help to shape future machine listening research on sound event classifiers and other related topics.

APPENDIX A

DCASE Challenge Tasks on Learning with Noisy Labels

A.1 Introduction

The Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge is a scientific evaluation promoting environmental sound research and evaluation on common publicly available datasets. The DCASE Challenge has had almost annual editions since 2013, supporting the development of approaches for scene and event analysis, and tracking the performance of machine listening systems over time. The DCASE Challenge is structured in multiple *Challenge Tasks* including tasks such as as acoustic scene classification, sound event classification, detection and localization, audio captioning, anomaly detection, and more. Further information about the DCASE Challenge and its multiple editions can be found in the DCASE website.[61]

In order to foster open research on sound recognition, the author of this thesis along with other researchers organized two DCASE Challenge Tasks in 2018 and 2019.[62] In particular, the motivation behind these tasks is *i)* to foster research on sound event classification using a larger vocabulary of classes compared to previous Challenge Tasks; *ii)* to consolidate the research topic of learning with labels of varying levels of reliability, where labels can feature different levels of noise.

Both Challenge Tasks were organized as a collaboration between the Music Technology Group at Universitat Pompeu Fabra (UPF) and Sound Understanding at Google Research. The Challenge Tasks organizers were the author of this thesis and Frederic Font from Universitat Pompeu Fabra, and Manoj

[61] http://dcase.community/

[62] In this Appendix, for simplicity, sometimes we will refer to the organized DCASE Challenge Tasks as simply the *task(s)*.

Plakal and Daniel P.W. Ellis from Google Research. The problem formulations and recognition tasks were designed jointly, whereas the audio datasets were mainly developed at the UPF and the baseline systems were mainly developed at Google Research. Both competitions were hosted on the Kaggle platform,[63] for which Addison Howard and Walter Reade of Kaggle provided assistance with the the task design and deployment.

One of the most important outcomes of these Challenge Tasks is the generation of open knowledge. On the one hand, the preparation of the tasks included the creation of openly available datasets. After the Challenges concluded, the full list of reference labels were released (including development and evaluation data), thus enabling further research beyond the Challenges. Likewise, the preparation of the tasks also included the open release of baseline system frameworks that participants could use to build their systems on top of them. On the other hand, during the Challenges a number of participants were active in the Kaggle discussion forums exchanging ideas and related scientific papers. Further, some participants even shared their code, ranging from beginners' guides for audio experimentation to sophisticated deep learning approaches. In addition, Challenge winners were asked to publish code for the winning solutions under an open license.

Next, we report on the organization of the 2018 (Section A.2) and 2019 (Section A.3) DCASE Challenge Tasks, introducing the motivation, task setup, dataset and main outcomes. Links to multiple relevant web artifacts on the Kaggle and DCASE websites are included, should the reader want to consult more detailed task descriptions, datasets, baseline systems, or results.

A.2 DCASE 2018 Task 2: General-purpose Audio Tagging of Freesound Content with AudioSet Labels

In previous DCASE Challenge editions, there had been two audio tagging tasks, each focused on a specific domain of sounds. In DCASE 2016 (Mesaros et al., 2018a), the task targeted domestic audio tagging for which the CHiME-Home dataset was used, including 7 sound categories and 6.8h of recordings (Foster et al., 2015). In DCASE 2017 (Mesaros et al., 2017b), the task focused on audio tagging in the context of smart cars, for which a larger dataset featuring 17 categories was utilized.

Recently, however, general-purpose sound event recognizers have gained attention, where a wide range of sounds events are considered, not tied to a specific

[63] https://www.kaggle.com/

domain. This research has been mostly triggered by AudioSet, a large-scale audio dataset structured with an ontology of 632 sound events (Gemmeke et al., 2017).

In this DCASE Challenge Task, we focus on general-purpose audio tagging using a dataset of 41 categories and almost 18h of training data. Specifically, the goal of this task is to build an audio tagging system that can categorize an audio clip as belonging to one of a set of 41 diverse categories drawn from the AudioSet Ontology (related to musical instruments, human sounds, domestic sounds, animals, etc.). One of the motivations for this task comes from the large amount of user-generated audio content that is available on the web, which can be a resource of great potential for sound recognition related research. The use of such data for training audio tagging systems poses issues that have not been addressed in previous DCASE Challenges. In particular, this task deals with user-generated audio clips retrieved from Freesound, which are very diverse in terms of acoustic content, recording techniques, clip duration, etc. Likewise, these audio clips sometimes feature incomplete and inconsistent user-provided metadata. To prepare the dataset for this task, some audio clips were manually labeled using the subset of 41 categories, while a larger set of clips was automatically categorized on the basis of their existing user-provided metadata. As a result, the dataset features a small amount of reliable annotations, and a large amount of non-verified annotations that could include a small amount of label noise.

Therefore, this task addresses two main challenges of *i)* recognizing an increased number of diverse sound events, and *ii)* leveraging subsets of training data featuring annotations of varying reliability. Submissions to this task will provide insight towards the development of broadly-applicable sound event classifiers. Potential applications include automatic description of multimedia content, and acoustic monitoring applications.

A.2.1 Task Setup

The goal of this task is to predict the category for each audio clip in a test set. The task setup is a multi-class classification problem, and hence the systems to be developed in this task can be denoted as *single-tag* audio tagging systems, as illustrated in Figure A.1. This task was hosted on Kaggle—a platform for machine learning competitions—and ran from March 30th to July 31st 2018. The resources associated to this task (dataset download, submission, and leaderboard) can be found on the Kaggle competition page.[64]

[64] https://kaggle.com/c/freesound-audio-tagging
Note that competition name on Kaggle is abbreviated from the full DCASE Task name to "Freesound General-Purpose Audio Tagging Challenge".

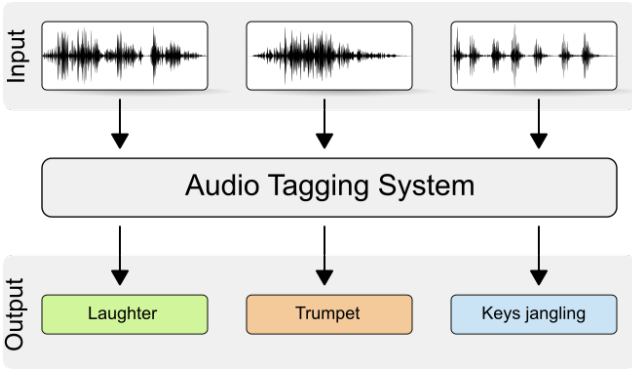

Figure A.1: Overview of a *single-tag* tagging system.

As described in Section A.2.2, the audio data for this task are split into a train set and a test set, both made publicly available when the competition launched. The train set, for which ground-truth annotations were provided, is used for system development while the test set is kept for the evaluation of the resulting systems. The test set, whose true labels were not released, is further divided into two divisions: *i)* 19% of the test samples are used to calculate the *public* leaderboard (providing a live ranking of all participants), and *ii)* the remaining 81% feeds the *private* leaderboard, used for the final ranking which is revealed only when the competition ends.

A.2.1.1 Evaluation Metric and Competition Rules

The task used mean Average Precision @ 3 (mAP@3) as the evaluation metric, as defined in the Evaluation section of the competition page.[64] This metric accepts up to three ranked predicted labels for each audio clip in the test set, and gives full credit if the correct label occurs first, with lesser credit for correct label predictions in second or third place.

Participants were required to run their systems on the test set and submit the system output—the predicted labels—in a comma-separated data file (CSV). Participants could submit a maximum of two submissions per day, and select two *final* submissions to be considered for the ranking. Additionally, participants were encouraged to submit a technical report describing their systems. A detailed description of the task rules can be found in the Rules section of the competition page,[64] and the most important points are summarized in the DCASE Challenge page.[65]

[65] http://dcase.community/challenge2018/task-general-purpose-audio-tagging#task-rules

A.2.1.2 Judges' Award

To complement the leaderboard results of the mAP-based ranking, the organizers of the task introduced a complementary Judges' Award to promote submissions using novel, problem-specific and efficient approaches. Details about the Judges' Award rules and requirements can be found in the Discussion section of the competition page.[66]

A.2.2 Dataset

The dataset used for the task was prepared by the task organizers during the months previous to the start of the competition, and is called "Freesound Dataset Kaggle 2018" (or *FSDKaggle2018* for short). FSDKaggle2018 can be considered a subset of FSD50K introduced in Chapter 3, except for a few audio clips that were discarded during FSD50K's curation process, and a few class labels that were merged into their parents for the final vocabulary of FSD50K. Next, we describe the creation process of FSDKaggle2018. The dataset can be downloaded from Zenodo.[67]

A.2.2.1 Annotation Procedure

FSDKaggle2018 is composed of audio content collected from Freesound, and it is organized using categories of the AudioSet Ontology. As a first step, we did a mapping of 268,261 Freesound clips to their corresponding AudioSet categories. To do that, we assigned a number of Freesound tags to almost all of the 632 AudioSet categories and, for each category, we selected audio clips from Freesound tagged with at least one of these tags. This process led to a number of automatically generated *candidate annotations* that express the potential presence of a sound category in an audio clip. These annotations are at the clip level and hence can be considered weak labels. However, some audio files are specific sound examples of the category under consideration, where the acoustic signal fills almost the entirety of the file, which could arguably be considered as strong labels.

In order to validate the candidate annotations, we used Freesound Annotator,[27] an online platform for the collaborative creation of open audio datasets (Section 3.2.3). We deployed a validation task in which Freesound Annotator users can manually verify the presence or absence of a candidate sound category in an audio clip with a *rating* mechanism. For every sound category, users first go through a training phase to get familiar with the category, read its description

[66] https://www.kaggle.com/c/freesound-audio-tagging/discussion/59932
[67] https://doi.org/10.5281/zenodo.2552859

provided by AudioSet, and listen to some selected sound examples. Then, users are presented with a series of audio clips, and prompted the question: *Is <category> present in the following sounds?* Users must select one of the response types listed in Table 3.1: Present and predominant (PP), Present but not predominant (PNP), Not Present (NP) or Unsure (U). Along with the audio clips, users are also given links to the corresponding Freesound sound pages where the original tags and descriptions are available and can be used as an aid for the validation process. Participants in the validation task included voluntaries from the Freesound community as well as researchers and students from the Music Technology Group.

Among the various features implemented in the validation task, it is worth mentioning the utilization of quality control mechanisms such as the periodic inclusion of verification clips to test the reliability of the submitted responses. Likewise, in order to choose which audio clips to present to each user, we adopt a prioritization scheme that considers inter-annotator agreement. More specifically, each candidate annotation is presented to several users (i.e., annotators) until agreement is attained by two different users on a response type. When a candidate annotation reaches an agreement status, it is considered validated and is no longer presented to other users.

A.2.2.2 Dataset Curation

After generating candidate annotations and collecting user ratings in the Freesound Annotator, each candidate annotation had a particular distribution of ratings {PP, PNP, NP, U} (see Table 3.1). Then, a curation step was carried out to select which categories and audio clips to be finally included in FSDKaggle2018. Considering all annotations, two annotation subsets were created for each sound category:

- **Manually-verified annotations**: composed of those annotations rated only as PP (a great majority with inter-annotator agreement but not all of them, hence PP & PP or single PP).
- **Non-verified annotations**: composed mainly of the *un-rated* candidate annotations, and complemented with a small amount of rated annotations. This small amount of rated annotations can include any rating distribution except *i)* those corresponding to the manually-verified portion, and *ii)* those that clearly denote an incorrect mapping (e.g., NP, NP & U, etc.).

For each sound category, a quality estimate QE for the non-verified annotations

can be computed according to

$$QE = \frac{\#PP + \#PNP}{\#PP + \#PNP + \#NP + \#U}, \tag{A.1}$$

where $\#X$ denotes the number of ratings of type X gathered in the category.

Next, a number of restrictions were applied sequentially to the categories and/or the audio clips within them. First, we discarded all categories not belonging to leaf nodes of the AudioSet hierarchy, leaving a total of 474 categories. Then, we removed audio clips shorter than 300ms and longer than 30s, as well as those clips with Creative Commons *Non-commercial* or *Sampling+* licenses. All sound categories that, after the previous filtering, did not have *i)* a minimum of number of manually-verified annotations, and *ii)* a minimum number of manually-verified + non-verified annotations, were discarded. Note that in order to accept the non-verified annotations in a category, a minimum QE was required (see Section A.2.2.3).

We observed that quite a few leaf sound categories were discarded because they did not have sufficient number of clips. In some of these cases, making use of the hierarchical relationships in AudioSet, we decided to aggregate the content of these leaf categories together with that of their immediate parents in order to create new candidate parent categories. Similar requirements (in terms of QE and amount of data) were applied to these newly formed categories for them to be accepted in the raw version of FSDKaggle2018.

After this process, an analysis was carried out in terms of *i)* number of *in-domain*[68] candidate annotations per audio clip and *ii)* semantic aspect of the resulting categories. The analysis revealed that the vast majority of the audio clips presented a single candidate annotation and, for the sake of simplicity, we decided to discard audio clips with multiple annotations.[69] We also discarded a few categories with somewhat abstract or vague meaning like "Recording" or "Effect unit".

Finally, the audio clips with manually-verified annotations for every category were split into roughly 70%/30% for train and test sets. The split was carried out considering clip duration (so as to have short and long clips in both sets). Then, we complemented the manually-verified portion of the train set with the non-verified annotations. This addition was performed such that the maximum number of clips per category was 300 in order to mitigate data imbalance among categories. The dataset curation resulted in the selected 11,073 sounds/annotations organized with 41 AudioSet categories.

[68] Considering only the set of valid categories at this point of the process, instead of all the AudioSet categories.

[69] Note that the automatically generated candidate annotations depend on the user-generated tags in Freesound and on the mapping to the AudioSet Ontology. Hence their reliability relies on the subsequent validation process.

A.2.2.3 Dataset Description

FSDKaggle2018 contains a total of 11,073 files provided as uncompressed PCM 16 bit, 44.1 kHz, mono audio files. All audio clips are released under either Creative Commons *Attribution* or *Zero* licenses. The clips are unequally distributed in 41 categories of the AudioSet Ontology. The full list can be inspected in the competition page.

The dataset most relevant characteristics are as follows:

- Audio clips are annotated with a single label.
- The duration of the audio clips ranges from 300ms to 30s due to the diversity of the sound categories and the preferences of Freesound users when recording sounds.
- The dataset is split into a train set and a test set.
- The **train set** is meant to be for system development and includes 9473 audio clips unequally distributed among 41 categories. The minimum number of audio clips per category in the train set is 94, and the maximum is 300. The total duration of the train set is almost 18h.
- Out of the 9473 clips from the train set, 3710 have manually-verified annotations and 5763 have non-verified annotations. The latter are properly flagged so that participants can opt to use this information during the development of their systems.
- The **test set** is composed of 1600 clips with manually-verified annotations and with a similar category distribution to that of the manually-verified portion of the train set. The minimum number of manually-verified audio clips per category in the test set is 25, and the maximum is 110. The test set is complemented with 7800 *padding* clips.[70] These clips, which are not used for scoring the systems, are added to prevent undesired practices (considering that the test set was made publicly available when the competition launched).

As mentioned in Section A.2.2.2, all **manually-verified annotations** are annotations validated as PP (Present and Predominant). This means that, in most cases, there is no additional acoustic material other than the labeled category. In few cases, there may be some additional sound events, but these additional events will be *out-of-vocabulary*, i.e., they won't belong to any of the 41 AudioSet categories of FSDKaggle2018. The **non-verified annotations** have a QE of *at least* 65% in each category. This means that some of them

[70] Hence, the dataset available from Kaggle contains 18,873 audio files.

are most probably inaccurate. It can happen that audio clips corresponding to some of the non-verified annotations present several sound sources (even though only one label is provided as ground truth). These additional sources are typically out-of-vocabulary, but in a few cases they could be within the vocabulary.

A.2.3 Challenge Outcomes

This scientific evaluation attracted 558 teams with a total of 5684 submissions in the Kaggle platform. At the end of the competition, 20 teams submitted technical reports to the DCASE Challenge describing their solutions.

Below we give a short summary of the main takeaways from the submissions to the DCASE Challenge. The full list of technical reports submitted to the DCASE Challenge is available from the DCASE website, along with multiple tables summarizing the main aspects of the systems.[71]

The issue of variable clip duration was usually addressed by selecting fixed-length chunks from the clips, either via random slicing or using fixed-length sliding windows. The most common acoustic representation was log-mel energies. Data augmentation was quite popular, especially mixup augmentation. Deep learning approaches were widely adopted, especially CNN architectures including mainly VGG, DenseNet, ResNet or ResNeXt, and also a few CRNN architectures. Nonetheless, conventional classifiers like support vector machine or gradient boosting machine were also used to some extent. Model ensembles were used heavily across the board, reaching up to a committee of 30 models. The problem of label noise was sometimes addressed with semi-supervised learning, and sometimes with noise robustness measures like label smoothing or loss masking.

A.3 DCASE 2019 Task 2: Audio Tagging with Noisy Labels and Minimal Supervision

In this DCASE Challenge Task, we seek to foster label noise research in general-purpose sound event tagging. We follow up on the DCASE Challenge 2018 Task 2 described in Section A.2, and propose to investigate the scenario where a small set of manually-labeled data is available, along with a larger set of noisy-labeled data, in a multi-label audio tagging setting, and using a vocabulary of 80 classes of everyday sounds.

The proposed task addresses two main research problems. The first problem is how to adequately exploit a large quantity of noisy labels, many of which are

[71] http://dcase.community/challenge2018/task-general-purpose-audio-tagging-results

incorrect and/or incomplete, and how to complement it with the supervision provided by a much smaller amount of reliable manually-labeled data (*minimal* supervision). The second problem is given by the acoustic mismatch between the noisy train set and the test set. Distribution shifts between data have been shown to cause substantial performance drops in machine learning, both for vision (Recht et al., 2019) and audio (Mesaros et al., 2018c). In our case, the noisy train set comes from a different web audio source than the test set, which is sometimes a real-world constraint.

A.3.1 Task Setup

The goal of this task is to predict appropriate labels for each audio clip in a test set. The predictions are to be done at the clip level, i.e., no start/end timestamps for the sound events are required. Some test clips bear one ground truth label while others bear multiple labels. Hence, the task setup is a *multi-label* classification problem and the systems to be developed can be denoted as multi-label audio tagging systems, as illustrated in Figure A.2. This task was hosted on the Kaggle platform from April 4th to June 10th 2019. The resources associated to this task (dataset download, submission, and leaderboard) can be found on the Kaggle competition page.[72]

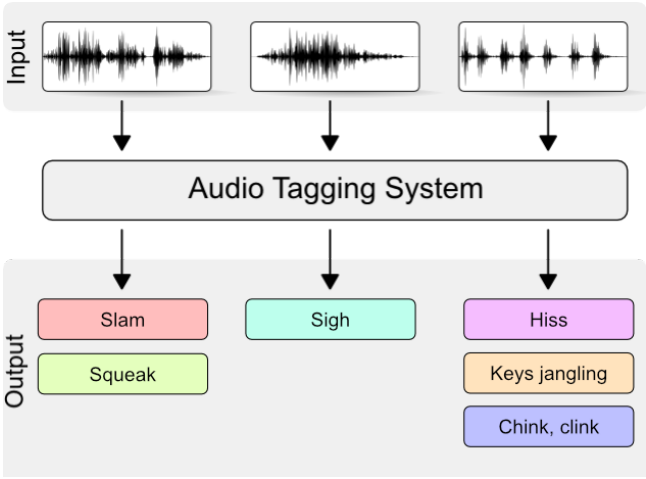

Figure A.2: Overview of a multi-label tagging system.

As described in Section A.3.2, the audio data for this task consists of a test set and two train sets: a *curated* train set and a *noisy* train set, that allow to experiment with training data of different levels of reliability and coming from

[72] https://www.kaggle.com/c/freesound-audio-tagging-2019
Note that the competition name on Kaggle is abbreviated from the full DCASE2019 Challenge task name to "Freesound Audio Tagging 2019".

different sources. System evaluation was carried out on Kaggle servers (see Section A.3.1.1) using the test set, which is further split into two divisions, for the *public* and *private* leaderboards. During the competition, the test subset corresponding to the public leaderboard was used to provide live ranking of all participants. To compute the final private leaderboard, at the end of the competition, systems were re-evaluated using the unseen private test set, of which neither the audio nor the labels were accessible to participants.

A.3.1.1 Evaluation Metric and Competition Rules

The task was evaluated using the *lwlrap* metric (see Section 2.3.7). This generalizes the mean reciprocal rank (MRR) used in the 2018 Challenge (Section A.2.1.1) to the case of multiple true labels per test item. In contrast to plain *lrap*, which averages precisions within a sample then across samples, thereby downweighting labels that occur on samples with many labels, *lwlrap* calculates the precision for each *label* in the test set, and gives them all equal contribution to the final metric. A Python implementation of *lwlrap* is provided online.[73]

This scientific evaluation was set up as a Kaggle *Kernels-only* competition. This means that all participants had to submit their systems as inference models in Kaggle Kernels (similar to *Jupyter Notebooks*), to be evaluated on remote servers. In addition, inference run-time was limited to a maximum of one hour in a Kernel with one GPU, and memory constraints were also imposed. These constraints aim to discourage the usage of large model ensembles that were prevalent in the previous 2018 Challenge edition. Participants could submit a maximum of two submissions per day, and select two *final* submissions to be considered for the private leaderboard ranking. A detailed description of the task rules can be found in the Rules section of the competition page;[72] the most important points are summarized in the DCASE Challenge page.[74]

To complement the leaderboard results of the *lwlrap* ranking, the task organizers introduced a complementary Judges' Award to promote submissions using novel, problem-specific and efficient approaches. Details about the Judges' Award rules can be found in the Judges' Award section of the Kaggle website.[72]

A.3.2 Dataset

The dataset used is called *FSDKaggle2019*, and it employs audio clips from the following sources:

[73] https://colab.research.google.com/drive/1AgPdhSp7ttY18O3fEoHOQKlt_3HJDLi8
[74] http://dcase.community/challenge2019/task-audio-tagging#task-rules

- Freesound Dataset (FSD): a dataset under development based on Freesound content organized with the AudioSet Ontology (Gemmeke et al., 2017).[75] These data are used to create the curated train set and the test set.
- The soundtracks of a pool of Flickr videos taken from the Yahoo Flickr Creative Commons 100M (YFCC100M) dataset (Thomee et al., 2016). These data are used to create the noisy train set.

FSDKaggle2019 is freely available from Zenodo,[76] all clips are provided as uncompressed PCM 16 bit 44.1 kHz mono audio files, its ground truth labels are provided at the clip-level (i.e., weak labels), and its partitioning is depicted in Figure A.3.

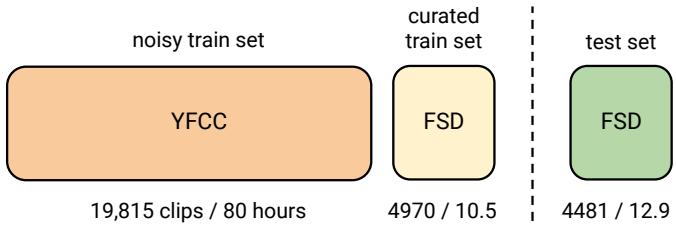

Figure A.3: Data split in FSDKaggle2019, including number of clips / duration in hours, and data origin. Colors depict quality of labels: orange, yellow and green correspond to noisy labels, correct but potentially incomplete labels, and exhaustive labels, respectively.

The human-labeled portions of FSDKaggle2019 (i.e., curated train set and test set) can be considered a subset of FSD50K introduced in Chapter 3, except for a few audio clips that were discarded during FSD50K's curation process.

A.3.2.1 Curated train set and test set

The first step carried out in the creation of FSDKaggle2019 was the definition of a vocabulary of 80 classes drawn from the AudioSet Ontology (Gemmeke et al., 2017). This vocabulary was chosen based on the following criteria: *i)* we consider leaf nodes of the AudioSet hierarchy for which there is enough data available in FSD, *ii)* we aim to encompass a diverse range of everyday sounds, and *iii)* we remove few clearly isolated classes (those with the weakest semantic relations with any of the rest), thus promoting confounds between semantically/acoustically similar classes to some extent. The main sound *families* (i.e.,

[75] This dataset under development would eventually be FSD50K, introduced in Chapter 3.
[76] https://doi.org/10.5281/zenodo.3612637

groups of sound classes) in the resulting vocabulary are, in descending order of prevalence, human sounds, domestic sounds, musical instruments, vehicles, animal sounds, natural sounds, materials, and mechanisms. The full list of 80 classes is available in the Data section of the Kaggle website.[72]

In a second step, we did a mapping of Freesound clips to the selected 80 class labels. To this end, a set of keywords was defined connecting the user-provided Freesound tags with the AudioSet labels. Using this mapping, for every class, we retrieved the audio clips that feature at least one of the defined keywords among their tags. This process led to a number of automatically-generated *candidate annotations* indicating the potential presence of a sound class in an audio clip (i.e., weak labels, as timing information is not included). Nonetheless, in some audio clips the target signal fills the clip length almost completely, which can be considered as a strong label. Subsequently, the candidate annotations were human-validated using a *validation* task deployed in Freesound Annotator[27] (Section 3.2.3). In this task, users verify the presence/absence of a candidate sound class in an audio clip with a *rating* mechanism. The vast majority of provided labels have inter-annotator agreement but not all of them. The outcome is a set of clips where the corresponding label(s) are correct; nevertheless, it can happen that a few of these audio clips present additional acoustic material beyond the provided label(s).

The resulting data were split into a train set and a test set. We refer to this train set as *curated* in order to distinguish it from the noisy set described in Section A.3.2.2. To mitigate train-test contamination, the split was carried out considering the clip uploaders in Freesound. We allocated all audio clips uploaded from the same user into either the curated train set or the test set, so that the sets are disjoint at the Freesound user level. The partition proportion was defined to limit the supervision provided in the curated train set, thus promoting approaches to deal with label noise.

Finally, labels in the test set were further refined using the refinement annotation tool described in Section 3.2.7. This annotation task consists of two stages: *i)* pre-existent labels can be re-validated, and *ii)* potentially missing labels can be added through exploration of the AudioSet Ontology (Favory et al., 2018). The outcome is a set of exhaustively labeled clips where the label(s) are correct and complete considering the target vocabulary; nonetheless, few clips could still present additional (unlabeled) acoustic content out of the vocabulary.

The main characteristics of the curated train set, noisy train set and test set are listed in Table A.1. The curated train set consists of 4970 clips with a total of 5752 labels. Labels per clip ranges from 1 to 6 with a mean of 1.2. The test set consists of 4481 clips with a total of 6250 labels. Labels per clip ranges from 1 to 6 with a mean of 1.4. Note the increased number of labels per clip with respect to the curated train set, due to the process of exhaustive labelling.

Table A.1: Main stats of the sets in FSDKaggle2019. *A few classes have slightly less than 75 clips.

Aspect	curated train	noisy train	test
Clips/class	∼75*	300	∼ 50 - 150
Total clips	4970	19,815	4481
Labels/clip	1.2	1.2	1.4
Clip length	∼0.3 - 30s	∼15s	∼0.3 - 30s
Total duration	∼10.5h	∼80h	∼12.9h
Labelling	correct (inexhaustive)	noisy	exhaustive

In both cases, clip length ranges from 0.3s to 30 due to the diversity of the sound classes and the preferences of Freesound users when recording/uploading sounds.

A.3.2.2 Noisy train set

The noisy train set was prepared using the YFCC100M dataset (Thomee et al., 2016), which has the advantages of *i)* being a very large and diverse dataset that is not correlated with Freesound in acoustics or domain, and *ii)* offering permissive Creative Commons licenses that allow ease of use, modification, and redistribution. The original dataset contained ∼99M photos and ∼793k videos from ∼581K Flickr users. We dropped videos with licenses that disallowed making derivatives or commercial use, videos that were no longer available, and videos with audio decode errors that we could not transcode, leaving us with ∼201K 44.1 kHz mono WAV files. Video length varied with a maximum of 20 minutes, and a mean of ∼37s and median of ∼20s.

The Flickr video metadata (title, description, tags) proved to be too sparse to meaningfully map to our class vocabulary. Therefore, we used a content-based approach where we generated video-level predictions from a variety of pre-trained audio models: a shallow fully-connected network as well as variants of VGG and ResNet (Hershey et al., 2017), all of which were trained on a large collection of YouTube videos using the AudioSet class vocabulary. We generated sliding windows of ∼1s containing log mel spectrogram patches and aggregated the per-window predictions (using either maximum or average pooling) to produce a video-level vector of class scores. For each of our 80 classes, we kept the top 300 videos by predicted score for that class. We browsed the video labels and selected the maximum-pooled VGG-like model as producing a balance between reasonable predictions and a substantial amount of label noise. As a further source of noise, each final clip was produced by

taking a random slice of a video of length up to 15 seconds (videos shorter than 15 seconds would be taken in their entirety). Hence, the label noise can vary widely in amount and type depending on the class, including in- and out-of-vocabulary noises (see Section 5.2.2).

As listed in Table A.1, the noisy train set consists of 19,815 clips with a total of 24,000 labels (300 * 80). Labels per clip ranges from 1 to 7 with a mean of 1.2. Clip length ranges from 1s to 15s (by construction), with a mean of 14.5s. Therefore, the per-class training data distribution in FSDKaggle2019 is, for most of the classes, 300 clips from the noisy set and 75 clips from the curated set. This means 80% noisy / 20% curated at the clip level, while at the duration level the proportion is more extreme considering the variable-length clips. Since most of the train data come from YFCC100M, acoustic domain mismatch between the train and test set can be expected. We conjecture this mismatch comes from a variety of reasons. For example, through acoustic inspection of a small sample of both data sources, we find a higher percentage of high quality recordings in Freesound. In addition, audio clips in Freesound are typically recorded with the purpose of capturing audio, which is not necessarily the case in YFCC100M.

A.3.3 Challenge Outcomes

This scientific evaluation attracted 880 teams with a total of 8618 submissions in the Kaggle platform. At the end of the competition, 14 teams submitted technical reports to the DCASE Challenge describing their solutions.

Below we give a short summary of the main takeaways from the submissions to the DCASE Challenge. The full list of technical reports submitted to the DCASE Challenge is available from the DCASE website, along with multiple tables summarizing the main aspects of the systems.[77]

Similar to the 2018 Challenge edition, the most popular acoustic representation was log-mel energies; however, in this edition more teams turned to raw waveforms as input representation. All submissions used deep learning approaches, especially CNN and CRNN architectures, including VGG, DenseNet, ResNe(X)t, Shake-Shake, Frequency-Aware CNNs, Squeeze-and-Excitation, and MobileNet. Also in a similar fashion to the 2018 Challenge edition, a widespread use of model ensembles was observed, sometimes heavily. Successful exploitation of the the noisy train set appeared to be challenging. Hence a number of participants focused on exploiting the smaller clean set using data augmentation methods such as mixup, multiple variants of SpecAugment, SpecMix, or Test Time Augmentation. To tackle the problem posed by noisy labels, a variety of approaches were utilized rather than a clear common

[77] http://dcase.community/challenge2019/task-audio-tagging-results

trend. The proposed approaches include semi-supervised learning for instance selection or pseudo-label generation, multi-task learning, robust loss functions and per-class loss weighting, stochastic weight averaging, adaptive-weighting of noisy samples, and MixMatch.

APPENDIX B

Publications by the Author

In this Appendix, we provide the full list of the author's publications during this thesis, either as a first author (12 papers) or through various collaborations (7 papers). This list of publications is also available from the author's Google Scholar profile.[78]

B.1 Peer-reviewed Journal Articles

1. **Fonseca, E.**, Favory, X., Pons, J., Font, F., & Serra, X. (2020). FSD50K: an Open Dataset of Human-Labeled Sound Events. Under review in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. (https://arxiv.org/abs/2010.00475) (Fonseca et al., 2020a)

2. **Fonseca, E.**, Hershey, S., Plakal, M., Ellis, D. P. W., Jansen, A., & Moore, R. C. (2020). Addressing Missing Labels in Large-Scale Sound Event Recognition Using a Teacher-Student Framework With Loss Masking. *IEEE Signal Processing Letters*, 27, 1235-1239. (https://arxiv.org/abs/2005.00878) (Fonseca et al., 2020b)

3. **Fonseca, E.**, Ferraro, A., & Serra, X. (2021). Improving Sound Event Classification by Increasing Shift Invariance in Convolutional Neural Networks. Under review in *IEEE Signal Processing Letters*. (https://arxiv.org/abs/2107.00623) (Fonseca et al., 2021a)

[78] https://scholar.google.com/citations?user=RZodv5QAAAAJ&hl=en

B.2 Peer-reviewed Conference Articles

1. **Fonseca, E.**, Jansen, A., Ellis, D. P. W., Wisdom, S., Tagliasacchi, M., Hershey, J. R., Plakal, M., Hershey, S., Moore, R. C., & Serra, X. (2021). Self-Supervised Learning from Automatically Separated Sound Scenes. In *Proceedings of the 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*.
(https://arxiv.org/abs/2105.02132) (Fonseca et al., 2021b)

2. **Fonseca, E.**, Ortego, D., McGuinness, K., O'Connor, N. E., & Serra, X. (2021). Unsupervised Contrastive Learning of Sound Event Representations. In *Proceedings of the 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 371-375.
(https://arxiv.org/abs/2011.07616) (Fonseca et al., 2021c)

3. **Fonseca, E.**, Font, F., & Serra, X. (2019). Model-agnostic Approaches to Handling Noisy Labels When Training Sound Event Classifiers. In *Proceedings of the 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 16-20.
(https://arxiv.org/abs/1910.12004) (Fonseca et al., 2019a)

4. **Fonseca, E.**, Plakal, M., Font, F., Ellis, D. P. W., & Serra, X. (2019). Audio Tagging with Noisy Labels and Minimal Supervision. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 69-73.
(https://arxiv.org/abs/1906.02975) (Fonseca et al., 2019c)

5. **Fonseca, E.**, Plakal, M., Ellis, D. P. W., Font, F., Favory, X., & Serra, X. (2019). Learning Sound Event Classifiers from Web Audio with Noisy Labels. In *Proceedings of the 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 21-25.
(https://arxiv.org/abs/1901.01189) (Fonseca et al., 2019b)

6. **Fonseca, E.**, Plakal, M., Font, F., Ellis, D. P. W., Favory, X., Pons, J., & Serra, X. (2018). General-purpose Tagging of Freesound Audio with AudioSet Labels: Task Description, Dataset, and Baseline. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 69-73.
(https://arxiv.org/abs/1807.09902) (Fonseca et al., 2018b)

7. **Fonseca, E.**, Gong, R., & Serra, X. (2018). A Simple Fusion of Deep and Shallow Learning for Acoustic Scene Classification. In *Proceedings of 5th Sound & Music Computing Conference (SMC 2018)*, pp. 265-272.
(https://arxiv.org/abs/1806.07506) (Fonseca et al., 2018a)

8. **Fonseca, E.**, Pons, J., Favory, X., Font, F., Bogdanov, D., Ferraro, A., Oramas, S., Porter, A., & Serra, X. (2017). Freesound Datasets: A Platform for the Creation of Open Audio Datasets. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, pp. 486-493.
(https://repositori.upf.edu/handle/10230/33299) (Fonseca et al., 2017b)

9. **Fonseca, E.**, Gong, R., Bogdanov, D., Slizovskaia, O., Gómez, E., & Serra, X. (2017). Acoustic Scene Classification by Ensembling Gradient Boosting Machine and Convolutional Neural Networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pp. 37-41.
(https://repositori.upf.edu/handle/10230/33454) (Fonseca et al., 2017a)

B.3 Peer-reviewed Conference Articles through Collaborations

1. Zinemanas, P., Rocamora, M., **Fonseca, E.**, Font, F., & Serra, X. (2021). Towards Interpretable Sound Event Detection with Attention Based on Prototypes. Accepted for publication in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*.
(To appear.) (Zinemanas et al., 2021)

2. Hershey, S., Ellis, D. P. W., **Fonseca, E.**, Jansen, A., Liu, C., Moore, R. C., & Plakal, M. (2021). The Benefit of Temporally-Strong Labels in Audio Event Classification. In *Proceedings of the 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 366-370.
(https://arxiv.org/abs/2105.07031) (Hershey et al., 2021)

3. Turpault, N., Serizel, R., Wisdom, S., Erdogan, H., Hershey, J.R., **Fonseca, E.**, Seetharaman, P., & Salamon, J. (2021). Sound Event Detection and Separation: a Benchmark on DESED Synthetic Soundscapes. In *Proceedings of the 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 840-844.
(https://arxiv.org/abs/2011.00801) (Turpault et al., 2021)

4. Wisdom, S., Erdogan, H., Ellis, D. P. W., Serizel, R., Turpault, N., **Fonseca, E.**, Salamon, J., Seetharaman, P., & Hershey, J.R. (2021). What's All the FUSS About Free Universal Sound Separation Data?. In *Proceedings of the 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 186-190.
(https://arxiv.org/abs/2011.00803) (Wisdom et al., 2021)

5. Turpault, N., Wisdom, S., Erdogan, H., Hershey, J.R., Serizel, R., **Fonseca, E.**, Seetharaman, P., & Salamon, J. (2020). Improving Sound Event Detection In Domestic Environments Using Sound Separation. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, pp. 205-209.
(https://arxiv.org/abs/2007.03932) (Turpault et al., 2020b)

6. Pérez-López, A., **Fonseca, E.**, & Serra, X. (2019). A Hybrid Parametric-Deep Learning Approach for Sound Event Localization and Detection. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 189-193.
(https://arxiv.org/abs/1908.10133) (Pérez-López et al., 2019)

7. Favory, F., **Fonseca, E.**, & Serra, X. (2018). Facilitating the Manual Annotation of Sounds When Using Large Taxonomies. In *Proceedings of the 23rd Conference of Open Innovations Association, FRUCT*, pp. 447-451.
(https://arxiv.org/abs/1811.10988) (Favory et al., 2018)

B.4 Technical Reports

- Gong, R., **Fonseca, E.**, Bogdanov, D., Slizovskaia, O., Gómez, E., & Serra, X. (2017). Acoustic Scene Classification by Fusing LightGBM and VGG-net Multichannel Predictions. Technical Report. *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events 2017 (DCASE2017 Challenge)*.
(http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Gong_189.pdf) (Gong et al., 2017)

APPENDIX C

Other Academic Contributions and Merits by the Author

In this Appendix, we provide a list of other academic contributions by the author related with the thesis work.

C.1 Academic Contributions

- Technical Program Co-Chair of DCASE Workshop 2021.
  (http://dcase.community/workshop2021/organizers)
- Co-organizer of DCASE Challenge 2018 Task 2 "General-purpose audio tagging of Freesound content with AudioSet labels".
  (http://dcase.community/challenge2018/task-general-purpose-audio-tagging)
- Co-organizer of DCASE Challenge 2019 Task 2 "Audio tagging with noisy labels and minimal supervision".
  (http://dcase.community/challenge2019/task-audio-tagging)

C.2 Awards

- Paper shortlisted for Best Student Paper Award in WASPAA 2021: "Self-Supervised Learning From Automatically Separated Sound Scenes" (Fonseca et al., 2021b).
- Developer of awarded proposal for Google Faculty Research Awards 2017 in the Machine Perception category.
  (https://research.google/outreach/past-programs/faculty-research-awards/?category=2017)

- Developer of awarded proposal for Google Faculty Research Awards 2018 in the Machine Perception category.
  (https://research.google/outreach/past-programs/faculty-research-awards/?category=2018)
- Awardee of 3 Nvidia GPU cards.

APPENDIX D

Resources

In this thesis, we strived to make available as many resources as possible for the sake of open research and reproducibility. Next, we provide the URLs to access the data and code resources developed in this thesis. These URLs have also been referenced where appropriate in the main body of this manuscript.

D.1 Datasets

- **FSD50K** is an audio dataset of human-labeled sound events containing 51,197 Freesound clips unequally distributed in 200 classes drawn from the AudioSet Ontology.
  (https://doi.org/10.5281/zenodo.4060431)
  (Companion site: https://annotator.freesound.org/fsd/release/FSD50K/)

- **FSDnoisy18k** is an audio dataset collected with the aim of fostering the investigation of label noise in sound event classification. It contains 42.5 hours of audio across 20 sound classes, including a small amount of manually-labeled data and a larger quantity of real-world noisy data.
  (https://doi.org/10.5281/zenodo.2529933)
  (Companion site: http://www.eduardofonseca.net/FSDnoisy18k/)

- **FSDKaggle2019** is an audio dataset containing 29,266 audio files annotated with 80 labels of the AudioSet Ontology. FSDKaggle2019 has been used for the DCASE Challenge 2019 Task 2.
  (https://doi.org/10.5281/zenodo.3612636)

- **FSDKaggle2018** is an audio dataset containing 11,073 audio files annotated with 41 labels of the AudioSet Ontology. FSDKaggle2018 has been used for the DCASE Challenge 2018 Task 2.
  (https://doi.org/10.5281/zenodo.2552859)

D.2 Code

- Code for the paper "Learning Sound Event Classifiers from Web Audio with Noisy Labels" (Fonseca et al., 2019b).
  (https://github.com/edufonseca/icassp19)
- Code for the paper "Model-agnostic Approaches to Handling Noisy Labels When Training Sound Event Classifiers" (Fonseca et al., 2019a).
  (https://github.com/edufonseca/waspaa19)
- Code for the paper "Unsupervised Contrastive Learning of Sound Event Representations" (Fonseca et al., 2021c).
  (https://github.com/edufonseca/uclser20)
- Code for the paper "Improving Sound Event Classification by Increasing Shift Invariance in Convolutional Neural Networks" (Fonseca et al., 2021a).
  (https://github.com/edufonseca/shift_sec)
- Code for the paper "FSD50K: an Open Dataset of Human-Labeled Sound Events" (Fonseca et al., 2020a).
  (https://github.com/edufonseca/FSD50K_baseline)

APPENDIX E

Additional Figures and Tables

In this Appendix, we provide one additional Figure to support the discussion of results in Chapter 4.

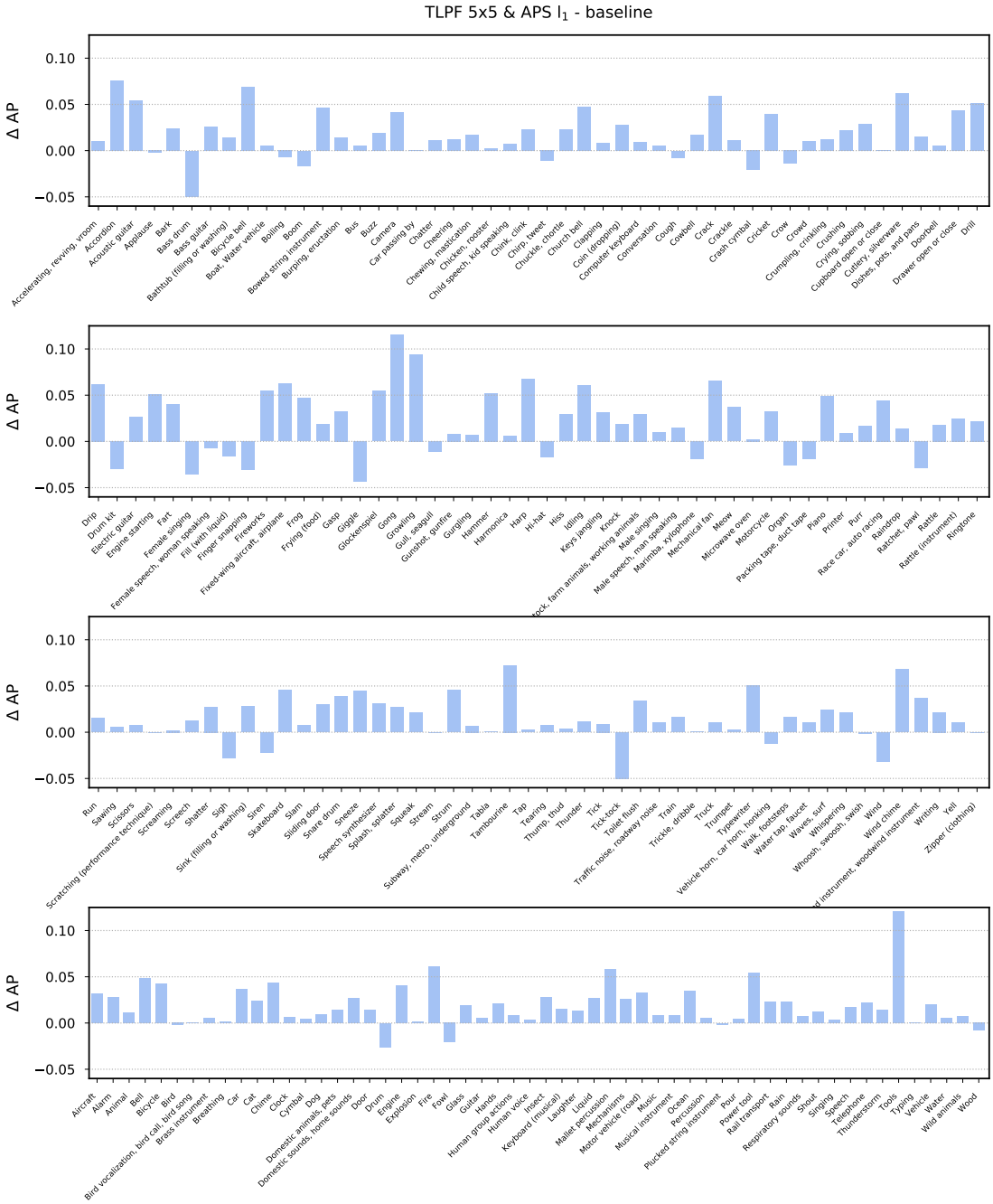

Figure E.1: Per-class increment/decrement of AP for all classes in FSD50K when inserting TLPF 5x5 and APS l_1 on VGG42 with mixup (see right column of Table 4.4). Top 3 rows show the 144 leaf nodes and bottom row comprise the 56 intermediate nodes.

APPENDIX F

Glossary

F.1 Acronyms

lwlrap	Label-Weighted Label-Ranking Average Precision
AP	Average Precision
APS	Adaptive Polyphase Sampling
BCE	Binary Cross-Entropy
CCE	Categorical Cross-Entropy
CNN	Convolutional Neural Network
CNNs	Convolutional Neural Networks
CRNN	Convolutional Recurrent Neural Network
CRNNs	Convolutional Recurrent Neural Networks
DCASE	Detection and Classification of Acoustic Scenes and Events
FAQs	Frequently Asked Questions
IBP	Intra-block Pooling
IQA	Internal Quality Assessment
IV	In-Vocabulary
kNN	k-Nearest Neighbour
LSR	Label Smoothing Regularization
MAC	Mean Absolute Change
MAE	Mean Absolute Error
mAP	Mean Average Precision
MFCCs	Mel-frequency Cepstral Coefficients
MIR	Music Information Retrieval
MixIT	Mixture Invariant Training
MLP	Multi-Layer Perceptron
MLPs	Multi-Layer Perceptrons
OOV	Out-Of-Vocabulary
QbE	Query by Example

ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RNNs	Recurrent Neural Networks
SEC	Sound Event Classification
SED	Sound Event Detection
SER	Sound Event Recognition
SET	Sound Event Tagging
SNR	Signal-to-Noise Ratio
T-F	Time-Frequency
TLPF	Trainable Low-pass Filter
VGG	Visual Geometry Group

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., & Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. [Cited on page 94.]

Abdoli, S., Cardinal, P., & Koerich, A. L. (2019). End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Systems with Applications*, *136*, 252–263. [Cited on page 30.]

Abeßer, J. (2021). USM-SED: A dataset for polyphonic sound event detection in urban sound monitoring scenarios. *arXiv preprint arXiv:2105.02592*. [Cited on page 91.]

Adavanne, S., Pertilä, P., & Virtanen, T. (2017). Sound event detection using spatial features and convolutional recurrent neural network. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 771–775. IEEE. [Cited on page 48.]

Adavanne, S., Politis, A., & Virtanen, T. (2019). A multi-room reverberant dataset for sound event localization and detection. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 10–14. New York University, NY, USA. [Cited on pages 5 and 42.]

Alayrac, J.-B., Recasens, A., Schneider, R., Arandjelović, R., Ramapuram, J., De Fauw, J., Smaira, L., Dieleman, S., & Zisserman, A. (2020). Self-supervised multimodal versatile networks. *Neural Information Processing Systems*. [Cited on pages 199 and 200.]

Algan, G. & Ulusoy, I. (2021). Image classification with deep learning in the presence of noisy labels: A survey. *Knowledge-Based Systems*, *215*, 106771. [Cited on pages 50 and 51.]

Andén, J. & Mallat, S. (2014). Deep scattering spectrum. *IEEE Transactions on Signal Processing*, *62*(16), 4114–4128. [Cited on page 30.]

Arandjelovic, R. & Zisserman, A. (2017). Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 609–617. [Cited on pages 57, 188, 199, and 200.]

Arazo, E., Ortego, D., Albert, P., O'Connor, N., & McGuinness, K. (2019). Unsupervised label noise modeling and loss correction. In *International Conference on Machine Learning*, pp. 312–321. PMLR. [Cited on page 145.]

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y. et al. (2017). A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 233–242. [Cited on pages 8, 50, 129, 138, and 144.]

Azulay, A. & Weiss, Y. (2018). Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*. [Cited on pages 6, 48, 105, and 120.]

Ba, J. & Caruana, R. (2014). Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, pp. 2654–2662. [Cited on pages 152 and 153.]

Baevski, A., Schneider, S., & Auli, M. (2019). vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*. [Cited on pages 10 and 56.]

Baevski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, *33*. [Cited on pages 10 and 56.]

Ballas, J. A. (1993). Common factors in the identification of an assortment of brief everyday sounds. *Journal of experimental psychology: human perception and performance*, *19*(2), 250. [Cited on page 64.]

Barchiesi, D., Giannoulis, D., Stowell, D., & Plumbley, M. D. (2015). Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, *32*(3), 16–34. [Cited on page 35.]

Barz, B. & Denzler, J. (2020). Do we train on test data? Purging CIFAR of near-duplicates. *Journal of Imaging*, *6*(6), 41. [Cited on pages 61 and 72.]

Becker, S. & Hinton, G. E. (1992). Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, *355*(6356), 161–163. [Cited on page 56.]

Bello, J. P., Silva, C., Nov, O., DuBois, R. L., Arora, A., Salamon, J., Mydlarz, C., & Doraiswamy, H. (2019). SONYC: A system for monitoring, analyzing, and mitigating urban noise pollution. *Communications of the ACM*, *62*(2), 68–77. [Cited on page 2.]

Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., & Oord, A. v. d. (2020). Are we done with ImageNet? *arXiv preprint arXiv:2006.07159*. [Cited on page 61.]

Bilen, Ç., Ferroni, G., Tuveri, F., Azcarreta, J., & Krstulović, S. (2020). A framework for the robust evaluation of sound event detection. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 61–65. IEEE. [Cited on page 36.]

Bogdanov, D., Porter, A., Boyer, H., Serra, X. et al. (2016). Cross-collection evaluation for music classification tasks. In *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*. ISMIR. [Cited on page 91.]

Bornschein, J., Visin, F., & Osindero, S. (2020). Small data, big decisions: Model selection in the small-data regime. In *International Conference on Machine Learning*, pp. 1035–1044. PMLR. [Cited on page 203.]

Bruguier, A., Misra, A., Narayanan, A., & Prabhavalkar, R. (2020). Anti-aliasing regularization in stacking layers. *Proc. Interspeech 2020*, pp. 314–318. [Cited on page 49.]

Cakir, E., Heittola, T., Huttunen, H., & Virtanen, T. (2015). Polyphonic sound event detection using multi label deep neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE. [Cited on pages 3 and 34.]

Cakir, E., Ozan, E. C., & Virtanen, T. (2016). Filterbank learning for deep neural network based polyphonic sound event detection. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 3399–3406. IEEE. [Cited on page 91.]

Cakır, E., Parascandolo, G., Heittola, T., Huttunen, H., & Virtanen, T. (2017). Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *25*(6), 1291–1303. [Cited on pages 20, 34, 48, 94, 95, and 97.]

Cartwright, M., Cramer, J., Mendez Mendez, A. E., Wang, Y., Wu, H.-H., Lostanlen, V., Fuentes, M., Dove, G., Mydlarz, C., Salamon, J., Nov, O., & Bello, J. P. (2020). SONYC-UST-V2: An urban sound tagging dataset with spatiotemporal context. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, pp. 16–20. Tokyo, Japan. [Cited on pages 5, 39, and 41.]

Cartwright, M., Cramer, J., Salamon, J., & Bello, J. P. (2019a). TriCycle: Audio representation learning from sensor network data using self-supervision. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 278–282. IEEE. [Cited on page 181.]

Cartwright, M., Dove, G., Méndez Méndez, A. E., Bello, J. P., & Nov, O. (2019b). Crowdsourcing multi-label audio annotation tasks with citizen scientists. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–11. [Cited on page 44.]

Cartwright, M., Mendez, A. E. M., Cramer, J., Lostanlen, V., Dove, G., Wu, H.-H., Salamon, J., Nov, O., & Bello, J. (2019c). SONYC urban sound tagging (SONYC-UST): A multilabel dataset from an urban acoustic sensor network. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 35–39. [Cited on pages 5, 29, 68, 72, and 174.]

Cartwright, M., Seals, A., Salamon, J., Williams, A., Mikloska, S., MacConnell, D., Law, E., Bello, J. P., & Nov, O. (2017). Seeing sound: Investigating the effects of visualizations and complexity on crowdsourced audio annotations. *Proceedings of the ACM on Human-Computer Interaction*, *1*(CSCW), 1–21. [Cited on page 68.]

Castellani, A., Schmitt, S., & Hammer, B. (2021). Estimating the electrical power output of industrial devices with end-to-end time-series classification in the presence of label noise. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases ECML/PKDD*. [Cited on page 214.]

Chaman, A. & Dokmanic, I. (2021). Truly shift-invariant convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3773–3783. [Cited on pages 49, 106, 110, 111, and 120.]

Chen, H., Xie, W., Vedaldi, A., & Zisserman, A. (2020a). VGGSound: A large-scale audio-visual dataset. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 721–725. IEEE. [Cited on pages 41 and 45.]

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020b). A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PMLR. [Cited on pages 10, 55, 56, 57, 167, 169, 171, 172, 173, 174, 175, 179, 180, 181, 182, 186, 187, 193, 197, 201, and 204.]

Chen, X., Fan, H., Girshick, R., & He, K. (2020c). Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*. [Cited on pages 10, 56, 57, 167, 174, and 179.]

Chen, Z.-M., Wei, X.-S., Wang, P., & Guo, Y. (2019). Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5177–5186. [Cited on page 53.]

Cheng, K.-H., Chou, S.-Y., & Yang, Y.-H. (2019). Multi-label few-shot learning for sound event recognition. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–5. IEEE. [Cited on page 76.]

Choi, K., Fazekas, G., Sandler, M., & Cho, K. (2017). Convolutional recurrent neural networks for music classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2392–2396. IEEE. [Cited on page 20.]

Cortes, C., Mohri, M., & Rostamizadeh, A. (2012). L2 regularization for learning kernels. *arXiv preprint arXiv:1205.2653*. [Cited on page 109.]

Cotton, C. V. & Ellis, D. P. (2011). Spectral vs. spectro-temporal features for acoustic event detection. In *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 69–72. IEEE. [Cited on pages 3 and 34.]

Cramer, J., Lostanlen, V., Farnsworth, A., Salamon, J., & Bello, J. P. (2020). Chirping up the right tree: Incorporating biological taxonomies into deep bioacoustic classifiers. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 901–905. IEEE. [Cited on pages 2, 41, 47, 73, 91, and 98.]

Crocco, M., Cristani, M., Trucco, A., & Murino, V. (2016). Audio surveillance: A systematic review. *ACM Computing Surveys (CSUR)*, *48*(4), 1–46. [Cited on page 2.]

Davis, J. & Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240. ACM. [Cited on page 94.]

Davis, S. & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, *28*(4), 357–366. [Cited on pages 20 and 30.]

Dekkers, G., Lauwereins, S., Thoen, B., Adhana, M. W., Brouckxon, H., van Waterschoot, T., Vanrumste, B., Verhelst, M., & Karsmakers, P. (2017). The SINS database for detection of daily activities in a home environment using an acoustic sensor network. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pp. 32–36. [Cited on page 5.]

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE. [Cited on pages 20, 43, 49, 61, and 62.]

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186. [Cited on page 56.]

DeVries, T. & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with Cutout. *arXiv preprint arXiv:1708.04552*. [Cited on pages 31 and 33.]

Dieleman, S., De Fauw, J., & Kavukcuoglu, K. (2016). Exploiting cyclic symmetry in convolutional neural networks. In *International conference on machine learning*, pp. 1889–1898. PMLR. [Cited on page 49.]

Dieleman, S. & Schrauwen, B. (2014). End-to-end learning for music audio. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6964–6968. IEEE. [Cited on page 47.]

Distante, A., Distante, C., Distante, & Wheeler (2020). *Handbook of Image Processing and Computer Vision*. Springer. [Cited on page 109.]

Dorfer, M. & Widmer, G. (2018). Training general-purpose audio tagging networks with noisy labels and iterative self-verification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 178–182. [Cited on pages 52, 95, and 112.]

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*. [Cited on page 20.]

Drugman, T., Urbain, J., Bauwens, N., Chessini, R., Aubriot, A.-S., Lebecque, P., & Dutoit, T. (2012). Audio and contact microphones for cough detection. In *Thirteenth Annual Conference of the International Speech Communication Association*. [Cited on page 2.]

Ebbers, J. & Häb-Umbach, R. (2019). Convolutional recurrent neural network and data augmentation for audio tagging with noisy labels and minimal supervision. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 64–68. New York University, NY, USA. [Cited on page 95.]

Ebrahimpour, M., Shea, T., Danielescu, A., Noelle, D., & Kello, C. (2020). End-to-end auditory object recognition via inception nucleus. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 146–150. IEEE. [Cited on pages 112 and 113.]

EBU Recommendation R 128 (2014). Loudness normalisation and permitted maximum level of audio signals. *European Broadcasting Union*. [Cited on page 68.]

Elizalde, B., Shah, A., Dalmia, S., Lee, M. H., Badlani, R., Kumar, A., Raj, B., & Lane, I. (2017). An approach for self-training audio event detectors using web data. In *25th European Signal Processing Conference (EUSIPCO)*, pp. 1863–1867. IEEE. [Cited on pages 52 and 136.]

Elizalde, B., Zarar, S., & Raj, B. (2019). Cross modal audio search and retrieval with joint embeddings based on text and audio. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4095–4099. IEEE. [Cited on page 91.]

Ellis, D. P. (2009). Gammatone-like spectrograms. *web resource: http://www.ee.columbia.edu/dpwe/resources/matlab/gammatonegram*. [Cited on pages 30 and 31.]

Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., & Madry, A. (2018). A rotation and a translation suffice: Fooling CNNs with simple transformations. [Cited on pages 6, 49, and 105.]

Eronen, A. J., Peltonen, V. T., Tuomi, J. T., Klapuri, A. P., Fagerlund, S., Sorsa, T., Lorho, G., & Huopaniemi, J. (2006). Audio-based context recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(1), 321–329. [Cited on page 25.]

Estellés-Arolas, E. & González-Ladrón-de Guevara, F. (2012). Towards an integrated crowdsourcing definition. *Journal of Information science*, *38*(2), 189–200. [Cited on page 43.]

Fan, J., Ma, X., Wu, L., Zhang, F., Yu, X., & Zeng, W. (2019). Light gradient boosting machine: An efficient soft computing model for estimating daily reference evapotranspiration with local and external meteorological data. *Agricultural Water Management*, *225*, 105758. [Cited on page 214.]

Favory, X., Drossos, K., Virtanen, T., & Serra, X. (2020a). Coala: Co-aligned autoencoders for learning semantically enriched audio representations. In *International Conference on Machine Learning (ICML), Workshop on Self-supervised learning in Audio and Speech*. [Cited on page 91.]

Favory, X., Fonseca, E., Font, F., & Serra, X. (2018). Facilitating the manual annotation of sounds when using large taxonomies. In *Proceedings of the 23rd Conference of Open Innovations Association, FRUCT*, 23, pp. 447–451. FRUCT Oy. [Cited on pages 75, 231, and 238.]

Favory, X., Font, F., & Serra, X. (2020b). Search result clustering in collaborative sound collections. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pp. 207–214. [Cited on page 91.]

Flexer, A. & Schnitzer, D. (2009). Album and artist effects for audio similarity at the scale of the web. *Children*, *15*(15.95), 4–07. [Cited on pages 72 and 99.]

Foggia, P., Petkov, N., Saggese, A., Strisciuglio, N., & Vento, M. (2015). Reliable detection of audio events in highly noisy environments. *Pattern Recognition Letters*, *65*, 22–28. [Cited on pages 3, 25, and 34.]

Fonseca, E., Favory, X., Pons, J., Font, F., & Serra, X. (2020a). FSD50K: an Open Dataset of Human-Labeled Sound Events. *arXiv preprint arXiv:2010.00475*. [Cited on pages 12, 24, 48, 112, 208, 214, 235, and 242.]

Fonseca, E., Ferraro, A., & Serra, X. (2021a). Improving sound event classification by increasing shift invariance in convolutional neural networks. *arXiv preprint arXiv:2107.00623*. [Cited on pages 12, 91, 208, 235, and 242.]

Fonseca, E., Font, F., & Serra, X. (2019a). Model-agnostic approaches to handling noisy labels when training sound event classifiers. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 16–20. IEEE. [Cited on pages 12, 47, 89, 91, 95, 113, 208, 236, and 242.]

Fonseca, E., Gong, R., Bogdanov, D., Slizovskaia, O., Gómez, E., & Serra, X. (2017a). Acoustic scene classification by ensembling gradient boosting machine and convolutional neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*. [Cited on pages 214 and 237.]

Fonseca, E., Gong, R., & Serra, X. (2018a). A simple fusion of deep and shallow learning for acoustic scene classification. In *Proceedings of the 15th Sound & Music Computing Conference (SMC 2018)*. Limassol, Cyprus. [Cited on pages 47, 137, 214, and 236.]

Fonseca, E., Hershey, S., Plakal, M., Ellis, D. P., Jansen, A., & Moore, R. C. (2020b). Addressing missing labels in large-scale sound event recognition using a teacher-student framework with loss masking. *IEEE Signal Processing Letters*, *27*, 1235–1239. [Cited on pages 6, 12, 47, 61, 71, 89, 91, 95, 209, and 235.]

Fonseca, E., Jansen, A., Ellis, D. P. W., Wisdom, S., Tagliasacchi, M., Hershey, J. R., Plakal, M., Hershey, S., Moore, R. C., & Serra, X. (2021b). Self-supervised learning from automatically separated sound scenes. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. [Cited on pages 12, 61, 167, 202, 209, 236, and 239.]

Fonseca, E., Ortego, D., McGuinness, K., O'Connor, N. E., & Serra, X. (2021c). Unsupervised contrastive learning of sound event representations. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 371–375. IEEE. [Cited on pages 12, 167, 168, 209, 214, 236, and 242.]

Fonseca, E., Plakal, M., Ellis, D. P. W., Font, F., Favory, X., & Serra, X. (2019b). Learning Sound Event Classifiers from Web Audio with Noisy Labels. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 5, 12, 39, 41, 89, 91, 208, 214, 236, and 242.]

Fonseca, E., Plakal, M., Font, F., Ellis, D. P., Favory, X., Pons, J., & Serra, X. (2018b). General-purpose tagging of Freesound audio with AudioSet labels: task description, dataset, and baseline. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*. [Cited on pages 12, 91, 174, 209, and 236.]

Fonseca, E., Plakal, M., Font, F., Ellis, D. P. W., & Serra, X. (2019c). Audio tagging with noisy labels and minimal supervision. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*. NY, USA. [Cited on pages 5, 12, 39, 41, 91, 209, and 236.]

Fonseca, E., Pons, J., Favory, X., Font, F., Bogdanov, D., Ferraro, A., Oramas, S., Porter, A., & Serra, X. (2017b). Freesound Datasets: a platform for the creation of open audio datasets. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, pp. 486–493. Suzhou, China. [Cited on pages 64 and 237.]

Font, F. (2015). *Tag recommendation using folksonomy information for online sound sharing platforms*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on page 3.]

Font, F., Roma, G., & Serra, X. (2013). Freesound technical demo. In *Proceedings of the 21st ACM international conference on Multimedia*, pp. 411–412. ACM. [Cited on page 45.]

Ford, L., Tang, H., Grondin, F., & Glass, J. (2019). A deep residual network for large-scale acoustic scene analysis. *Proc. Interspeech 2019*, pp. 2568–2572. [Cited on pages 53 and 97.]

Foster, P., Sigtia, S., Krstulovic, S., Barker, J., & Plumbley, M. D. (2015). CHiME-home: A dataset for sound source recognition in a domestic environment. In *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE. [Cited on pages 3, 8, 38, 39, 43, 61, 68, and 220.]

Frénay, B. & Verleysen, M. (2014). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, *25*(5). [Cited on pages 8, 50, and 132.]

Gao, J., Li, Z., Nevatia, R. et al. (2017). Knowledge concentration: Learning 100k object classifiers in a single CNN. *arXiv preprint arXiv:1711.07607*. [Cited on page 82.]

Gaver, W. W. (1993). What in the world do we hear?: An ecological approach to auditory event perception. *Ecological psychology*, *5*(1), 1–29. [Cited on page 23.]

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., & Ritter, M. (2017). Audio Set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. New Orleans, LA. [Cited on pages 4, 8, 9, 15, 22, 23, 24, 26, 29, 36, 37, 39, 40, 43, 53, 61, 62, 66, 79, 84, 87, 91, 94, 112, 123, 131, 134, 150, 151, 154, 155, 160, 173, 185, 199, 200, 221, and 230.]

Gharib, S., Drossos, K., Cakir, E., Serdyuk, D., & Virtanen, T. (2018). Unsupervised adversarial domain adaptation for acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 138–142. [Cited on page 91.]

Gharib, S., Drossos, K., Fagerlund, E., & Virtanen, T. (2019). VOICe: A sound event detection dataset for generalizable domain adaptation. *arXiv preprint arXiv:1911.07098*. [Cited on pages 5 and 42.]

Ghosh, A., Kumar, H., & Sastry, P. (2017). Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31. [Cited on pages 51 and 143.]

Goldberger, J. & Ben-Reuven, E. (2017). Training deep neural-networks using a noise adaptation layer. [Cited on page 8.]

Gong, R., Fonseca, E., Bogdanov, D., Slizovskaia, O., Gomez, E., & Serra, X. (2017). Acoustic scene classification by fusing LightGBM and VGG-net multichannel predictions. Tech. rep., DCASE2017 Challenge. [Cited on page 238.]

Gong, Y., Chung, Y.-A., & Glass, J. (2021a). AST: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*. [Cited on pages 20, 21, and 34.]

Gong, Y., Chung, Y.-A., & Glass, J. (2021b). PSLA: Improving audio event classification with pretraining, sampling, labeling, and aggregation. *arXiv preprint arXiv:2102.01243*. [Cited on pages 6, 20, 33, 47, 48, 113, 118, 127, 199, 200, and 202.]

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press. [Cited on pages 5, 25, 31, 46, 47, 113, 139, and 140.]

Green, D. M., Swets, J. A. et al. (1966). *Signal detection theory and psychophysics*, vol. 1. Wiley New York. [Cited on page 36.]

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Pires, B., Guo, Z., Azar, M. et al. (2020). Bootstrap your own latent: A new approach to self-supervised learning. In *Neural Information Processing Systems*. [Cited on pages 10 and 56.]

Guastavino, C. (2018). Everyday sound categorization. *Computational Analysis of Sound Scenes and Events*, pp. 183–213. [Cited on page 23.]

Guzhov, A., Raue, F., Hees, J., & Dengel, A. (2021). ESResNet: Environmental sound classification based on visual domain models. In *International Conference on Pattern Recognition (ICPR)*, pp. 4933–4940. IEEE. [Cited on page 20.]

Gygi, B. (2001). *Factors in the identification of environmental sounds*. Ph.D. thesis, Indiana University Bloomington, IN. [Cited on pages xxv, 17, 18, and 19.]

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., & Sugiyama, M. (2018). Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, pp. 8527–8537. [Cited on pages 8, 51, 130, 142, 145, and 162.]

Han, W., Coutinho, E., Ruan, H., Li, H., Schuller, B., Yu, X., & Zhu, X. (2016). Semi-supervised active learning for sound classification in hybrid learning environments. *PloS one*, *11*(9), e0162075. [Cited on pages 45, 52, and 215.]

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738. [Cited on page 193.]

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778. [Cited on pages 20, 21, 95, 154, 155, and 174.]

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pp. 630–645. Springer. [Cited on page 137.]

Heittola, T. (2021). *Computational Audio Content Analysis in Everyday Environments*. Ph.D. thesis, Tampere University. [Cited on pages xxv, 18, and 19.]

Heittola, T., Çakır, E., & Virtanen, T. (2018). The machine learning approach for analysis of sound scenes and events. In *Computational Analysis of Sound Scenes and Events*, pp. 13–40. Springer. [Cited on pages 25 and 26.]

Hendrycks, D., Lee, K., & Mazeika, M. (2019a). Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pp. 2712–2721. PMLR. [Cited on pages 50 and 201.]

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., & Song, D. (2019b). Natural adversarial examples. *arXiv preprint arXiv:1907.07174*. [Cited on page 68.]

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B. et al. (2017). CNN architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135. IEEE. [Cited on pages 6, 36, 37, 94, 112, 154, 216, and 232.]

Hershey, S., Ellis, D. P. W., Fonseca, E., Jansen, A., Liu, C., Moore, R. C., & Plakal, M. (2021). The benefit of temporally-strong labels in audio event classification. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 18, 40, 82, 216, and 237.]

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *NIPS Deep Learning and Representation Learning Workshop*. [Cited on page 153.]

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. [Cited on pages 154 and 155.]

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141. [Cited on page 147.]

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708. [Cited on pages xxi, 20, 95, 147, and 148.]

Hüwel, A., Adiloğlu, K., & Bach, J.-H. (2020). Hearing aid research data set for acoustic environment recognition. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 706–710. IEEE. [Cited on page 2.]

Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456. [Cited on pages 46, 95, 113, 137, and 192.]

Iqbal, T., Cao, Y., Kong, Q., Plumbley, M. D., & Wang, W. (2020). Learning with out-of-distribution data for audio classification. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 636–640. IEEE. [Cited on pages 52, 89, and 95.]

Jaegle, A., Gimeno, F., Brock, A., Zisserman, A., Vinyals, O., & Carreira, J. (2021). Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*. [Cited on page 34.]

Jansen, A., Ellis, D. P., Hershey, S., Moore, R. C., Plakal, M., Popat, A. C., & Saurous, R. A. (2020). Coincidence, Categorization, and Consolidation: Learning to recognize sounds with minimal supervision. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 121–125. IEEE. [Cited on pages 55, 57, 167, 180, 182, 183, 188, 189, 190, 193, 194, 196, 198, 199, 201, and 202.]

Jansen, A., Gemmeke, J. F., Ellis, D. P., Liu, X., Lawrence, W., & Freedman, D. (2017). Large-scale audio event discovery in one million youtube videos. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 786–790. IEEE. [Cited on page 2.]

Jansen, A., Plakal, M., Pandya, R., Ellis, D. P., Hershey, S., Liu, J., Moore, R. C., & Saurous, R. A. (2018). Unsupervised learning of semantic audio representations. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 126–130. IEEE. [Cited on pages 6, 54, 57, 61, 95, 167, 170, 182, 183, 190, 193, 194, 199, and 201.]

Jantzen, S. G., Sutherland, B. J., Minkley, D. R., & Koop, B. F. (2011). GO Trimming: Systematically reducing redundancy in large gene ontology datasets. *BMC research notes*, *4*(1), 267. [Cited on page 24.]

Jati, A., Kumar, N., Chen, R., & Georgiou, P. (2019). Hierarchy-aware loss function on a tree structured label space for audio event detection. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6–10. IEEE. [Cited on page 91.]

Jeong, I.-Y. & Lim, H. (2018). Audio tagging system using densely connected convolutional networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 197–201. [Cited on page 147.]

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., & Fei-Fei, L. (2018). Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pp. 2304–2313. PMLR. [Cited on page 51.]

Kavalerov, I., Wisdom, S., Erdogan, H., Patton, B., Wilson, K., Le Roux, J., & Hershey, J. R. (2019a). Universal sound separation. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 175–179. [Cited on page 91.]

Kavalerov, I., Wisdom, S., Erdogan, H., Patton, B., Wilson, K., Le Roux, J., & Hershey, J. R. (2019b). Universal sound separation. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 175–179. IEEE. [Cited on page 185.]

Kawakami, K., Wang, L., Dyer, C., Blunsom, P., & van den Oord, A. (2020). Learning robust and multilingual speech representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pp. 1182–1192. [Cited on pages 9, 10, 55, and 204.]

Kharitonov, E., Rivière, M., Synnaeve, G., Wolf, L., Mazaré, P.-E., Douze, M., & Dupoux, E. (2021). Data augmenting contrastive learning of speech representations in the time domain. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pp. 215–222. IEEE. [Cited on page 57.]

Kim, B. & Pardo, B. (2019). Sound event detection using point-labeled data. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE. [Cited on pages 95 and 153.]

Kingma, D. P. & Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*. [Cited on pages 94, 112, 137, 155, and 192.]

Kong, Q., Cao, Y., Iqbal, T., Wang, Y., Wang, W., & Plumbley, M. D. (2020a). PANNs: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Pro-*

cessing, *28*, 2880–2894. [Cited on pages 6, 20, 21, 30, 33, 47, 48, 53, 94, 95, 96, 97, 112, 113, 118, 192, 199, 200, 201, 202, and 215.]

Kong, Q., Xu, Y., Wang, W., & Plumbley, M. D. (2020b). Sound event detection of weakly labelled data with CNN-transformer and automatic threshold optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *28*, 2450–2460. [Cited on pages 36 and 48.]

Krause, B. (2012). *The great animal orchestra: finding the origins of music in the world's wild places*. Little, Brown. [Cited on page 1.]

Krizhevsky, A. & Hinton, G. (2009). Learning multiple layers of features from tiny images. Tech. rep., University of Toronto. [Cited on page 61.]

Kumar, A. & Ithapu, V. K. (2019). Secost: Sequential co-supervision for weakly labeled audio event detection. *arXiv preprint arXiv:1910.11789*. [Cited on page 89.]

Kumar, A. & Ithapu, V. K. (2020). SeCoST: Sequential co-supervision for large scale weakly labeled audio event detection. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 666–670. IEEE. [Cited on page 53.]

Kumar, A. & Raj, B. (2017). Audio event and scene recognition: A unified approach using strongly and weakly labeled data. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3475–3482. IEEE. [Cited on pages 9, 52, and 91.]

Kumar, A., Shah, A., Hauptmann, A. G., & Raj, B. (2019). Learning sound events from webly labeled data. In *International Joint Conference on Artificial Intelligence (IJCAI)*. [Cited on page 53.]

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T. et al. (2020). The Open Images dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*. [Cited on pages 5 and 59.]

Lafay, G., Benetos, E., & Lagrange, M. (2017). Sound event detection in synthetic audio: Analysis of the DCASE 2016 task results. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 11–15. [Cited on page 88.]

Law, E. L., Von Ahn, L., Dannenberg, R. B., & Crawford, M. (2007). TagATune: A game for music and sound annotation. In *Proceedings of the International Symposium on Music Information Retrieval*, vol. 3, p. 2. [Cited on page 43.]

Le-Khac, P. H., Healy, G., & Smeaton, A. F. (2020). Contrastive representation learning: A framework and review. *IEEE Access*, *8*, 193907–193934. [Cited on pages 56, 166, 173, 187, 189, and 193.]

Lee, H., Pham, P., Largman, Y., & Ng, A. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. *22*, 1096–1104. [Cited on pages 10, 47, 54, and 57.]

Lee, K. & Ellis, D. P. (2009). Audio-based semantic concept classification for consumer video. *IEEE Transactions on Audio, Speech, and Language Processing*, *18*(6), 1406–1416. [Cited on pages 20 and 25.]

Lei, H., Choi, J., Janin, A., & Friedland, G. (2011). User verification: Matching the uploaders of videos across accounts. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2404–2407. IEEE. [Cited on page 90.]

Li, J., Seltzer, M. L., Wang, X., Zhao, R., & Gong, Y. (2017a). Large-scale domain adaptation via teacher-student learning. *arXiv preprint arXiv:1708.05466*. [Cited on page 88.]

Li, J., Zhao, R., Huang, J.-T., & Gong, Y. (2014). Learning small-size DNN with output-distribution-based criteria. In *Fifteenth Annual Conference of the International Speech Communication Association*. [Cited on page 153.]

Li, W., Wang, L., Li, W., Agustsson, E., & Van Gool, L. (2017b). WebVision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*. [Cited on pages 5 and 59.]

Lim, H., Park, J., & Han, Y. (2017). Rare sound event detection using 1D convolutional recurrent neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pp. 80–84. [Cited on page 48.]

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer. [Cited on pages 5 and 59.]

Liu, H. & Motoda, H. (2002). On issues of instance selection. *Data Mining and Knowledge Discovery*, *6*(2), 115. [Cited on pages 99 and 158.]

Lostanlen, V., Andén, J., & Lagrange, M. (2018). Extended playing techniques: the next milestone in musical instrument recognition. In *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, pp. 1–10. [Cited on page 20.]

Lostanlen, V., Salamon, J., Farnsworth, A., Kelling, S., & Bello, J. P. (2019). Robust sound event detection in bioacoustic sensor networks. *PloS one*, *14*(10), e0214168. [Cited on page 2.]

Luo, Y. & Mesgarani, N. (2019). Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing*, *27*(8), 1256–1266. [Cited on page 185.]

Lyon, R. F. (2017). *Human and Machine Hearing: Extracting Meaning from Sound*. Cambridge University Press. [Cited on pages 2 and 17.]

Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., & Van Der Maaten, L. (2018). Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 181–196. [Cited on page 47.]

Malach, E. & Shalev-Shwartz, S. (2017). Decoupling" when to update" from" how to update". In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 961–971. [Cited on page 51.]

Malfait, L., Berger, J., & Kastner, M. (2006). P. 563—the itu-t standard for single-ended speech quality assessment. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(6), 1924–1934. [Cited on page 88.]

Mandel, M. I. & Ellis, D. P. (2005). Song-level features and support vector machines for music classification. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR 2005)*. [Cited on pages 72 and 99.]

Mandel, M. I. & Ellis, D. P. (2008). A web-based game for collecting music metadata. *Journal of New Music Research*, *37*(2), 151–165. [Cited on page 43.]

Martín-Morató, I., Harju, M., & Mesaros, A. (2021). Crowdsourcing strong labels for sound event detection. [Cited on page 44.]

Masterton, B., Heffner, H., & Ravizza, R. (1969). The evolution of human hearing. *The Journal of the Acoustical Society of America*, *45*(4), 966–985. [Cited on page 1.]

McFee, B. (2018). Statistical methods for scene and event classification. *Computational Analysis of Sound Scenes and Events*, pp. 103–146. [Cited on pages 25 and 47.]

McFee, B., Humphrey, E., & Urbano, J. (2016). A plan for sustainable mir evaluation. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 285–291. [Cited on page 43.]

McFee, B., Kim, J. W., Cartwright, M., Salamon, J., Bittner, R. M., & Bello, J. P. (2018a). Open-source practices for music signal processing research: Recommendations for transparent, sustainable, and reproducible audio research. *IEEE Signal Processing Magazine*, *36*(1), 128–137. [Cited on page 59.]

McFee, B., Salamon, J., & Bello, J. P. (2018b). Adaptive pooling operators for weakly labeled sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *26*(11), 2180–2193. [Cited on page 61.]

Meire, M., Karsmakers, P., & Vuegen, L. (2019). The impact of missing labels and overlapping sound events on multi-label multi-instance learning for sound event classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*. NY, USA. [Cited on pages 53 and 71.]

Mesaros, A., Heittola, T., Benetos, E., Foster, P., Lagrange, M., Virtanen, T., & Plumbley, M. D. (2017a). Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *26*(2), 379–393. [Cited on pages 2 and 88.]

Mesaros, A., Heittola, T., Benetos, E., Foster, P., Lagrange, M., Virtanen, T., & Plumbley, M. D. (2018a). Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, *26*(2), 379–393. [Cited on page 220.]

Mesaros, A., Heittola, T., Diment, A., Elizalde, B., Shah, A., Vincent, E., Raj, B., & Virtanen, T. (2017b). DCASE 2017 challenge setup: Tasks, datasets and baseline system. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pp. 85–92. [Cited on pages 42, 48, 88, and 220.]

Mesaros, A., Heittola, T., & Ellis, D. (2018b). Datasets and evaluation. In *Computational Analysis of Sound Scenes and Events*, pp. 147–179. Springer. [Cited on pages 29, 35, and 71.]

Mesaros, A., Heittola, T., Eronen, A., & Virtanen, T. (2010). Acoustic event detection in real life recordings. In *2010 18th European Signal Processing Conference*, pp. 1267–1271. IEEE. [Cited on pages 3, 25, and 34.]

Mesaros, A., Heittola, T., & Virtanen, T. (2016). TUT database for acoustic scene classification and sound event detection. In *24th European Signal Processing Conference 2016 (EUSIPCO 2016)*. Budapest, Hungary. [Cited on pages 42 and 88.]

Mesaros, A., Heittola, T., & Virtanen, T. (2018c). Acoustic scene classification: an overview of DCASE 2017 challenge entries. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pp. 411–415. IEEE. [Cited on page 228.]

Mesaros, A., Heittola, T., & Virtanen, T. (2018d). A multi-device dataset for urban acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 9–13. [Cited on page 79.]

Messner, E., Fediuk, M., Swatek, P., Scheidl, S., Smolle-Jüttner, F.-M., Olschewski, H., & Pernkopf, F. (2020). Multi-channel lung sound classification with convolutional recurrent neural networks. *Computers in Biology and Medicine*, p. 103831. [Cited on page 2.]

Meyer, M., Wenner, M., Hibert, C., Walter, F., & Thiele, L. (2021). Using system context information to complement weakly labeled data. *arXiv preprint arXiv:2107.10236*. [Cited on page 214.]

Minderer, M., Bachem, O., Houlsby, N., & Tschannen, M. (2020). Automatic shortcut removal for self-supervised representation learning. In *International Conference on Machine Learning*, pp. 6927–6937. PMLR. [Cited on page 180.]

Miyazaki, K., Komatsu, T., Hayashi, T., Watanabe, S., Toda, T., & Takeda, K. (2020). Convolution-augmented transformer for semi-supervised sound event detection. Tech. rep., DCASE2020 Challenge. [Cited on page 34.]

Morfi, V. & Stowell, D. (). Data-efficient weakly supervised learning for low-resource audio event detection using deep learning. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*. [Cited on pages 87, 94, and 133.]

Morrison, M. & Pardo, B. (2019). OtoMechanic: Auditory automobile diagnostics via query-by-example. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 169–173. New York University, NY, USA. [Cited on page 2.]

Nair, V. & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 807–814. [Cited on page 46.]

Nandan, A. & Vepa, J. (2020). Language Agnostic Speech Embeddings for Emotion Classification. In *International Conference on Machine Learning (ICML) Workshop*. [Cited on pages 57, 169, 178, 182, and 186.]

Narendra, N. & Alku, P. (2020). Glottal source information for pathological voice detection. *IEEE Access*, *8*, 67745–67755. [Cited on page 214.]

Nguyen, D. T., Mummadi, C. K., Ngo, T. P. N., Nguyen, T. H. P., Beggel, L., & Brox, T. (2019). SELF: Learning to filter noisy labels with self-ensembling. In *International Conference on Learning Representations*. [Cited on page 51.]

Nguyen, T. N. T., Nguyen, N. K., Jones, D. L., & Gan, W. S. (2018). DCASE 2018 task 2: iterative training, label smoothing, and background noise normalization for audio event tagging. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*. [Cited on page 52.]

Niizumi, D., Takeuchi, D., Ohishi, Y., Harada, N., & Kashino, K. (2021). BYOL for audio: Self-supervised learning for general-purpose audio representation. *arXiv preprint arXiv:2103.06695*. [Cited on pages 91, 174, and 204.]

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, *70*, 1373–1411. [Cited on pages 89 and 156.]

Oord, A. v. d., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*. [Cited on pages 10, 56, and 175.]

Oppenheim, A. V., Buck, J. R., & Schafer, R. W. (2001). *Discrete-time signal processing. Vol. 2*. Upper Saddle River, NJ: Prentice Hall. [Cited on pages 49, 107, and 109.]

Palanisamy, K., Singhania, D., & Yao, A. (2020). Rethinking CNN models for audio classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [Cited on page 20.]

Parascandolo, G., Huttunen, H., & Virtanen, T. (2016). Recurrent neural networks for polyphonic sound event detection in real life recordings. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6440–6444. IEEE. [Cited on pages 3 and 34.]

Parcollet, T. & Ravanelli, M. (2021). The Energy and Carbon Footprint of Training End-to-End Speech Recognizers. In *Proc. Interspeech 2021*. [Cited on page 217.]

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., & Le, Q. V. (2019). SpecAugment: A simple data augmentation method for automatic speech recognition. *Proc. Interspeech 2019*, pp. 2613–2617. [Cited on pages 31, 33, 167, 171, 178, 179, 186, and 192.]

Park, H. & Yoo, C. D. (2020). CNN-based learnable gammatone filterbank and equal-loudness normalization for environmental sound classification. *IEEE Signal Processing Letters*, *27*, 411–415. [Cited on pages 31 and 91.]

Peeters, G. & Deruty, E. (2010). Sound indexing using morphological description. *IEEE Transactions on audio, speech, and language processing*, *18*(3), 675–687. [Cited on page 23.]

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., & Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. [Cited on page 50.]

Pérez-López, A., Fonseca, E., & Serra, X. (2019). A hybrid parametric-deep learning approach for sound event localization and detection. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 189–193. New York University, NY, USA. [Cited on pages 48, 94, and 238.]

Petrescu, L., Petrescu, C., Mitruț, O., Moise, G., Moldoveanu, A., Moldoveanu, F., & Leordeanu, M. (2020). Integrating biosignals measurement in virtual reality environments for anxiety detection. *Sensors*, *20*(24), 7088. [Cited on page 214.]

Phaye, S. S. R., Benetos, E., & Wang, Y. (2019). Subspectralnet–using subspectrogram based convolutional neural networks for acoustic scene classification. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 825–829. IEEE. [Cited on page 47.]

Piczak, K. J. (2015a). Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. IEEE. [Cited on pages 3 and 34.]

Piczak, K. J. (2015b). ESC: Dataset for environmental sound classification. In *Proceedings of the ACM International Conference on Multimedia*, pp. 1015–1018. ACM. [Cited on pages 3, 8, 23, 26, 38, 39, 43, 46, 61, and 174.]

Ponomarchuk, A., Burenko, I., Malkin, E., Nazarov, I., Kokh, V., Avetisian, M., & Zhukov, L. (2021). Project Achoo: A practical model and application for COVID-19 detection from recordings of breath, voice, and cough. *arXiv preprint arXiv:2107.10716*. [Cited on page 214.]

Porter, M. F. et al. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137. [Cited on page 65.]

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do ImageNet classifiers generalize to ImageNet? In *International Conference on Machine Learning*, pp. 5389–5400. PMLR. [Cited on pages 61 and 228.]

Reed, S. E., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., & Rabinovich, A. (2015). Training deep neural networks on noisy labels with bootstrapping. In *International Conference on Learning Representations*. [Cited on pages 8, 51, 132, and 143.]

Riccardi, G. & Hakkani-Tur, D. (2005). Active learning: Theory and applications to automatic speech recognition. *IEEE transactions on speech and audio processing*, *13*(4), 504–511. [Cited on page 44.]

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. et al. (2015). ImageNet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 211–252. [Cited on pages 4, 17, 59, and 171.]

Sabou, M., Bontcheva, K., Derczynski, L., & Scharl, A. (2014). Corpus annotation through crowdsourcing: Towards best practice guidelines. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 14)*, pp. 859–866. [Cited on pages 43 and 70.]

Saeed, A., Grangier, D., & Zeghidour, N. (2021). Contrastive learning of general-purpose audio representations. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3875–3879. IEEE. [Cited on pages 167, 168, 173, 182, 194, 202, and 204.]

Sainath, T. N., Vinyals, O., Senior, A., & Sak, H. (2015). Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584. IEEE. [Cited on pages 20 and 97.]

Salamon, J. & Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, *24*(3), 279–283. [Cited on pages 6, 31, 35, 47, 88, and 137.]

Salamon, J., Jacoby, C., & Bello, J. P. (2014). A dataset and taxonomy for urban sound research. In *Proceedings of the ACM International Conference on Multimedia*, pp. 1041–1044. ACM. [Cited on pages 3, 8, 23, 26, 38, 39, 43, 46, 61, 68, 87, and 174.]

Salamon, J., MacConnell, D., Cartwright, M., Li, P., & Bello, J. P. (2017). Scaper: A library for soundscape synthesis and augmentation. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 344–348. IEEE. [Cited on pages 42, 48, and 87.]

Sánchez-Hevia, H. A., Ayllón, D., Gil-Pita, R., & Rosa-Zurera, M. (2017). Maximum likelihood decision fusion for weapon classification in wireless acoustic sensor networks. *IEEE/ACM transactions on audio, speech, and language processing*, *25*(6), 1172–1182. [Cited on page 2.]

Schaeffer, P. (1966). Traite des objets musicaux. *Paris: Seuil*. [Cited on pages 23 and 63.]

Schaeffer, P. (2016). *Traité des objets musicaux*. Le Seuil. [Cited on page 84.]

Schilit, B., Adams, N., & Want, R. (1994). Context-aware computing applications. In *Mobile Computing Systems and Applications*, pp. 85–90. IEEE. [Cited on page 1.]

Scikit-learn documentation 0.23.2 (2020a). Metrics and scoring: quantifying the quality of predictions. https://scikit-learn.org/stable/modules/model_evaluation.html. [Cited on page 36.]

Scikit-learn documentation 0.23.2 (2020b). Precision-Recall. https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html. [Cited on page 36.]

Sechidis, K., Tsoumakas, G., & Vlahavas, I. (2011). On the stratification of multi-label data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 145–158. Springer. [Cited on pages 73, 79, and 98.]

Sener, O. & Savarese, S. (2018). Active learning for convolutional neural networks: A core-set approach. In *6th International Conference on Learning Representations, ICLR*. [Cited on page 45.]

Serizel, R., Turpault, N., Shah, A., & Salamon, J. (2020). Sound event detection in synthetic domestic environments. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 86–90. IEEE. [Cited on page 68.]

Shah, A., Kumar, A., Hauptmann, A. G., & Raj, B. (2018). A closer look at weak label learning for audio events. *CoRR*, *abs/1804.09288*. [Cited on pages 52, 87, 132, and 133.]

Shor, J., Jansen, A., Maor, R., Lang, O., Tuval, O., de Chaumont Quitry, F., Tagliasacchi, M., Shavitt, I., Emanuel, D., & Haviv, Y. (2020). Towards learning a universal non-semantic representation of speech. In *Proc. Interspeech 2020*. [Cited on page 10.]

Shorten, C. & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*(1), 1–48. [Cited on page 31.]

Shrivaslava, H., Yin, Y., Shah, R. R., & Zimmermann, R. (2020). Mt-Gcn for multi-label audio tagging with noisy labels. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 136–140. IEEE. [Cited on page 91.]

Shrivastava, H., Yin, Y., Shah, R. R., & Zimmermann, R. (2020). MT-GCN for multi-label audio-tagging with noisy labels. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 136–140. IEEE. [Cited on page 53.]

Shuyang, Z., Heittola, T., & Virtanen, T. (2018). An active learning method using clustering and committee-based sample selection for sound event classification. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pp. 116–120. IEEE. [Cited on page 45.]

Shuyang, Z., Heittola, T., & Virtanen, T. (2020). Active learning for sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *28*, 2895–2905. [Cited on page 45.]

Simonyan, K. & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*. [Cited on pages 95, 106, and 112.]

Singh, S., Pankajakshan, A., & Benetos, E. (2019). Audio tagging using linear noise modelling layer. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 234–238. New York University, NY, USA. [Cited on page 52.]

Song, H., Kim, M., Park, D., Shin, Y., & Lee, J.-G. (2020). Learning from noisy labels with deep neural networks: A survey. *arXiv preprint arXiv:2007.08199*. [Cited on page 50.]

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958. [Cited on pages 50, 138, and 192.]

Stowell, D. & Plumbley, M. (2014). An open dataset for research on audio field recording archives: freefield1010. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society. [Cited on page 46.]

Sun, C., Shrivastava, A., Singh, S., & Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 843–852. [Cited on pages 3 and 158.]

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9. [Cited on pages 20, 171, and 178.]

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826. [Cited on pages 50 and 139.]

Tagliasacchi, M., Gfeller, B., de Chaumont Quitry, F., & Roblek, D. (2020). Pre-training audio representations with self-supervision. *IEEE Signal Processing Letters*, *27*, 600–604. [Cited on pages 56 and 201.]

Tanaka, D., Ikami, D., Yamasaki, T., & Aizawa, K. (2018). Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560. [Cited on pages 51 and 132.]

Tayal, K., Ghosh, R., & Kumar, V. (2020). Model-agnostic methods for text classification with inherent noise. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pp. 202–213. [Cited on page 214.]

Thomee, B., Shamma, D. A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., & Li, L.-J. (2016). YFCC100M: The New Data in Multimedia Research. *Communications of the ACM*, *59*(2), 64–73. [Cited on pages 230 and 232.]

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., & Isola, P. (2020). What makes for good views for contrastive learning? *arXiv preprint arXiv:2005.10243*. [Cited on pages 166, 170, 177, and 184.]

Tokozume, Y., Ushiku, Y., & Harada, T. (2018). Learning from between-class examples for deep sound recognition. In *International Conference on Learning Representations*. [Cited on pages 6, 30, 31, 32, and 114.]

Toth, P. & Martello, S. (1990). *Knapsack problems: Algorithms and computer implementations*. Wiley. [Cited on page 73.]

Tsipas, N., Dimoulas, C. A., Kalliris, G. M., & Papanikolaou, G. (2013). Collaborative annotation platform for audio semantics. In *Audio Engineering Society Convention 134*. Audio Engineering Society. [Cited on pages 43 and 44.]

Tsouvalas, V., Saeed, A., & Ozcelebi, T. (2021). Federated self-training for semi-supervised audio recognition. *arXiv preprint arXiv:2107.06877*. [Cited on page 91.]

Turpault, N., Serizel, R., Salamon, J., & Shah, A. P. (2019). Sound event detection in domestic environments with weakly labeled data and soundscape synthesis. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 253–257. New York University, NY, USA. [Cited on pages 5, 23, 42, 91, 92, and 96.]

Turpault, N., Serizel, R., & Vincent, E. (2020a). Limitations of weak labels for embedding and tagging. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135. IEEE. [Cited on page 87.]

Turpault, N., Serizel, R., Wisdom, S., Erdogan, H., Hershey, J. R., Fonseca, E., Seetharaman, P., & Salamon, J. (2021). Sound event detection and separation: a benchmark on DESED synthetic soundscapes. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 840–844. IEEE. [Cited on pages 184 and 237.]

Turpault, N., Wisdom, S., Erdogan, H., Hershey, J. R., Serizel, R., Fonseca, E., Seetharaman, P., & Salamon, J. (2020b). Improving sound event detection in domestic environments using sound separation. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, pp. 205–209. [Cited on pages 92, 96, 184, and 238.]

Tzanetakis, G. & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, *10*(5), 293–302. [Cited on page 20.]

Tzinis, E., Casebeer, J., Wang, Z., & Smaragdis, P. (2021). Separate but together: Unsupervised federated learning for speech enhancement from Non-IID data. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. [Cited on page 214.]

Valenti, M., Squartini, S., Diment, A., Parascandolo, G., & Virtanen, T. (2017). A convolutional neural network approach for acoustic scene classification. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1547–1554. IEEE. [Cited on page 47.]

Van Grootel, M., Andringa, T. C., & Krijnders, J. (2009). DARES-G1: Database of annotated real-world everyday sounds. In *Proceedings of the NAG/DAGA International Conference on Acoustics*. [Cited on page 43.]

Van Leeuwen, D. A. & Brümmer, N. (2007). An introduction to application-independent evaluation of speaker recognition systems. In *Speaker classification I*, pp. 330–353. Springer. [Cited on page 36.]

Vasconcelos, C., Larochelle, H., Dumoulin, V., Roux, N. L., & Goroshin, R. (2020). An effective anti-aliasing approach for residual networks. *arXiv preprint arXiv:2011.10675*. [Cited on pages 49, 106, and 116.]

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., & Belongie, S. (2017). Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 839–847. [Cited on pages 52 and 136.]

Verma, P. & Berger, J. (2021). Audio transformers: Transformer architectures for large scale audio understanding. adieu convolutions. *arXiv preprint arXiv:2105.00335*. [Cited on page 127.]

Virtanen, T., Plumbley, M. D., & Ellis, D. (2018). Introduction to sound scene and event analysis. In *Computational analysis of sound scenes and events*, pp. 3–12. Springer. [Cited on page 20.]

VLFeat.org (2020). Plotting AP and ROC curves. https://www.vlfeat.org/overview/plots-rank.html. [Cited on page 36.]

Wang, D. & Brown, G. J. (2006). *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE press. [Cited on page 30.]

Wang, L., Kawakami, K., & van den Oord, A. (2020). Contrastive predictive coding of audio with an adversary. *Proc. Interspeech 2020*, pp. 826–830. [Cited on pages 57 and 199.]

Wang, L., Luc, P., Recasens, A., Alayrac, J.-B., & Oord, A. v. d. (2021). Multimodal self-supervised learning of general audio representations. *arXiv preprint arXiv:2104.12807*. [Cited on page 216.]

Wang, L. & van den Oord, A. (2020). Multi-format contrastive learning of audio representations. In *Self-Supervised Learning for Speech and Audio Processing Workshop, NeurIPS*. [Cited on pages 6, 167, 168, 180, 182, 183, 193, 194, 199, 200, and 201.]

Wang, W. (2010). *Machine Audition: Principles, Algorithms and Systems: Principles, Algorithms and Systems*. IGI Global. [Cited on pages 2 and 17.]

Wang, X., Hua, Y., Kodirov, E., & Robertson, N. M. (2019a). IMAE for noise-robust learning: Mean absolute error does not treat examples equally and gradient magnitude's variance matters. *arXiv preprint arXiv:1903.12141*. [Cited on page 51.]

Wang, Y., Liu, W., Ma, X., Bailey, J., Zha, H., Song, L., & Xia, S.-T. (2018). Iterative learning with open-set noisy labels. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pp. 8688–8696. [Cited on page 132.]

Wang, Y., Mendez, A. E. M., Cartwright, M., & Bello, J. P. (2019b). Active learning for efficient audio annotation and classification with a large amount of unlabeled data. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 880–884. IEEE. [Cited on page 45.]

Wang, Y., Neves, L., & Metze, F. (2016). Audio-based multimedia event detection using deep recurrent neural networks. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2742–2746. IEEE. [Cited on page 2.]

Whitman, B., Flake, G., & Lawrence, S. (2001). Artist detection in music with minnowmatch. In *Neural Networks for Signal Processing XI: Proceedings of the 2001 IEEE Signal Processing Society Workshop (IEEE Cat. No. 01TH8584)*, pp. 559–568. IEEE. [Cited on page 72.]

Wisdom, S., Erdogan, H., Ellis, D. P., Serizel, R., Turpault, N., Fonseca, E., Salamon, J., Seetharaman, P., & Hershey, J. R. (2021). What's all the FUSS about free universal sound separation data? In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 186–190. IEEE. [Cited on pages 68, 91, 92, 185, and 237.]

Wisdom, S., Hershey, J. R., Wilson, K., Thorpe, J., Chinen, M., Patton, B., & Saurous, R. A. (2019). Differentiable consistency constraints for improved deep speech enhancement. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 900–904. IEEE. [Cited on page 186.]

Wisdom, S., Tzinis, E., Erdogan, H., Weiss, R. J., Wilson, K., & Hershey, J. R. (2020). Unsupervised sound separation using mixture invariant training. In *Advances in Neural Information Processing Systems*. [Cited on pages 184, 185, and 214.]

Wiskott, L. & Sejnowski, T. J. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, *14*(4), 715–770. [Cited on page 188.]

Won, M., Chun, S., Nieto, O., & Serra, X. (2020a). Data-driven harmonic filters for audio representation learning. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 536–540. IEEE. [Cited on page 31.]

Won, M., Ferraro, A., Bogdanov, D., & Serra, X. (2020b). Evaluation of CNN-based automatic music tagging models. In *Proceedings of the 17th Sound and Music Computing Conference*. [Cited on pages 20, 96, and 112.]

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G. J. (2017). Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5028–5037. [Cited on page 49.]

Wu, Z., Xiong, Y., Yu, S. X., & Lin, D. (2018). Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742. [Cited on pages 175 and 193.]

Xu, K., Cai, H., Liu, X., Gao, Z., & Zhang, B. (2017). North atlantic right whale call detection with very deep convolutional neural networks. *The Journal of the Acoustical Society of America*, *141*(5), 3944–3945. [Cited on page 2.]

Xu, Y., Li, W. J., & Lee, K. K. (2008). *Intelligent wearable interfaces*. John Wiley & Sons. [Cited on page 1.]

Yadav, S. & Foster, M. E. (2021). GISE-51: A scalable isolated sound events dataset. *arXiv preprint arXiv:2103.12306*. [Cited on page 91.]

Yamakawa, N., Kitahara, T., Takahashi, T., Komatani, K., Ogata, T., & Okuno, H. G. (2010). Effects of modelling within-and between-frame temporal variations in power spectra on non-verbal sound recognition. In *Eleventh Annual Conference of the International Speech Communication Association*. [Cited on pages xxv, 18, and 19.]

Yuan, B., Chen, J., Zhang, W., Tai, H.-S., & McMains, S. (2018). Iterative cross learning on noisy labels. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 757–765. IEEE. [Cited on page 51.]

Zeghidour, N., Teboul, O., de Chaumont Quitry, F., & Tagliasacchi, M. (2021). LEAF: A learnable frontend for audio classification. *International Conference on Learning Representations*. [Cited on pages 31 and 109.]

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, (ICLR)*. [Cited on pages 8, 50, and 129.]

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. *International Conference on Learning Representations*. [Cited on pages 31, 32, 50, 106, 113, 114, 117, 139, 141, and 142.]

Zhang, R. (2019). Making convolutional networks shift-invariant again. In *International Conference on Machine Learning*, pp. 7324–7334. PMLR. [Cited on pages 6, 49, 105, 106, 107, 109, 116, and 120.]

Zhang, Z. & Sabuncu, M. R. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *32nd Conference on Neural Information Processing Systems (NeurIPS)*. [Cited on pages 51, 132, 143, 144, 145, and 146.]

Zhang, Z. & Schuller, B. (2012). Semi-supervised learning helps in sound event classification. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 333–336. IEEE. [Cited on page 52.]

Zheng, T., Seetharaman, P., & Pardo, B. (2016). SocialFX: Studying a crowdsourced folksonomy of audio effects terms. In *Proceedings of the 2016 ACM on Multimedia Conference*, pp. 182–186. ACM. [Cited on page 44.]

Zhong, Y., Hu, Y., Huang, H., & Silamu, W. (2020). A lightweight model based on separable convolution for speech emotion recognition. In *Proc. Interspeech 2020*, vol. 11, pp. 3331–3335. [Cited on page 214.]

Zhu, B., Xu, K., Kong, Q., Wang, H., & Peng, Y. (2020). Audio tagging by cross filtering noisy labels. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *28*, 2073–2083. [Cited on page 53.]

Zinemanas, P., Rocamora, M., Fonseca, E., Font, F., & Serra, X. (2021). Toward interpretable polyphonic sound event detection with attention maps based on local prototypes. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*. [Cited on page 237.]