Universitat Autònoma de Barcelona

Towards Smart Fashion: Visual Recognition of Products and Attributes

Vacit Oğuz Yazıcı

Universitat Autònoma
de Barcelona

Towards Smart Fashion: Visual Recognition of Products and Attributes

A dissertation submitted by **Vacit Oğuz Yazıcı** to the Universitat Autònoma de Barcelona in fulfilment of the degree of **Doctor of Philosophy** in the Departament de Ciències de la Computació.

Bellaterra, October 1, 2021

Directors	**Dr. Joost van de Weijer** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Arnau Ramisa Ayats** Amazon Inc.
Thesis committee	**Dr. Francesc Moreno Noguer** Institut de Robòtica i Informàtica Industrial Universitat Politècnica de Catalunya **Dr. Xavier Roca** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Javier Romero** Facebook Reality Labs Research **Dr. Lu Yu** School of Mathematical and Computer Sciences Heriot Watt University **Dr. Bogdan Raducanu** Centre de Visió per Computador Universitat Autònoma de Barcelona

This document was typeset by the author using LaTeX 2$_\varepsilon$.

ISBN: 978-84-122714-6-1

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

The last five years have been both mentally and physically challenging. I am grateful to many people who supported me and helped me to finish this thesis, and I would like to take this opportunity to express my gratitude. First, I would like to thank to my fellow LAMP members for the support and insightful reading group sessions. I will miss our Christmas dinners and *calçotades*. I feel lucky to have met like-minded PhD students since the first day I arrived to Barcelona. For that, I would like to thank to Gemma, Arash, Dena, Onur and many others for their companionship. I am also grateful to Pau and Ali for discussions about papers and new ideas. I am very grateful to CVC staff, especially to Mireia, Claire, Gigi and Montse. Thanks to them for not getting fed up with my endless questions and always helping me. I appreciate all the efforts by my co-authors Bartek, Xialei, Lu, Abel and Luis who contributed to this thesis substantially. I would like to extend my thanks to AGAUR for their valuable support during the development of this thesis.

As an industrial PhD student, I spent most of my time in Wide Eyes Technologies over these five years. I am thankful to all my colleagues, but the research team deserves a special mention. Thanks to Marc for saving the day many times with his stellar coding skills. Kudos to Mikel for being a valuable member of the team in such a short time. Thanks to Antonio for making the office hours more fun and endless funny GIFs.

There are three people who were directly involved in my PhD journey. Firstly, thanks to my supervisor Arnau for his enormous contributions to my publications. He taught me a lot both as a supervisor and a colleague. Secondly, thanks to my boss Long, a brilliant engineer to whom I am very grateful for the invaluable experience that I gained over these five years. Finally, thanks to Joost who is the best thesis director one could ask for. The last five years would be much harder without his great expertise, guidance and tolerance for my company schedule.

Finally, I will say few words about my parents to whom this thesis is dedicated. *Canım annem, bu tezi sana ve rahmetli babama ithaf ediyorum. Bugüne kadar bana verdiğiniz her şey için teşekkür ederim. Söylemek istediklerim buraya sığamayacak kadar çok. Tek diyebileceğim, bu hayatta senin mutluluğundan daha önemli bir şey yok. Seni çok seviyorum.*

Abstract

Artificial intelligence is innovating the fashion industry by proposing new applications and solutions to the problems encountered by researchers and engineers working in the industry. In this thesis, we address three of these problems. In the first part of the thesis, we tackle the problem of multi-label image classification which is very related to fashion attribute recognition. In the second part of the thesis, we address two problems that are specific to fashion. Firstly, we address the problem of main product detection which is the task of associating correct image parts (e.g. bounding boxes) with the fashion product being sold. Secondly, we address the problem of color naming for multicolored fashion items.

The task of multi-label image classification consists in assigning various concepts such as objects or attributes to images. Usually, there are dependencies that can be learned between the concepts to capture label correlations (*chair* and *table* classes are more likely to co-exist than *chair* and *giraffe*). If we treat the multi-label image classification problem as an orderless set prediction problem, we can exploit recurrent neural networks (RNN) to capture label correlations. However, RNNs are trained to predict ordered sequences of tokens, so if the order of the predicted sequence is different than the order of the ground truth sequence, there will be penalization although the predictions are correct. Therefore, in the first part of the thesis, we propose an orderless loss function which will order the labels in the ground truth sequence dynamically in a way that the minimum loss is achieved. This results in a significant improvement of RNN models on multi-label image classification over the previous methods.

However, RNNs suffer from long term dependencies when the cardinality of set grows bigger. The decoding process might stop early if the current hidden state cannot find any object and outputs the *termination token*. This would cause the remaining classes not to be predicted and lower recall metric. Transformers can be used to avoid the long term dependency problem exploiting their self-attention modules that process sequential data simultaneously. Consequently, we propose a novel transformer model for multi-label image classification which surpasses the state-of-the-art results by a large margin.

In the second part of thesis, we focus on two fashion-specific problems. *Main*

product detection is the task of associating image parts with the fashion product that is being sold, generally using associated textual metadata (product title or description). Normally, in fashion e-commerces, products are represented by multiple images where a person wears the product along with other fashion items. If all the fashion items in the images are marked with bounding boxes, we can use the textual metadata to decide which item is the main product. The initial work treated each of these images independently, discarding the fact that they all belong to the same product. In this thesis, we represent the bounding boxes from all the images as nodes in a fully connected graph. This allows the algorithm to learn relations between the nodes during training and take the entire context into account for the final decision. Our algorithm results in a significant improvement of the state-of-the-art.

Moreover, we address the problem of color naming for multicolored fashion items, which is a challenging task due to the external factors such as illumination changes or objects that act as clutter. In the context of multi-label classification, the vaguely defined lines between the classes in the color space cause ambiguity. For example, a shade of *blue* which is very close to *green* might cause the model to incorrectly predict the color *blue* and *green* at the same time. Based on this, models trained for color naming are expected to recognize the colors and their quantities in both single colored and multicolored fashion items. Therefore, in this thesis, we propose a novel architecture with an additional head that explicitly estimates the number of colors in fashion items. This removes the ambiguity problem and results in better color naming performance.

Key words: *multi-label image classification, orderless recurrent models, transformers for multi-label classification, main product detection, graph convolutional networks, color naming*

Resum

La intel·ligència artificial innova la indústria de la moda proposant noves aplicacions i solucions als problemes als quals s'enfronten els investigadors i enginyers que treballen en la indústria. En aquesta tesi abordem tres d'aquests problemes. A la primera part de la tesi investiguem el problema de la classificació multi-etiqueta d'imatges, que està molt relacionat amb el reconeixement d'atributs de moda. A la segona part de la tesi abordem dos problemes específics de la moda. En primer lloc, abordem el problema de la detecció del producte principal, que és la tasca d'associar les parts correctes de la imatge (per exemple, delimitades mitjançant regions d'interès rectangulars) amb el producte de moda que es ven. En segon lloc, abordem el problema de el reconeixement categòric de colors per a robes multicolors.

La tasca de la classificació d'imatges multi-etiqueta consisteix a assignar diversos conceptes com objectes o atributs a una imatge. En general, es poden aprendre dependències entre els conceptes per capturar correlacions d'etiquetes (les classes *cadira* i *taula* tenen mes probabilitats de coexistir que *cadira* i *girafa*). Si tractem el problema de classificació multi-etiqueta d'imatges com un problema de predicció de conjunts de conceptes sense un ordre específic, podem aprofitar les xarxes neuronals recurrents (RNN) per capturar aquestes correlacions d'etiquetes. No obstant això, les RNN són entrenades per predir seqüències ordenades de símbols, de manera que si l'ordre de la seqüència predita és diferent a l'ordre de la seqüència de l'anotació de referència associada amb la imatge, la xarxa neuronal patirà una penalització tot i que les prediccions siguin correctes. Per tant, en la primera part de la tesi, proposem una funció objectiu per ordenar dinàmicament la seqüència d'etiquetes en l'anotació de referència de manera que s'aconsegueixi la mínima discrepància en la predicció. Això dóna com a resultat una millora significativa dels models RNN en la classificació multi-etiqueta d'imatges comparat amb els mètodes anteriors.

No obstant això, les RNN pateixen dependències a llarg termini quan la cardinalitat del conjunt augmenta. El procés de descodificació es pot aturar abans si l'estat intern actual no pot trobar cap objecte i genera el *símbol de terminació*. Això provocaria que les classes restants no es prediguessin i un percentatge menor d'identificació de conceptes. Els models *transformer* es poden utilitzar per evitar el

problema de dependència a llarg termini explotant els seus mòduls d'auto-atenció que processen seqüències completes de dades simultàniament. En conseqüència, proposem un nou model de *transformer* per a la classificació multi-etiqueta d'imatges que supera els resultats de l'estat de l'art per un ampli marge.

A la segona part de la tesi, ens enfoquem en dos problemes específics de la moda. La *detecció de producte principal* és la tasca d'associar parts de la imatge amb el producte de moda que es ven, generalment utilitzant metadades textuals associats (títol o descripció del producte). Normalment, en les botigues electròniques de moda, els productes estan representats per diverses imatges en què una persona porta el producte principal juntament amb altres peces de roba. Si totes les peces en les imatges estan anotades amb regions d'interès rectangulars, podem usar les metadades textuals per decidir quin article és el producte principal. El mètode que actualment representa l'estat de l'art en aquest camp tracta cadascuna d'aquestes imatges de forma independent, descartant que totes pertanyen al mateix producte. En aquesta tesi, representem les regions d'interès rectangulars de totes les imatges com vèrtexs en un graf completament connectat. Això permet que l'algorisme aprengui les relacions entre els vèrtexs durant l'entrenament i tingui en compte tot el context per a la decisió final. El nostre algoritme dóna com a resultat una millora significativa respecte a l'estat de l'art.

A més, abordem el problema del reconeixement categòric de colors per a robes multicolors, que és una tasca difícil a causa de factors externs com canvis d'il·luminació o oclusions causades per altres objectes. En el context de la classificació multi-etiqueta d'imatges, les fronteres difuses entre les classes en l'espai de color causen ambigüitat. Per exemple, un to de color *blau* que és molt proper al *verd* pot fer que el model predigui incorrectament els colors *blau* i *verd* al mateix temps. No obstant això, s'espera que un model intel·ligent de reconeixement categòric de colors sigui capaç a més d'encertar el nombre correcte de colors a predir en roba d'un o diversos colors. Per això, en aquesta tesi proposem una arquitectura nova amb una sortida addicional que prediu explícitament el nombre de colors en robes de moda. Això elimina el problema de l'ambigüitat i millora significativament els resultats.

Paraules clau: *classificació d'imatges multi-etiqueta, models recurrents sense ordre, transformers per a classificació de multi-etiqueta, detecció de productes principals, xarxes convolucionals de grafs, reconeixement categòric de colors*

Resumen

La inteligencia artificial innova la industria de la moda proponiendo nuevas aplicaciones y soluciones a los problemas que afrontan los investigadores e ingenieros que trabajan en la industria. En esta tesis abordamos tres de estos problemas. En la primera parte de la tesis investigamos el problema de la clasificación multi-etiqueta de imágenes, que está muy relacionado con el reconocimiento de atributos de moda. En la segunda parte de la tesis abordamos dos problemas específicos de la moda. En primer lugar, abordamos el problema de la detección del producto principal, que es la tarea de asociar las partes correctas de la imagen (por ejemplo, delimitadas mediante regiones de interés rectangulares) con el producto de moda que se vende. En segundo lugar, abordamos el problema del reconocimiento categorico de colores para ropas multicolores.

La tarea de la clasificación de imágenes multi-etiqueta consiste en asignar varios conceptos como objetos o atributos a una imagen. Por lo general, se pueden aprender dependencias entre los conceptos para capturar correlaciones de etiquetas (las clases *silla* y *mesa* tienen mas probabilidades de coexistir que *silla* y *jirafa*). Si tratamos el problema de clasificación multi-etiqueta de imágenes como un problema de predicción de conjuntos de conceptos sin un orden especifico, podemos aprovechar las redes neuronales recurrentes (RNN) para capturar dichas correlaciones de etiquetas. Sin embargo, las RNN son entrenados para predecir secuencias ordenadas de símbolos, por lo que si el orden de la secuencia predicha es diferente al orden de la secuencia de la anotación de referencia asociada con la imagen, la red neuronal sufrirá una penalización aunque las predicciones sean correctas. Por lo tanto, en la primera parte de la tesis, proponemos una función objetivo que ordenará dinámicamente la secuencia de etiquetas en la anotación de referencia de manera que se logre la mínima discrepancia con la predicción. Esto da como resultado una mejora significativa de los modelos RNN en la clasificación multi-etiqueta de imágenes con respecto a los métodos anteriores.

Sin embargo, las RNN sufren dependencias a largo plazo cuando la cardinalidad del conjunto aumenta. El proceso de decodificación puede detenerse antes si el estado interno actual no puede encontrar ningún objeto y genera el *símbolo de terminación*. Esto provocaría que las clases restantes no se predijeran y una tasa menor de identificación de conceptos. Los modelos *transformer* se pueden utilizar

para evitar el problema de dependencia a largo plazo explotando sus módulos de auto-atención que procesan secuencias completas de datos simultáneamente. En consecuencia, proponemos un modelo de *transformer* novedoso para la clasificación multi-etiqueta de imágenes que supera los resultados del estado del arte por un amplio margen.

En la segunda parte de la tesis, nos enfocamos en dos problemas específicos de la moda. La *detección de producto principal* es la tarea de asociar partes de la imagen con el producto de moda que se vende, generalmente utilizando metadatos textuales asociados (título o descripción del producto). Normalmente, en los e-commerces de moda, los productos están representados por varias imágenes en las que una persona lleva el producto principal junto con otras prendas de ropa. Si todas las prendas en las imágenes están anotadas con regiones de interés rectangulares, podemos usar los metadatos textuales para decidir qué artículo es el producto principal. El método que actualmente representa el estado del arte en este campo trata cada una de estas imágenes de forma independiente, descartando que todas pertenecen al mismo producto. En esta tesis, representamos las regiones de interés rectangulares de todas las imágenes como vértices en un grafos completamente conectado. Esto permite que el algoritmo aprenda las relaciones entre los vértices durante el entrenamiento y tenga en cuenta todo el contexto para la decisión final. Nuestro algoritmo da como resultado una mejora significativa respecto al estado del arte.

Además, abordamos el problema del reconocimiento categórico de colores para ropas multicolores, que es una tarea difícil debido a factores externos como cambios de iluminación u oclusiones causadas por otros objetos. En el contexto de la clasificación multi-etiqueta de imágenes, las fronteras difusas entre las clases en el espacio de color causan ambigüedad. Por ejemplo, un tono de color *azul* que es muy similar al color *verde* puede hacer que el modelo prediga incorrectamente los colores *azul* y *verde* al mismo tiempo. Sin embargo, se espera que un modelo inteligente de reconocimiento categórico de colores sea capaz ademas de acertar el numero correcto de colores a predecir en ropa de uno o varios colores. Por ello, en esta tesis proponemos una arquitectura novedosa con una salida adicional que predice explícitamente el número de colores en ropas de moda. Esto elimina el problema de la ambigüedad y mejora significativamente los resultados.

Palabras clave: *clasificación de imágenes multi-etiqueta, modelos recurrentes sin orden, transformers para clasificación de multi-etiqueta, detección de productos principales, redes convolucionales de grafos, reconocimiento categórico de colores*

Contents

List of Figures

List of Tables

1 Introduction

Visual perception is a human cognitive ability that enables us to interpret our surroundings. Light reflected from objects passes through our eyes and stimulates light-sensitive neurons. These signals are transmitted to the visual cortex of the brain where the signals are turned into images that we see. This ability makes us recognize static and dynamic objects around us and helps us to avoid, engage or interact with them. However, for computers, images are mathematical arrays that consist of pixels which are represented by three values whose different combinations form different colors. A simple computer can store, delete or visualize an image. However, unlike humans, it cannot directly extract any semantically meaningful information. Extracting these representations from images is the main focus of the computer vision (CV) field. Computer vision is about converting images that consist of scalar values into semantically meaningful representations. Thanks to the advances in computer vision, today we have self-driving cars, intelligent security systems that can recognize intruders, gaming console cameras that perform real-time gesture recognition, etc. We can state that, in today's world, CV-based systems are widely spread and used.

One of the sectors that uses CV-based systems is the fashion industry, which is a multi-billion dollar global enterprise. Due to the widespread use of online shopping, e-commerce retailers try to provide a smoother experience for customers who might look for a specific type of garment by displaying more content which resembles the garment of interest in category, shape, color or, in other words, in fashion attributes. Therefore, it is crucial to have fashion attribute information for all items to recommend more relevant content to a potential buyer. It is possible that, especially in case of large retailers, every day tens of thousands of new available items are added to the catalog for customers. This makes the manual process of obtaining fashion attribute information tedious. Therefore, e-commerce retailers exploit CV-based systems to extract fashion attribute information from images automatically, making the CV-based systems valuable labour-saving technology. Apart from attribute recognition systems, visual search engines that search fashion items similar to an input image or virtual dressing rooms that enable users to try on clothes virtually are other examples of CV-based systems in fashion.

Despite the prevalence of CV-based systems in fashion, there is still a long way to go. The performance of the CV-based systems still falls behind human vision.

Moreover, there are still problems in the fashion industry that can be addressed by recent advances in computer vision and machine learning techniques.

1.1 Computer Vision for Fashion

Convolutional neural networks (CNN) are a type of artificial neural networks that exploit convolutional filters for the computation of features. Although the first functional implementations were proposed in the 1990s [66, 67], due to the hardware constraints and computation intensive nature of CNNs, the computer vision community kept relying on hand crafted features for generic tasks. However, thanks to the advances in GPU technology, fast implementations of CNNs became possible. Krizhevsky et al. [62] proposed the AlexNet model which was the first GPU-based deep CNN model evaluated on the Imagenet dataset [30] and it won the ILSVRC challenge. This became a milestone and deep CNN models started to be widely used by the computer vision research community.

Early computer vision works for fashion [8, 13, 33, 37, 123] mostly focused on learning from hand crafted features such as histogram of pixels in various color spaces or from feature descriptors such as SIFT [87], HOG [28], LBP [94] and etc. The most common tasks were visual search [37, 123], fashion attribute recognition [9, 13, 132] and fashion item parsing [38]. Experiments were conducted on datasets either collected by the authors or that were made public previously [9, 38, 79, 132].

After CNNs became the new normal for the computer vision community, the number of fashion papers that tackled different computer vision tasks substantially increased due to its potential profit-making implementations in the industry. Hadi et al. [46] proposed a new task called *street-to-shop* which is about matching a real-world example of a garment (i.e. street image) to the same garment in an e-commerce image where the background is plain and white (i.e. shop image). They collected a huge dataset from several online retailers and made it public for the research community. Liang et al. [73] proposed a novel system for jointly parsing fashion images given the image-level fashion tags. However, the first full scale deep CNN model for fashion was proposed by [84]. The authors introduced the DeepFashion dataset which is annotated with fashion attributes and clothing landmarks. Moreover, they proposed a novel model, FashionNet, which jointly learns to predict attributes and landmarks. Some of its authors introduced the DeepFashion2 dataset [40] which is the enriched version of the first dataset with richer annotations such as denser landmarks and per-pixel mask annotations.

The availability of large datasets propelled research on many fashion applications. We will here mention a small selection. Wu et al. [127] proposed a novel model that leverages natural language feedback for interactive fashion item retrieving. Ru-

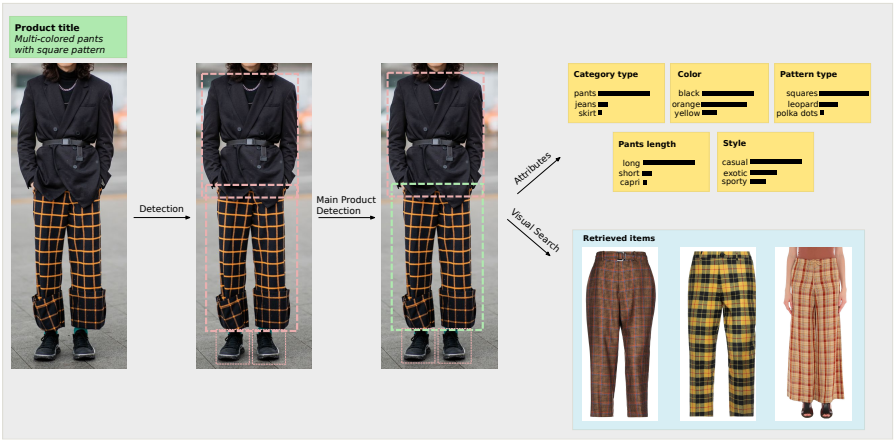


Figure 1.1 – Usual steps of a fashion product processing pipeline: fashion item detection, main product detection, attribute recognition and visual search.

bio et al. [99] introduced a unique fashion problem, main product detection, which is the task of identifying the bounding boxes that contain the product being sold in the gallery of images of the product. Li et al. [71] proposed the first end-to-end trainable outfit recommender system based on deep CNN networks. Zu et al. [142] proposed a novel model, called FashionGAN, which is a two-stage GAN framework that generates fashion outfits given the textual descriptions. Consequently, CNN based models are widely used to solve many vision based fashion problems some of which can be seen in Figure 1.1. In the following, we identify three fashion problems which will be investigated in depth in this thesis.

1.1.1 Multi-label Image Classification

In the first part of the thesis, we will focus on multi-label image classification. Multi-label image classification is the task of assigning various concepts such as object classes, attributes or tags to images. As can be seen in Figure 1.1, a fashion item might be associated with a large number of tags (e.g. pants, black, orange, pockets, long pants, cotton etc.) which makes multi-label image classification a very challenging task in computer vision for fashion. It is also clear that there are interdependencies between the labels: jeans usually have blue color or black bags usually have leather texture. Therefore, it is crucial to have a model that is able to label complex interdependencies and is robust to confusion caused by highly similar fashion categories. In that context, traditional approaches like Binary Cross-

Figure 1.2 – Traditional models for multi-label image classification might use the same visual evidence for different predictions. In the example above, they might incorrectly predict a bicycle and motorcycle at the same time using the visual evidence of hand grips and wheels.

Entropy (BCE) provide satisfactory performances, but might fail in case of high similarities between the concepts, since they do not discount evidence already used in support of another label (see Figure 1.2).

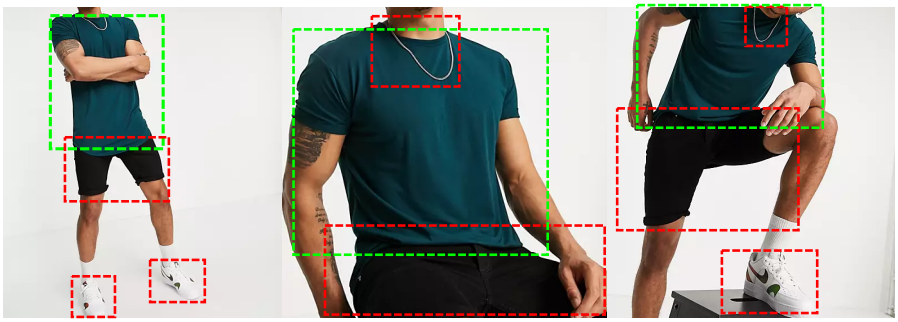
There are other types of machine learning models that do not suffer from the same problem. For example, recurrent neural networks (RNN) keep an internal state that factors in the previously predicted labels before every new one, which can be used for learning label dependencies. Therefore, they were proposed to be used for set prediction tasks like multi-label classification [107]. However, a critical limitation is that the RNN loss would penalize correct predictions if they are not generated in the same order as in the ground truth label sequence, which would make training a very accurate model impossible in practice. Several works [56, 119] tried to address this issue by imposing a fixed order. In that way, during training RNN learns to make the predictions always in the same order and in theory avoids the penalization caused by mismatched orders. Although imposing a fixed order alleviates the problem, it is far from solving it. For example, if visual evidence of an object in an image is large, the RNN may choose to predict that object in the first step. However, if the object is not the first label in the imposed order, the loss will penalize the prediction. Several works [15, 117] proposed novel methods to make RNNs works with orderless data. Vinyals et al. [117] proposed to learn the optimal order during the training without considering the predicted order. Chen et al. [15] removed the order by predicting all the labels in each time step, but that forced them to also introduce an additional module which prevents the model from generating duplicate predictions. Although these new models obtain better

results with respect to previous models, they are short of fully exploiting RNNs for multi-label image classification. *In this thesis, we propose a novel orderless loss for recurrent models to improve multi-label image classification performance.*

Although RNNs are good solutions for learning sequential data, they are known to suffer from long term dependencies. LSTMs were proposed in order to tackle this issue by [50]. Although LSTMs alleviate this issue with the proposed memory cell, they do not remove it. Therefore, RNN models in general are known to still suffer from long term dependencies especially in case of sets with high cardinalities (e.g. long sequences). Typically, RNNs generate class predictions decoding the previous hidden state and/or attention. If the decoding process is interrupted due to a bad hidden state or attention, an RNN is forced to output the *termination token* which causes that objects in not-yet attended regions are not predicted. Although this problem was never addressed directly, there were several works that proposed to replace the RNN with other models. Chen et al. [19] replaced the RNN with graph convolutional networks (GCN) to explicitly learn label dependencies. By avoiding the RNN decoding process and explicitly learning the label dependencies with GCNs, they managed to surpass the previous methods by a large margin. On the other hand, transformers have demonstrated excellent performance in sequence prediction. The proposal of replacing RNNs with self-attention layers reduced the computational complexity and path length between long term dependencies. *Consequently, in this thesis, we investigate models that prevent the pitfalls of sequence prediction, and test methods that process all data in parallel. These models can therefore perform a holistic prediction of the labels present in an image.*

1.1.2 Deep Learning for Fashion Problems

In the second part of the thesis, we will investigate two important fashion problems. Main product detection is one of the most important steps in the fashion product parsing pipeline, which was introduced by [99]. In fashion, a product (e.g. *sweater*) is represented by multiple images in a gallery, worn by a model who also happens to wear a complete outfit (e.g. *earrings*, *sweater*, *pants*, *shoes*). Main product detection is about associating correct parts of these multiple images (bounding boxes) with the product that is being sold (*sweater*), usually using the textual metadata. In the first step of the fashion product parsing pipeline, a fashion item detector is run on images to obtain the bounding boxes. Then, using the given textual metadata (product title or product description), the bounding box that bounds the main item is selected. If this process fails at some point, all future queries about the item will be wrong since the category information associated with the main product will be defective. In Figure 1.3, an example product with the main product highlighted in green can be seen.



Product title: Skinny longline t-shirt in blue

Figure 1.3 – Main product detection consists of associating the correct bounding boxes (green) with the main product given the textual metadata. If the context provided by all images is used during training, the task becomes easier since the main product (t-shirt) is more apparent and dominant than other items.

Rubio et al. [99] proposed a novel model to solve the main product detection problem. However, their model treats each bounding box that belongs to the same product independently, thus it does not take similarities and dissimilarities between the bounding boxes into account. Therefore, the model does not have the full context during training nor during evaluation which leads to inferior performance. If the relation between the images that belong to the same product was learned during training, the model would obtain better results. *Therefore, we propose a holistic main product detection method that takes into account all bounding boxes simultaneously to make the final decision in this thesis.*

As a second fashion application, we investigate color naming. Color naming is one of the most challenging vision based tasks due to reasons such as discrepancies between the physical nature of color and human perception, external factors like varying illumination and objects that act as clutter. Existing research on color in computer vision mostly focused on naming the 11 basic colors that were defined in the pioneering work of [6]. Since then color naming was addressed by the computer vision community and many novel datasets were introduced [78, 110, 111].

In fashion, the difficulty of color naming is significantly raised due to the essential extra step of segmenting the fashion item from the background or other hindrances such as human skin, hair or other fashion items (see Figure 1.4a). Moreover, most of the fashion items are multicolored which makes the task much more challenging: in order to be useful for the fashion industry, the system must return the right number of colors to describe it. Therefore, models trained for color naming

(a) (b)

Figure 1.4 – In the image on the left, predicting the color names in the cardigan is challenging due to the background and overlapping fashion items (blue turtleneck top and blue jeans pants). In the image on the right, if the model is not sure about labeling the item as *red* or *orange*, both labels might have high probabilities. In the context of multi-label classification, it can be deduced that the item is multicolored (*red* and *orange*), but actually the item is single colored.

are expected to recognize the colors (and their importance) in both single colored and multicolored fashion items. In this context, it is clear that classifying all the color palette into a predefined set of classes is not an easy task. For example, a lighter shade of the color green might look like the color blue due to the human perception or the shade being very close to blue in the color space. The color models might suffer from the same phenomenon: if we consider the same example, the model might be tricked into predicting both blue and green colors, although it is obvious that only the green color is present. This is a prevalent problem in the industry that the research community had not yet addressed. An example sample that displays this problem can be seen in Figure 1.4b. *In this thesis, we propose a novel architecture with the additional auxiliary task of explicitly estimating the number of color names for multicolored fashion items.*

1.2 Objectives and Approach

In this thesis, we aim to improve the state of the art in three areas where conventional computer vision methods fall short of providing the level of accuracy required by the fashion industry. In Chapters 2 & 3, we focus on improving multi-label clas-

sification. In addition to the fashion industry, the proposed approaches benefit a wider range of application domains that use multi-label classification. In Chapter 4, we introduce a novel model for the main product detection task that improves the state-of-the-art. Finally, in Chapter 5, we propose a novel architecture to improve color naming for multicolored fashion items.

1.2.1 Multi-label Image Classification

In this part of the thesis, we introduce novel losses and models to improve the performance of multi-label image classification models. First, we introduce a novel way of exploiting RNN models for multi-label classification. Then, to avoid the architectural limitations of RNN models, we introduce a novel visual transformer model.

Recurrent models for multi-label image classification. Recurrent models are instrumental to learn label dependencies when the input data is sequential. If we consider the multi-label classification problem as a set prediction problem, we can treat a label set as an orderless sequential data. In order to avoid penalizations caused by misalignment between the ground truth sequence and prediction sequence, several works proposed to impose a fixed order [56, 119]. The fixed order approach does not take arrangement of labels in images into account. A dominant object, which the RNN model can be inclined to predict in the first step, might be the last label in the ground truth sequence due to the fixed order. However, in this thesis, our objective is to propose a novel orderless loss which takes the natural order of labels in the image into account. In Chapter 2, we show the limitations of the fixed order approaches and common problems such as duplicate or missed predictions caused by the fixed order approaches. Our proposed recurrent models do not suffer from these limitations.

Visual transformers for multi-label image classification. Although we surpass the limitations of the fixed order approaches and achieve state-of-the-art results with orderless recurrent models, we still encounter some problems due to the recursive nature. In Chapter 3, our objective is to avoid processing data recursively to not suffer from long term dependencies and to employ a model that can handle orderless sequential data. Therefore, we propose a novel visual transformer model. We show how recurrent models fail when the cardinality of the predicted set grows higher. We also include additional novelties to the model to improve the performance.

1.2.2 Deep Learning for Fashion Problems

In this part of the thesis, we introduce novel architectures to tackle two fashion problems: main product detection and color naming. Firstly, for main product detection, we propose a novel architecture that exploits graph convolutional networks to represent all bounding boxes of a fashion product as nodes in a graph. This graphical representation allows us to take all bounding boxes into account when we make the final decision of classifying a node as positive (main product) or negative (not main product) nodes. Secondly, for color naming of multicolored fashion items, we propose a novel architecture with an additional classification head that explicitly estimates the number of colors in a fashion item alleviating the ambiguity caused by similar colors.

Main product detection. In Chapter 4, we show the limitations caused by the model that was proposed by [99]. Our objective is to show that exploiting inter-image relations between the bounding boxes that belong to the same product is more beneficial for the main product detection task. We employ graph convolutional networks to represent all bounding boxes as nodes in a fully connected graph. We propose several setups where we treat the products differently and compare them with the baselines. We display the superior results of our models on the dataset by [99] and on our newly introduced datasets.

Color naming. In Chapter 5, we introduce the problem of color naming for multi-colored fashion items which had not received much attention from the research community. Our objective is to solve the problem of inaccurate multi-label predictions caused by the uncertainty of the color model. We show that explicitly predicting the number of colors in an item alleviates this uncertainty and improves the color naming performance for multicolored fashion items. We show that our model performs almost as well as humans in this highly ambiguous and difficult task.

1.3 Industrial PhD in Wide Eyes Technologies

In this section, we describe the work conducted in Wide Eyes Technologies which is the industrial partner of this thesis. First, we briefly introduce the company. Then, in the following paragraphs, we describe the works conducted over the last five years.

Wide Eyes Technologies is a business-to-business company that uses computer vision and machine learning techniques to offer solutions for fashion companies to increase their sales. It was founded in 2013 and is based in Barcelona. The first product of the company was a mobile application called *Similify* which was used

for retrieving similar fashion products to a query item that was segmented with a contour by users on a image. After the success of the application, the company switched to developing APIs for businesses. *Shop By Image* was the first API that was developed and had similar functionalities to those of *Similify*. It was followed by the *Similar Recommendation* API, and finally the *Auto-Tagging* API which is used for fashion attribute recognition.

The work conducted as a machine learning engineer at Wide Eyes Technologies generally consisted of four responsibilities: *data management*, *R&D*, *coding & code management* and *model deployment*. In the next paragraphs, we will explain each of the responsibilities briefly.

Training deep CNN models requires large amounts of data. The performance of CNN models on the ImageNet dataset over the years shows us that, if deeper models (hence high computational expenses) are affordable, CNN models can surpass even the human performance [101] with large amounts of data. In that context, we can claim that data management is one of the the most important steps of machine learning engineering considering the high expectations by customers in the industry. In that context, Wide Eyes Technologies works with huge amounts of data. Therefore, managing and annotating these huge amounts of data are critical to the success of the deep CNN models.

Another aspect of data management is enriching the label space. When the number of clients grows bigger over the years, it is normal that some clients start to have more specific requests about the products. They might demand more classes or even more attributes to have a better correspondence between the products and their catalog. This requires the process of defining new classes or attributes considering the existent taxonomy structure. Sometimes, the newly added classes are subsets of the existing ones which requires coordinating a re-labelling of the existing data. Therefore, it is crucial to consider potential demands by the clients in the future in order to reduce the amount of work necessary for redefining the attribute structure as much as possible.

Another important part of data management is data curation. When the dataset grows bigger and gets richer with new samples and attributes, it also becomes harder to preserve the quality of labels. Due to the human errors in the annotation process, the percentage of noisy labels in the dataset might increase. Although deep CNN networks are robust to noisy labels to a certain degree, it becomes harder to maintain the performance considering complex dependencies between classes and attributes. Moreover, assuring that each category has enough samples for training (e.g. having a balanced dataset) is also essential. All of these steps have been part of the work at Wide Eyes Technologies.

The conducted R&D at Wide Eyes Technologies over five years is the bulk of the company work and this thesis. In that context, the most important part has

been finding novel ideas, empirically validating them and comparing the results with the state-of-the-art. It was essential to focus on problems that were closely related to the company work to avoid divergence between the efforts invested in the company and PhD as much as possible. Apart from that, many models with different functionalities were trained continuously. Moreover, updating and enriching our codebase with most supported libraries used in the research community was also part of the work. After the training step, using the internal tools to visualize the trained models was part of the evaluation process.

Continuously training and evaluating models with different functionalities required a considerable amount of time on programming. Every piece of code was stored in a git repository which was accessible by the research team. It was common that multiple members working on different parts of the same project at the same time. Therefore, it was essential to have sound code development skills and team coordination strategies to keep the code clean and clear for debugging. It is critical to have a bug free code to avoid having to repeat the same experiment (which might last a couple of weeks). In that context, making unit tests for our codebase was important part of the work.

Finally, deploying the trained models for production was also a fair amount of work. As expected, deploying and maintaining these models on the cloud is one of the major expenses for an AI company. Therefore, it is crucial to reduce the computation costs and memory usage as much as possible. This might require use of different frameworks for more efficient computations or architectural modifications of the models. After these improvements and modifications, it is also important to verify that the output of the models does not deviate or is within the expected output interval.

Part I
Multi-label Image Classification

2 Orderless Recurrent Models for Multi-label Classification*

2.1 Introduction

RNNs have demonstrated good performance in many tasks that require processing variable length sequential data. One of the most popular types of RNN is the Long-Short Term Memory networks (LSTM) [50]. LSTMs improve over earlier RNNs, especially addressing the vanishing gradient problem, and have advanced the state of the art in machine translation [107] and speech recognition [43], among other tasks. They have also been combined with deep Convolutional Neural Networks (CNN) and used for computer vision tasks, such as image captioning [118], and video representations [106]. Furthermore, LSTMs have been shown to be useful for traditionally non-sequential tasks, like multi-label classification [15, 56, 76, 119].

Multi-label classification is the task of assigning a wide range of visual concepts to images. These concepts could include object classes or actions, but also attributes such as colors, textures, materials, or even more abstract notions like mood. The large variety of concepts makes this a very challenging task and, to successfully address it, methods should learn the dependencies between the many concepts: boats are not common in office spaces, and penguins are seldom seen in deserts. Another problem of multi-label classification is the fact that similarities between classes may make the model uncertain about a particular object (e.g. it could be either a bicycle or a motorcycle) while being sure that both are not simultaneously present in the image. Consequently, it should choose one of the labels and not both, but traditional approaches like Binary Cross-Entropy (BCE) do not discount evidence already used in support of another label, and would predict both. In practice, these dependencies between the labels turn the task of multi-label classification into a structured labelling problem [76].

Image captioning, where the task is to generate a natural language sentence describing the image, is highly related to multi-label classification. The main difference is that in image captioning the ordering constraint imposed by the recurrent

*This chapter is based on a publication in the Conference of Computer Vision and Pattern Recognition (CVPR, 2020) [133].

Rare: *dog*
Freq: *dog, frisbee*
PLA: *dog, frisbee*

Rare: *bus, car, person*
Freq: *car, bus*
PLA: *bus, car, person*

Figure 2.1 – Estimated labels for various approaches. In the **Rare** (rare-first) approach bigger and more frequent classes might cause other classes to be ignored (*frisbee* in the left figure), meanwhile in the **Freq** (frequent-first) approach smaller frequent classes are ignored (*person* in the right figure). Our approach **PLA** circumvents these problems and correctly assigns the labels to both images.

neural network comes naturally, as sentences have a sequential nature, and RNNs are considered the appropriate model to generate an ordered list of words [118, 129]. Recently, it was found that recurrent networks also obtain good results in (orderless) structured labelling tasks like multi-label classification, and that they were good at modelling the dependencies in label space. Typically this is implemented by replacing the BCE “multi-head" of the network with an LSTM module, using it to generate a variable length sequence of labels, plus a termination token. However, this approach has a caveat: the LSTM loss will penalize otherwise correct predictions if they are not generated in the same ordering as in the ground truth label sequence. This seriously hinders convergence, complicates the training process, and generally results in inferior models.

Several recent works have tried to address this issue by imposing an arbitrary, but consistent, ordering to the ground truth label sequences [56, 119]. The rationale is that if the labels are presented always in the same order, the network will, in turn, predict them in the same order as well. Despite alleviating the problem, these approaches are short of solving it, and many of the original issues are still present. For example, in an image that features a clearly visible and prominent *dog*, the LSTM may chose to predict that label first, as the evidence for it is very large. However, if *dog* is not the label that happened to be first in the chosen ordering, the network

will be penalized for that output, and then penalized again for not predicting *dog* in the “correct" step according to the ground truth sequence. In this work, we observe that this leads to more difficult convergence, as well as sub-optimal results, like a label being predicted several times for the same image by the trained model.

In contrast with related works, we do not impose a predefined order to the output sequence, since this does not respond to any real constraint that the model should fulfill. Instead, we dynamically choose the ordering that minimizes the loss during training by re-arranging the ground truth sequence to match as closely as possible the sequence of predicted labels. We propose two ways of doing that: *predicted label alignment* (PLA) and *minimal loss alignment* (MLA). We empirically show that these approaches lead to faster training (see Figure 2.5), and also eliminate other nuisances like repeated labels in the predicted sequence. Furthermore, we obtain state-of-the-art results on the MS-COCO, WIDER Attribute and PA-100K datasets.

2.2 Related Work

Deep recurrent networks. Recurrent neural networks [100], are neural networks that include loops, and can process the same input (plus an internal state used for passing messages between iterations) several times with the same weights. The original formulation of RNNs is notoriously difficult to train because of the exploding/vanishing gradient problems, which are exacerbated with long input sequences. Later research found solutions to these problems, including the particularly successful models Gated Recurrent Unit [22] and Long-Short Term Memory [50].

Despite originally being designed for sequential data, LSTM networks have also been used for orderless data, or sets [15, 117]. Vinyals et al. [117] explores the different types of orderless data that can be processed by an LSTM network, and proposes different architectures and training procedures to deal with them. Chen et al. [15] proposes a method for order-free usage of recurrent networks for multi-label image annotation. Both these methods are discussed in more detail after we have introduced our approach to the training of orderless recurrent networks (see Section 2.3.3).

Multi-label classification. Unlike in traditional (single-label) classification, in multi-label classification each image can be associated with more than one concept. Yet, initial approaches for multi-label classification in the literature treat each occurrence of a label independently from the others [42, 126], thus not taking advantage of label correlations.

Earlier works that tried to leverage label correlations exploited graphical models

such as Conditional Random Fields (CRFs) [41] or Dependency Networks [45]. Chen et al. [14] combine CRFs with deep learning algorithms to explore dependencies between the output variables. Read et al. [98] propose using a chain of binary classifiers to do multi-label classification. Most of the approaches mentioned come with relatively high computation costs since they need to model explicitly the pairwise label correlations.

On the other hand, RNN-based multi-label classification does not incur these high computation costs, since the low dimensional RNN layers work well to model the label correlations [56, 119]. The idea to exploit RNN models to capture label correlations was originally proposed in [56] and [119]. Wang et al. [119] combine CNN and RNN architectures, and learn a joint image-label embedding space to characterize the label semantic dependencies. Since LSTMs produce sequential outputs, they use a frequent-first ordering approach. Jin et al. [56] use a CNN to encode images and input them to an RNN that generates the predictions. They use frequent-first, dictionary-order, rare-first and random order in their experiments and compare the results of different methods. Liu et al. [76] use a similar architecture, but they make the CNN and RNN models explicitly address the label prediction and label correlation tasks respectively. Instead of using a fully connected layer between the CNN and RNN models, they input class probabilities predicted by the CNN model to the RNN. In that way, they supervise both models during the training. They use rare-first ordering in their model to assign more importance to the less common labels. Chen et al. [15] use a BCE loss to compute predictions in each time step to remove the order of the labels. However, none of these approaches adapt the order dynamically according to the predictions, and they only achieve marginal improvements over CNN models. Concurrent to our work, Pineda et al. [97] present a comprehensive comparison between CNN and CNN-RNN methods on various datasets with different characteristics.

Image captioning. Earlier works on image captioning with deep learning have adapted the *encoder-decoder* framework in which an RNN (usually an LSTM) is used to “translate" image features into a sentence, one word at a time [17, 34, 36, 57, 59, 61, 88, 116, 128]. These image features are usually generated using a CNN that *encodes* (or *translates*) the image into a higher-level representation, and next an RNN *decodes* this representation back into natural language. An important part of the success of this type of models is that the whole system is trained end-to-end, and so both components can co-adapt to yield the best results. See [3, 51] for recent surveys on image caption generation.

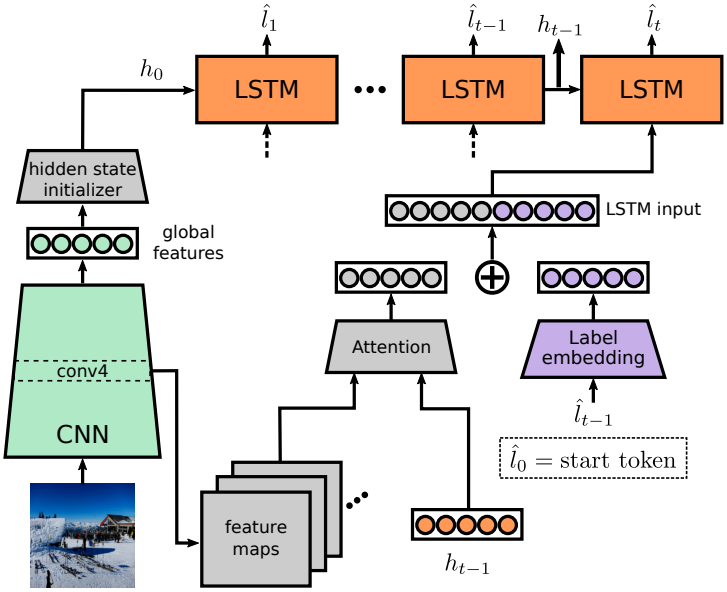


Figure 2.2 – CNN-RNN architecture used in this work, containing of an image CNN encoder, an LSTM text decoder and an attention mechanism. We show that this simple architecture can obtain state-of-the-art results by substituting the loss function by an orderless loss function.

2.3 Method

2.3.1 Image-to-sequence Model

For the task of multi-label classification we consider a CNN-RNN architecture, first proposed in [119]. This type of model consists of a CNN (encoder) part that extracts a compact visual representation from the image, and of an RNN (decoder) part that uses the encoding to generate a sequence of labels, modeling the label dependencies. Different authors experimented with different choices of visual representation to feed to the RNN: in [119], images and labels are projected to the same low-dimensional space to model the image-text relationship, while [76] uses the predicted class probabilities, and [56] experiments with different internal layers of the CNN. In our approach, we use the final fully connected layer to initialize the hidden state of the RNN. Once initialized, the RNN model predicts a new label every time step until an end signal is generated.

The choice of RNN typically used in CNN-RNN models is the Long-Short Term Memory. Unlike prior RNN models, LSTM mitigates the vanishing gradient problem by introducing a forget gate f, an input gate i and an output gate o to an RNN layer. With these gates, it can learn long term dependencies in a sequential input. The equations that govern the forward propagation through the LSTM at time step t and with an input vector x_t are the following:

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_0 x_t + U_0 h_{t-1} + b_0) \\ c_t &= f_t \odot c_{t-1} + i_t \odot tanh(W_c x_t + U_c h_{t-1} + bc) \\ ht &= o_t \odot tanh(c_t) \end{aligned} \tag{2.1}$$

where c_t and h_t are the model cell and hidden states, while i_t, f_t, o_t are the input, forget and output gates' activations respectively. W, U and b are the weights and biases to be learned, and the σ and $tanh$ are the sigmoid and hyperbolic tangent functions respectively. At time step t, the model uses as input the predicted output embedding from the previous time step. The predictions for the current time step t are computed in the following way:

$$\begin{aligned} x_t &= E \cdot \hat{l}_{t-1} \\ h_t &= \text{LSTM}(x_t, h_{t-1}, c_{t-1}) \\ p_t &= W \cdot h_t + b \end{aligned} \tag{2.2}$$

where E is a word embedding matrix and $\hat{l}_{t-1}$ is the predicted label index in the previous time step. The prediction vector is denoted by p_t, and W and b are the weights and the bias of the fully connected layer.

We also include the attention module that was proposed in [129]. Linearized activations from the fourth convolutional layer are used as input for the attention module, along with the hidden state of the LSTM at each time step, thus the attention module focuses on different parts of the image every time. These attention weighted features are then concatenated with the word embedding of the class predicted in the previous time step, and given to the LSTM as input for the current time step. As pointed out in [119], it is hard to represent small objects with global features, so an attention module alleviates the problem of ignoring smaller objects during the prediction step. A diagram of our model architecture is provided in Figure 2.2.

2.3.2 Training Recurrent Models

To train the model a dataset with pairs of images and sets of labels is used. Let (I, L) be one of the pairs containing an image I and its n labels $L = \{l_1, l_2, ..., l_n\}$, $l_i \in \mathbb{L}$, with $\mathbb{L}$ the set of all labels with cardinality $m = |\mathbb{L}|$, including the start and end tokens.

The predictions p_t of the LSTM are collected in the matrix $P = [p_1\ p_2\ ...\ p_n]$, with $P \in \mathbb{R}^{m \times n}$. When the number of predicted labels k is larger than n, we only select the first n prediction vectors. In case k is smaller than n we pad the matrix with empty vectors to obtain the desired dimensionality. We can now define the standard cross-entropy loss for recurrent models as:

$$\begin{aligned} \mathscr{L} &= tr\left(T\ log\left(P\right)\right) \\ \text{with} \quad & T_{tj} = 1 \text{ if } l_t = j \\ & T_{tj} = 0 \text{ otherwise} \end{aligned} \tag{2.3}$$

where $T \in \mathbb{R}^{n \times m}$ contains the ground truth label for each time step[†]. The loss is computed by comparing the prediction of the model at step t with the corresponding label at the same step of the ground truth sequence.

As can be seen in Equation 2.3, the order of the ground truth labels is critical to determine the loss a given prediction will receive (see Figure 2.3). For inherently orderless tasks like multi-label classification, where labels often come in random order, it becomes essential to minimize unnecessary penalization, and several approaches have been proposed in the literature. The most popular solution to improve the alignment between ground truth and predicted labels consists on defining an arbitrary criteria by which the labels will be sorted. Wang et al. [119] count occurrences of labels in the dataset and sort the labels according to their occurrence in descending order, and is consequently called the *frequent-first* approach. Jin et al. [56] use a *rare-first* approach and *dictionary-order* in addition to the frequent-first approach. Unlike the frequent-first approach, the rare-first promotes the rare classes in the dataset, while dictionary-order sorts the labels in alphabetical order. The rare-first approach was also adopted by Liu et al. [76].

Sorting the ground truth labels with a fixed, arbitrary, criteria is shown to improve results with respect to using a random ordering, since the network can learn to predict in the defined order, and avoid part of the loss. However, this will delay convergence, as the network will have to learn the arbitrary ordering in addition to predicting the correct labels given the image. Furthermore, any misalignment between the predictions and the labels will still result in higher loss and mislead-

[†]Here we consider that $l_1 = \{1, ..., m\}$ is the class-index.

Predicted labels: *person, tie*
Default order labels: *tie, person*
Loss with default order: 7.59
PLA order labels: *person, tie*
Loss with PLA: *0.04*

Predicted labels: *umbrella, person, surfboard*
Default order labels: *person, surfboard, umbrella*
Loss with default order: 6.27
PLA order labels: *umbrella, person, surfboard*
Loss with PLA: 0.87

Figure 2.3 – Comparison of an ordered loss to our orderless PLA loss. Imposing any order (default order in this example) leads to high losses even though the labels are correct. PLA solves this problem by dynamically adapting the order.

ing updates to the network. Additionally, the frequency of a label in a dataset is independent of the size of the object in a given image. Less frequent but bigger objects can cause the LSTM prediction to stop earlier because of their dominance in the image and their ranking in the prediction step. This issue can be observed in Figure 2.1, both for the frequent-first and rare-first approaches.

2.3.3 Orderless Recurrent Models

To alleviate the problems caused by imposing a fixed order to the labels, we propose to align them to the predictions of the network before computing the loss. We consider two different strategies to achieve this.

The first strategy, called *minimal loss alignment (MLA)* is computed with:

$$\begin{aligned} \mathscr{L} = \min_{T} \quad & tr\left(T \log(P)\right) \\ \text{subject to} \quad & T_{tj} \in \{0,1\}, \sum_j T_{tj} = 1, \\ & \sum_t T_{tj} = 1 \quad \forall j \in L, \\ & \sum_t T_{tj} = 0 \quad \forall j \notin L \end{aligned} \tag{2.4}$$

where $T \in \mathbb{R}^{n \times m}$ is a permutation matrix, which is constrained to have a ground truth label for each time step: $\sum_j T_{tj} = 1$, and that each label in the ground truth L should be assigned to a time step. The matrix T is chosen in such a way as to minimize the summed cross entropy loss. This minimization problem is an

assignment problem and can be solved with the Hungarian [64] algorithm.

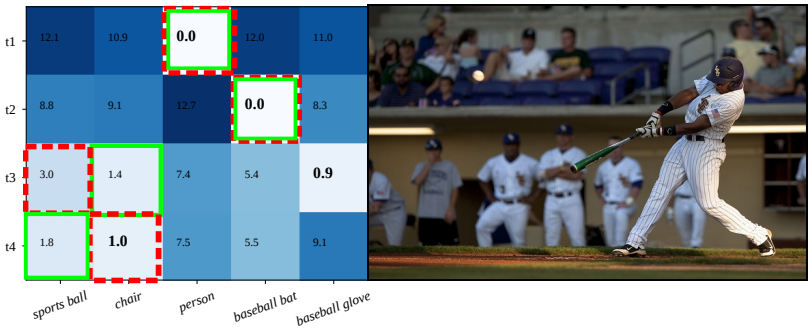
We also consider the *predicted label alignment (PLA)* solution. If we predict a label which is in the set of ground truth labels for the image, then we do not wish to change it. That leads to the following optimization problem:

$$\begin{array}{ll} \mathscr{L} = \min\limits_{T} & tr\left(T\log(P)\right) \\ \text{subject to} & T_{tj} \in \{0,1\},\ \sum_j T_{tj} = 1, \\ & T_{tj} = 1 \text{ if } \hat{l}_t \in L \text{ and } j = \hat{l}_t, \\ & \sum_t T_{tj} = 1 \ \forall\ j \in L, \\ & \sum_t T_{tj} = 0 \ \forall\ j \notin L \end{array} \tag{2.5}$$

where $\hat{l}_t$ is the label predicted by the model at step t. Here we first fix those elements in the matrix T for which we know that the prediction is in the ground truth set L, and apply the Hungarian algorithm to assign the remaining labels (with same constraints as Eq. 2.4). This second approach results in higher losses than the first one (Eq. 2.4), since there are more restrictions on matrix T. Nevertheless, this method is more consistent with the labels which were actually predicted by the LSTM.

To further illustrate our proposed approach to train orderless recurrent models we consider an example image and its cost matrix (see Figure 2.4). The cost matrix shows the cost of assigning each label to the different time steps. The cost is computed as the negative logarithm of the probability at the corresponding time step. Although the MLA approach achieves the order that yields the lowest loss, in some cases this can cause misguided gradients as it does in the example in the figure. The MLA approach puts the label *chair* in the time step t_3, although the network already predicts it in the time step t_4. Therefore, the gradients force the network to output *chair* instead of *sports ball* although *sports ball* is also one of the labels.

Orderless training of recurrent models has been previously addressed in [15, 117]. Vinyals et al. [117] study the usage of recurrent models to represent sets for which no apparent order of the elements exists. Their method considers two phases: first a uniform prior over all orders is assumed for an initial number of iterations of training, after which in the second phase ancestral sampling is used to sample an ordering. Unlike our method, which proposes to adapt the label order according to the predicted order, their method aims to find the optimal order of the labels (without considering the predicted order). Their method has only been evaluated on a toy problem. More related to our work is the research of Chen et al. [15], which applies a recurrent model without imposing any ordering. This is done by estimating all the labels in the image at every step of the recurrent model. They replace the standard cross entropy loss of the LSTM by a binary cross entropy (BCE).



PLA labels: *person*, *baseball bat*, *sports ball*, *chair* **(loss 4.50)**
MLA labels: *person*, *baseball bat*, *chair*, *sports ball* **(loss 3.63)**

Figure 2.4 – The cost matrix, image and different label orders decided by PLA and MLA (best viewed in color). Predicted classes are made bold. See text for explanation.

A drawback of this approach is that the LSTM will repeat labels already predicted before. Therefore an additional module needs to be introduced which prevents the method from repeating already predicted labels. An additional drawback of this method is that there is no end-token, so a threshold should be learned to stop the sequence.

2.4 Experiments

2.4.1 Datasets and Setting

We evaluate our models on four datasets: MS-COCO [75], NUS-WIDE [23], WIDER Attribute [70] and PA-100K [80]. **MS-COCO** is used for image segmentation, image captioning, and object detection. It can be also used for multi-label classification since it has labels for 80 objects. It consists of 82,081 training and 40,137 test images. **NUS-WIDE** consists of 269,648 images with a total number of 5,018 unique labels. However, annotations for 81 labels are more trustworthy and used for evaluation. After removing the images that do not belong to any of the 81 labels, 209,347 images remain. Following [58, 76], we use 150,000 of these images for training, and rest for testing. For a fair comparison, we create 3 different splits and we pick the best scores on each split and average them to get the final scores. **WIDER Attribute** is a dataset which has 14 human attributes in 13,789 images with 57,524 annotated bounding boxes (28,345 for training and 29,179 for test). **PA-100K** is built for evaluating a

pedestrian attribute recognition task. It consists of 100,000 pedestrian images with 26 attributes. The size of the training, validation and test sets are 80,000, 10,000 and 10,000 respectively.

Evaluation metrics. We use *per-class* and *overall* precision, recall and F1 scores. The *per-class* metric averages precision and recall scores for each class, and the geometric mean of these averaged values gives the *per-class* F1 score. In the *overall* metric, precision and recall scores are computed for all images, and the geometric mean of precision and recall gives the *overall* F1 score. Only for PA-100K dataset, instead of evaluating accuracy of each label independently, we evaluate accuracy of image-wise class predictions to be able to compare the results with other models. Next, we are interested to see if our method actually adapts dynamically the order to the image, or just learns another (more optimal) fixed order of the classes. To this end, we use an *order-rigidness* measure on the test set. For each pair of classes, two possible orderings exist (e.g. for classes A and B that would be A-B or B-A); to compute the order-rigidness, we add the number of occurrences of the most frequent order for each pair of classes in the same image, and divide it by the total number of co-occurrences of any pair. We remove all but one of every duplicate prediction without penalization. We show order-rigidness and the percentage of images with duplicate predictions in Table 2.1.

Network training. We implemented the architecture (see Figure 2.2) using the PyTorch framework [96]. For the encoder part and BCE models we use the VGG16 [105], ResNet-50 (for PA-100K) and ResNet-101 [49] architectures, and the decoder part is an LSTM with a 512 dimensional internal layer. The word embeddings learned during the training have dimension 256, and the attention module, 512. To train the BCE models, the SGD optimizer is used with learning rate 0.01 and momentum 0.9. For the LSTM models the encoder and the decoder are trained with the ADAM optimizer and Stochastic Weight Averaging [54] with a cyclical learning rate scheduler, decreasing from 10^{-3} to 10^{-6} in 3 iterations. The BCE models are trained for 40 epochs, and if no improvement is observed after 3 epochs, then we multiply the current learning rate by 0.1. For the LSTM models, we fine-tune from the best BCE model and train for 30 epochs more. All the BCE models are pretrained on ImageNet [101]. Finally, we do not use the beam search algorithm; we just greedily take the maximum predicted output. Random affine transformations and contrast changes are applied as data augmentation[‡].

[‡]Our code is available at https://github.com/voyazici/orderless-rnn-classification

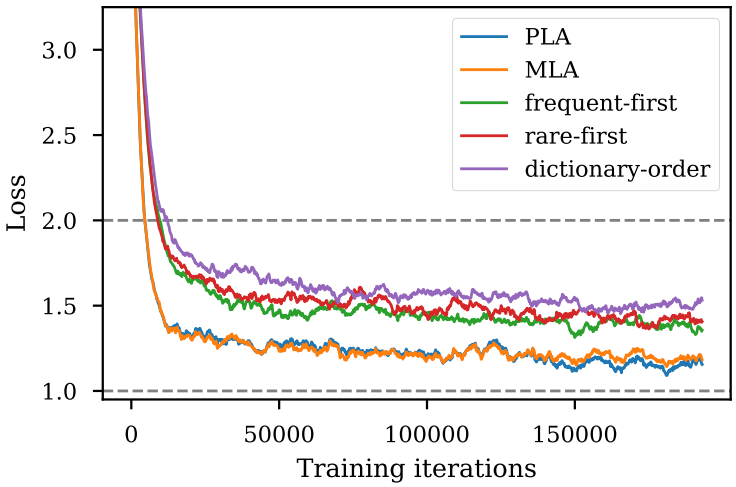


Figure 2.5 – Loss curves for different training strategies of CNN-RNN models on the MS-COCO. The graph clearly shows that our strategies, MLA and PLA, obtain significantly lower losses. PLA obtains a slightly better loss from 140,000 iterations onward. This is also reflected in the better performance of PLA for multi-label classification.

Table 2.1 – Ratio of duplicates and order-rigidness of different ordering methods on the MS-COCO validation dataset. Results show that our methods do not produce any duplicates and manage to produce label predictions with varying orders (as measured by the order-rigidness).

Algorithms	Ratio of duplicates	Order-rigidness
Random order	57.86%	67.00%
Freq. first	23.84%	100.00%
Rare-first	29.61%	100.00%
Dict. order	32.90%	100.00%
MLA	0.10%	82.87%
PLA	0.04%	80.25%

2.4.2 Comparison Ordering Approaches and Analysis

We first compare our method to other CNN-RNN optimization strategies such as frequent-first and rare-first, and evaluate several properties of the different methods. Next we compare it against the state-of-the-art on the MS-COCO, NUS-WIDE, WIDER Attribute and PA-100K datasets.

Evaluating training strategies. First, we compare the different strategies to train CNN-RNN architectures presented in literature: frequent-first [119], rare-first [56] and dictionary-order [56].

As can be seen in Figure 2.5, our proposed strategies MLA and PLA, which dynamically align the ground truth labels with the predicted labels, train faster and obtain a lower overall loss. The rare and frequent-first approaches obtain substantially higher losses. A significant part of the difference between our approaches and these baselines is that they could potentially obtain a non-zero loss on images in which the model perfectly predicts the correct classes but in the wrong order, as can be seen in Figure 2.3. For these images, the backpropagated gradient will try to force the prediction to be in the predefined order (a wasted effort in terms of improving the accuracy) despite the sub-optimality of such order for the particular image in some cases, like when the object that should be predicted first is much smaller than the other objects (see Figure 2.1).

Next, we analyze the number of duplicate labels generated by the various learning strategies (see Table 2.1). To provide a baseline reference, we also include *random order* in the table, which refers to a setting where during training the order of the ground truth labels is randomly selected for each mini-batch. The results show that our method manages to learn not to repeat labels that have already been predicted for an image. In principle, one might think, this should be easy for an LSTM to learn. However, because of the imposed order for the frequent and rare-first approaches, and the resulting confusing backpropagated gradients, the LSTM does not learn this and produces many duplicates. Note that duplicates are not penalizing the overall accuracy of the system, since we remove them in a post-processing step. We would also like to point out here that Chen et al. [15] require an explicit module for the removal of duplicates generated by their approach, while we train a model that does not generate duplicates in the first place.

In Table 2.1, we show the results for order-rigidness. They show that the methods which impose a fixed order are actually always predicting labels in that order, as indicated by a 100.00% score. Our methods obtain a 80.25% and a 82.87% score, showing that we have no fixed order for the labels and that it is dynamically adjusted to the image.

In Table 2.2, we show the efficiency of the proposed methods. They require one forward pass and then we apply the Hungarian algorithm to align LSTM predictions and labels (see Eqs. 2.4 and 2.5). PLA alignment is faster because it applies the algorithm only to the wrongly predicted labels.

Table 2.2 – Comparison of different ordering methods with average computation times per-image on MS-COCO for ResNet-101.

	Training			Test
	Forward	Alignment	Backward	
Fixed Order	6.50 ms	0 ms	14.80 ms	5.90 ms
MLA		0.80 ms (Eq. 2.4)		
PLA		0.25 ms (Eq. 2.5)		

Table 2.3 – Results of different ordering methods on MS-COCO.

Algorithms	C-P	C-R	C-F1	O-P	O-R	O-F1
BCE [119]	59.3	58.6	58.9	61.7	65.0	63.3
BCE	68.1	59.2	63.3	72.2	65.8	68.8
Freq. first	70.3	56.5	62.6	72.2	64.5	68.1
Rare-first	65.7	61.3	63.4	70.8	64.7	67.6
Dict. order	**71.0**	55.9	62.5	**74.1**	62.4	67.7
MLA	68.4	60.4	64.1	72.2	66.7	69.3
PLA	68.7	60.6	64.3	72.7	66.9	69.7
PLA (atten.)	70.2	**62.0**	**65.8**	73.8	**67.7**	**70.6**

2.4.3 Experimental Results

Comparison of different ordering methods. The results of different ordering algorithms can be seen in Table 2.3. We observed that the BCE models that we train yield much higher results than the ones cited in previous works, which were originally reported by [119] and [52] for the MS-COCO and NUS-WIDE respectively. Although it is not very clear from [119], we think that the difference between our model and theirs is that, during the training, they freeze all the layers except the last one, since when we impose the same restriction to our model, we obtain similar results. Instead, when we allow for full training of the image encoder the results improve significantly, as reported in Table 2.3. Interestingly, the fully trained BCE models obtain results similar to those of the CNN-RNN models with the same CNN module when they are trained using the rare-first or frequent-first strategies. We would like to indicate that the results reported in Table 2.3 are lower than the results that are in the other tables, since we do not exploit augmentations, train fewer epochs and use a part of training set as validation to tune the hyperparameters in these experiments.

When we compare the various strategies of alignment, we see that both our

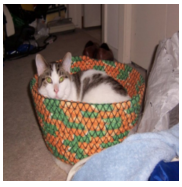
methods, MLA and PLA, clearly outperform the other strategies. Among the other approaches frequent-first yields the best result, although rare-first gives better results with the *per-class* metric since it assigns more weight to the less common classes. In Figure 2.6, we display some qualitative results for the PLA and frequent-first approaches. We can also see different orders yielded by these approaches. The images are chosen to emphasize the problems with the approaches that use predefined orders. As can be seen in the images, the frequent-first approach always predicts the labels in the same order. This leads to confusion in case of dominant but less frequent objects or minor but more frequent objects in an image. Then, this confusion leads to duplicate predictions in different time steps.

The superior performance of PLA with respect to MLA is interesting: we have empirically found that it is better to align with the actual predictions of the network than to align to obtain the minimal loss, as done in MLA. This phenomenon can be observed in Figure 2.5. Although MLA yields lower losses than PLA in the beginning, PLA's alignment with actual predictions of the network leads to a final lower loss and better accuracies. We think that so many penalizations of correctly predicted labels (see Figure 2.4) takes the optimization further away from the global minimum. In fact, the penalization rate during training is as high as 8% for some classes (e.g. baseball bat). For this reason, we select PLA as our default method instead of MLA when we compare our results with the SOTA. Finally, the attention model improves results with a significant gain.

To further investigate the advantages of the LSTM method over BCE, we compare the co-occurrence matrices of the PLA and BCE method with the co-occurrence matrix of the ground truth for the test set. The co-occurrence matrix is computed with $\sum_i l_i^T l_i$, where l_i are the ground truth labels for image i, and respectively replacing the ground truth labels by the predicted labels $\hat{l}_i$. Next, the co-occurrence matrix of the ground truth is subtracted to that of the predicted labels (the diagonals are ignored, since the co-occurrences of elements with themselves are irrelevant for our analysis). Figure 2.7 presents the co-occurrence matrices of the BCE and LSTM (PLA) models on the sports supercategory of the MS-COCO dataset. We can observe that BCE has higher co-occurrence values, and larger differences with the ground truth. Given that BCE predicts all labels independently from each other, it cannot prevent re-using evidence already used by another prediction. This can be seen, for example, in the many extra co-occurrences of predictions for skis and snowboards, as well as for sports ball and frisbee. Similar cases can be observed on the co-occurrence matrices of other supercategories in Figure 2.8. The levels of co-occurrence in BCE are noticeably higher than those on PLA, as it re-uses the same parts of image for different predictions of similar objects (e.g. bike and motorbike). This can be observed especially on the animals, food, vehicle and

Labels (freq.first): person, skis
Loss (freq. first): **0.54**
Labels (PLA): person, skis
Loss (PLA): **0.23**
Preds. freq. first: person, skis, skis
Preds. PLA: person, skis

Labels (freq.first): person, umbrella
Loss (freq. first): **3.35**
Labels (PLA): person, umbrella
Loss (PLA): **2.13**
Preds. freq. first: person, chair, bench
Preds. PLA: person, chair, umbrella



Labels (freq.first): cat
Loss (freq. first): **8.66**
Labels (PLA): cat
Loss (PLA): **1.58**
Preds. freq. first: person, cat
Preds. PLA: cat, bed

Labels (freq.first): person, couch, potted plant, remote
Loss (freq. first): **1.07**
Labels (PLA): person, remote, potted plant, couch
Loss (PLA): **0.78**
Preds. freq. first: person, couch, remote
Preds. PLA: person, remote, potted plant, couch

Labels (freq.first): person, chair, sports ball, tennis racket
Loss (freq. first): **0.36**
Labels (PLA): person, tennis racket, sports ball, chair
Loss (PLA): **0.16**
Preds. freq. first: person, chair, tennis racket, tennis racket

Labels (freq.first): teddy bear
Loss (freq. first): **1.02**
Labels (PLA): teddy bear
Loss (PLA): **0.09**
Preds. freq. first: bear
Preds. PLA: teddy bear

Labels (freq.first): dining table, pizza
Loss (freq. first): **0.89**
Labels (PLA): pizza, dining table
Loss (PLA): **0.26**
Preds. freq. first: pizza
Preds. PLA: pizza, dining table

Labels (freq.first): person, surfboard, kite
Loss (freq. first): **0.82**
Labels (PLA): person, kite, surfboard
Loss (PLA): **0.37**
Preds. freq. first: person, kite
Preds. PLA: person, kite, surfboard

Labels (freq.first): person, sports ball, tennis racket
Loss (freq. first): **0.47**
Labels (PLA): person, tennis racket, sports ball
Loss (PLA): **0.10**
Preds. freq. first: person, tennis racket
Preds. PLA: person, tennis racket, sports

Labels (freq.first): person, car, frisbee
Loss (freq. first): **0.85**
Labels (PLA): person, frisbee, car
Loss (PLA): **0.23**
Preds. freq. first: person, frisbee, frisbee
Preds. PLA: person, frisbee, car

Labels (freq.first): person, dining table, knife, tie, cake
Loss (freq. first): **0.94**
Labels (PLA): person, tie, cake, dining table, knife
Loss (PLA): **0.32**
Preds. freq. first: person, knife, knife, cake
Preds. PLA: person, tie, cake, dining table, knife

Labels (freq.first): orange, apple
Loss (freq. first): **0.57**
Labels (PLA): orange, apple
Loss (PLA): **0.37**
Preds. freq. first: orange
Preds. PLA: orange, apple

Figure 2.6 – Comparisons of orders yielded by the PLA and frequent-first approaches (wrong or duplicate predictions are underlined).

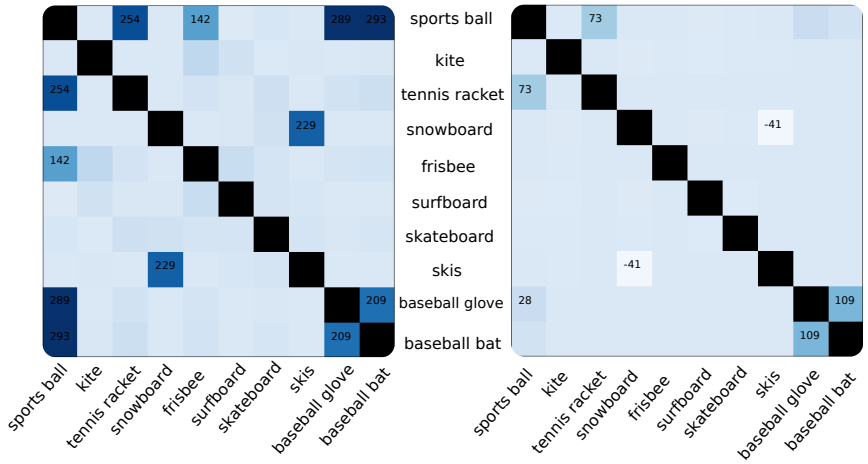


Figure 2.7 – Difference of BCE (left) and PLA model (right) co-occurrence matrices with ground truth co-occurrence matrix on the sports super-category of MS-COCO.

kitchen super-categories. In the animals super-category the BCE model overshoots co-occurrence of dogs-cats and horses-cows, while in the vehicles the confusion is on the buses, trucks and cars. In the kitchen super-category the confusion is the worst since most of the images are images of entire kitchens and the BCE model uses the entire scene for different predictions. Recurrent models, instead, naturally factor in previous predictions at every time step, which leads to a more realistic co-occurrence in the predictions.

Comparison to state-of-the-art. We compare our results with several models, grouped into two categories: models that use a CNN-RNN jointly and models that use alternative approaches. CNN-RNN [119], SR CNN-RNN [76] and Chen et al. [15] are directly related to our model (see Section 2.2 for details). Also in this category, Chen et al. [16] use an LSTM to predict the next region to attend according to the hidden state and the current region, after which they fuse the predictions of each time step. Similarly, Li et al. [69] use a recurrent network to highlight image regions to attend, but then employ reinforcement learning to select which regions should be used for the actual prediction. Among the alternative approaches, MS-CNN+LQP [92] tries to explicitly predict the number of tags in images, LSEP [72] uses a pairwise ranking approach for training a CNN, and MLIC-KD-WSD [81] use knowledge distillation from a teacher network which is trained on a weakly super-

[§]Input size of 224 × 224 (smaller than ResNet101) to compare with [15].

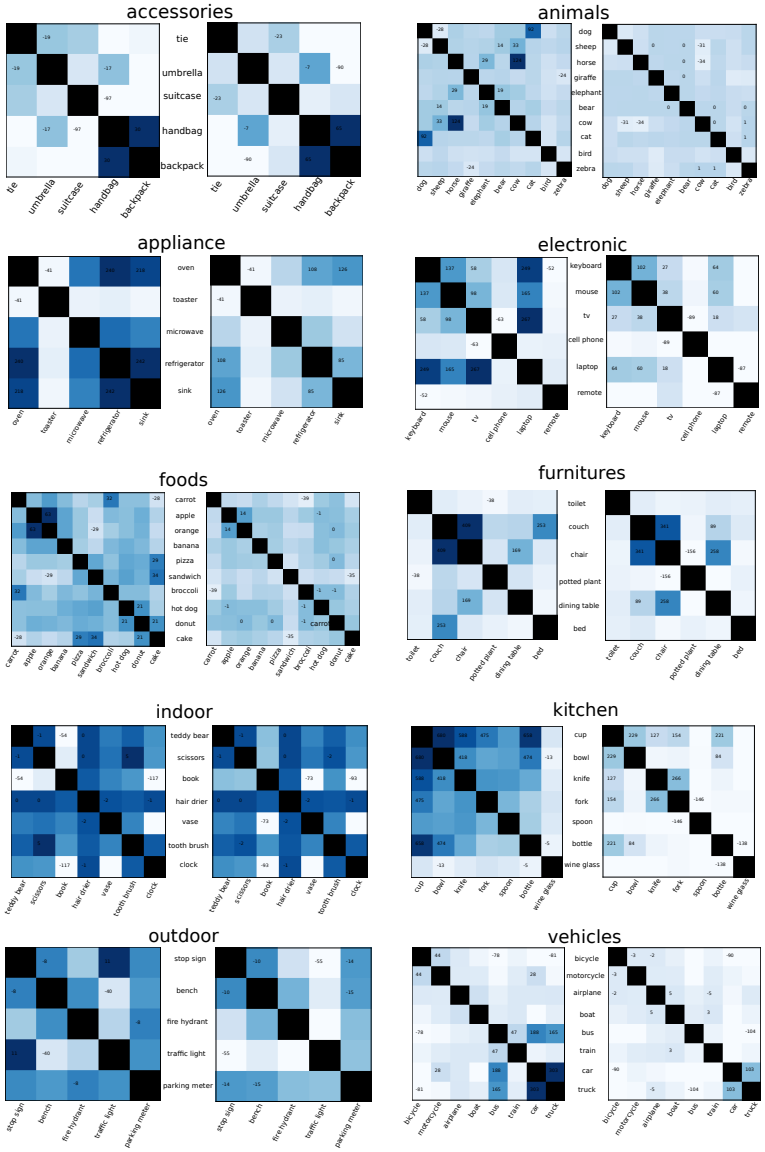


Figure 2.8 – Co-occurence matrices for BCE (left) and PLA (right) models. BCE re-uses evidence to predict different objects, and hence has higher co-occurence levels due to false positives.

Table 2.4 – Comparison with state-of-the-art on MS-COCO.

Algorithms	Architectures	C-P	C-R	C-F1	O-P	O-R	O-F1
CNN-RNN [119]	VGG16	66.0	55.6	60.4	69.2	66.4	67.8
Chen et al. [15]	ResNet152	71.6	54.8	62.1	74.2	62.2	67.7
SR CNN-RNN [76]	VGG16	67.4	59.8	63.4	76.6	68.7	72.5
Chen et al. [16]	VGG16	78.8	57.2	66.2	**84.0**	61.6	71.1
Li et al. [69]	VGG16	71.9	59.6	65.2	74.3	69.7	71.8
MS-CNN+LQP [92]	ResNet101	67.5	60.9	64.0	70.2	67.9	69.1
LSEP [72]	VGG16	73.5	56.4	63.8	76.3	61.8	68.3
MLIC-KD-WSD [81]	VGG16	-	-	69.2	-	-	74.0
SRN [141]	ResNet101	81.6	65.4	71.2	82.7	69.9	75.8
ACfs [44]	ResNet101	77.4	68.3	72.2	79.8	73.1	76.3
PLA	VGG 16	73.7	63.2	68.1	78.3	68.8	73.2
PLA	ResNet101	**80.4**	68.9	**74.2**	81.5	73.3	**77.1**
PLA	ResNet152 [§]	75.3	**69.6**	72.4	76.9	**74.0**	75.4

Table 2.5 – Comparison with state-of-the-art on NUS-WIDE.

Algorithms	C-P	C-R	C-F1	O-P	O-R	O-F1
CNN-RNN [119]	40.5	30.4	34.7	49.9	61.7	55.2
Chen et al. [15]	59.4	50.7	54.7	69.0	71.4	70.2
SR CNN-RNN [76]	55.7	50.2	52.8	70.6	71.4	71.0
Li et al. [69]	44.2	49.3	46.6	53.9	68.7	60.4
LSEP [72]	**66.7**	45.9	54.4	**76.8**	65.7	70.8
MLIC-KD-WSD [81]	-	-	**58.7**	-	-	**73.7**
PLA	60.7	**52.4**	56.2	72.0	**72.8**	72.4

vised detection task. ACfs [44], proposes a two-branch network with an original image and its transformed image as inputs and imposes an additional loss to ensure the consistency between attention heatmaps of both versions. SRN [141] proposes a Spatial Regularization Network that generates attention maps for all labels and models the label correlations via learnable convolutions. HP-Net [80] proposes a novel attention module to train multi-level and multi-scale attention-strengthened features for pedestrian analysis. The results on the mentioned datasets can be seen in Tables 2.4, 2.5, 2.6 and 2.7.[¶]

On the MS-COCO dataset, we get higher F1 scores than all other CNN-RNN models. MLIC-KD-WSD [81] also achieves notable results by exploiting knowledge

[¶]Ge et al. [39] also evaluate on COCO, however they require additional semantic maps which makes their model incomparable.

Table 2.6 – Comparison with state-of-the-art on WIDER Attribute.

Algorithms	C-P	C-R	C-F1	O-P	O-R	O-F1
SRN [141]	-	-	75.9	-	-	81.3
ACfs [44]	81.3	74.8	77.6	84.1	80.7	82.4
PLA	**81.7**	**75.9**	**78.7**	**85.0**	**81.4**	**83.1**

Table 2.7 – Comparison with state-of-the-art on PA-100K.

Algorithms	Precision	Recall	F1	Accuracy
DM [68]	82.2	80.4	81.3	70.4
HP-Net [80]	83.0	82.1	82.5	72.2
ACfs [44]	**89.0**	86.3	**87.6**	79.4
PLA	88.5	**86.7**	87.6	**79.8**

distillation from a teacher network. Only for MS-COCO, we resize input images to 288×288 to be able to compare PLA ResNet-101 model with the ACfs [44]. We also show the results for the ResNet-152 architecture to compare with [15].

On the NUS-WIDE dataset, we surpass all other CNN-RNN models by a significant margin. Our results are especially remarkable for per-class F1 score, a more relevant metric for unbalanced datasets such as this. The globally best results are achieved in terms of overall and per-class F1 score by MLIC-KD-WSD [81]. All the models that are compared on NUS-WIDE dataset use the VGG16 as the backbone network. We do not display the results of [92] since they use a different split of the dataset.

To be comparable with other models, we use ResNet-101 and ResNet-50 architectures for WIDER Attribute and PA-100K respectively. These two datasets have human attributes that are related to gender, appearance, clothing, etc. as labels. Therefore, label correlations are not common among these two datasets, which is a drawback for CNN-RNN models. In spite of that, our CNN-RNN model manages to surpass the other models.

2.5 Conclusions

We proposed an approach for training orderless LSTM models applied to multi-label classification task. Previous methods imposed an ordering on the labels to train the LSTM model, Typically frequent-first or rare-first orderings were used. Instead, we proposed two alternative losses which dynamically order the labels based on the prediction of the LSTM model. Our approach is unique in that seldomly gener-

ates any duplicate prediction, and that it minimizes the loss faster than the other methods. Results show that a standard CNN-RNN architecture, when combined with our proposed orderless loss, obtains the state-of-the-art results for multi-label classification on several datasets.

3 Visual Transformers with Primal Object Queries for Multi-label Image Classification

3.1 Introduction

The task of predicting the presence of visual concepts in images is known as multi-label classification. The visual concepts in general refer to a set of objects, but could also refer to other visual concepts such as attributes. Multi-label classification is difficult because of the wide range of classes that is typically considered, the wide variety of scales in which these classes can occur, and the complex inter-dependencies between classes [7, 125].

The field of multi-label image classification has seen much progress in recent years. Earlier works exploited graphical models to model label relations explicitly [41, 45]. Then, CNN-RNN models were proposed [76, 119] to capture label correlations, as we also investigated in Chapter 2. Later, to learn label dependencies explicitly, graph convolutional networks were proposed [44]. Even though significant progress has been made in multi-label image classification in recent years, systems still suffer from problems common to recursive methods (such as modeling long-term dependencies) or fail to capture the complex relations between the many visual concepts involved in multi-label classification.

Transformers were first proposed in [113]. Unlike recurrent models, transformers process data simultaneously. In case of recurrent models, if the decoding process is interrupted the decoder is forced to output a *termination* token, which will cause the not-yet attended classes to be missed. Due to the non-sequential nature of transformers, they do not suffer from this problem. Therefore, although they were firstly used for various NLP tasks [25, 32, 143], they became also widely used for other tasks. The first visual CNN-transformer model for a vision task was proposed for object detection [11]. Recently, a novel and pure transformer model that uses a sequence of image patches as input data was proposed for image classification [35]. Lanchantin et al. [65] proposed the first transformer model for multi-label image classification.

Carion et al. [11] introduced the concept of *object queries*. As a set of learnable positional encodings, they are added to query and key tensors in attention modules

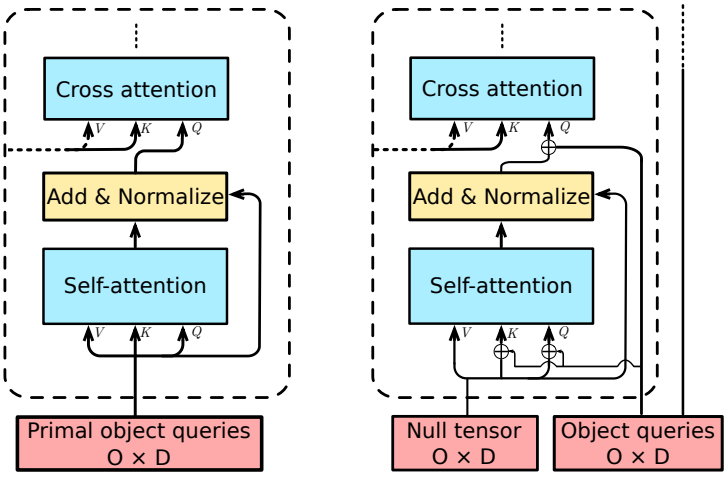


Figure 3.1 – Comparison between the primal object queries and object queries. Primal object queries are given directly to the first decoder layer instead of being added as positional encodings in every decoder layer. This leads to faster convergence and better performance.

of every decoder layer. However, using the same set of object queries in different decoder layers is not optimal for training due to the different set of relations learned by each of the layers. As we empirically demonstrate in the experimental section, it leads to lower performance and slower convergence. Therefore, in this work, we introduce *primal object queries* that differ from standard object queries in the way that they are input to a transformer decoder stack (see Figure 3.1). Moreover, we improve the mixup technique for multi-label classification to achieve significantly better results.

The main contributions in this work are:

- We introduce *primal object queries* that obtain better results and yield faster convergence than standard object queries by 79.0% and 38.6% on MS-COCO and NUS-WIDE datasets respectively.
- We improve the mixup for multi-label classification and show that it results in a significant improvement.
- Evaluation shows that we improve the state-of-the-art class wise F1 score by 2.1% and 1.8% on the MS-COCO and NUS-WIDE datasets respectively.

3.2 Related Work

Transformers. Transformers were first proposed in [113], and since became the state-of-the-art approach for sequence-to-sequence tasks such as machine translation and visual question answering. The transformer self-attention module attends all the sequential data at once, hence handling the long sequences better than RNNs, which struggle with long-term dependencies. A challenging aspect of transformers is the large number of parameters they require, which require large amounts of data to fit properly. Since such large-scale training datasets are scarce, and an important limitation to use transformer models in many practical applications, Devlin et al. [32] proposed a way of training transformers in an unsupervised manner on readily available large unsupervised text corpus, and showed that with a simple fine-tuning procedure state-of-the-art results could be achieved on various tasks. This led to a wider popularization of transformer models, and usage in areas different than the originally proposed ones, such as image captioning [25, 143] and object detection [11]. Carion et al. [11] introduced the new concept of object queries to be given input for the transformer through the training. These object queries are learned through the training, and according to the analysis done by the authors, each object query learns to focus on different areas of images and box sizes. Dosovitskiy et al. [35] proposed the first pure transformer model for image classification. They split an image into smaller patches and convert it to a sequential data (which also consists of tokens) to be processed by transformer encoder layers. Although they showed that transformers can give promising results for pure vision tasks, they still fell behind other convolutional models. Yuan et al. [136] proposed a new tokenization system that reduced the number of parameters and achieved comparable results with other convolutional models. Only very recently, Lanchantin et al. [65] introduced the first transformer model for multi-label image classification. They exploited self-attention modules to learn label dependencies for the purpose of predicting a set of labels given a set of masked label embeddings and image features.

Multi-label classification. Multi-label classification seeks to predict a variable number of labels for every single image, ideally capturing all the relevant visual concepts, such as objects or attributes, that appear on the image. Traditional methods for multi-label classification ignore label correlation, which can help boost the performance of certain under-represented classes. The few early works that tried to leverage label correlations for multi-label classification exploited graphical models such as Conditional Random Fields (CRFs) [41] or Dependency Networks [45]. More recently, the idea to exploit RNN models to capture label correlations was proposed in [56] and [119], where the low dimensional internal state of the network was used

to model label dependencies. Wang et al. [119] combined CNN and RNN architectures, and learned a joint image-label embedding space to learn label dependencies. However, since LSTMs produce sequential outputs, a fixed order was imposed to the labels, and the model learned to predict the output in the same order. This led to problems such as skipping predictions for classes that appear earlier in the sequence but less relevant in the image. In the previous chapter, we proposed a CNN-RNN model which was trained with an orderless loss function to avoid the drawbacks of imposing a fixed label order in RNN. Finally, ML-GCN [19], exploited graph convolutional networks to capture label dependencies.

Mixup. Zhang et al. [137] proposed to blend images and their associated labels randomly to improve generalization of the models. It was shown that the mixup was beneficial to avoid overconfident predictions in several tasks such as image classification [108, 137], object detection [139], text classification [108] and semantic segmentation [53]. Verma et al. [115] proposed to combine hidden states of paired samples in addition to their images and labels. Islam et al. [53] conditioned the mixup on the labels of paired samples. If the mixup is done without any constraints, then the majority of the pairs will mostly include the most frequent classes. To avoid the model to have a strong bias for frequent classes, they combined images based on a uniform distribution across categories. Wang et al. [121] is the first work that exploited the mixup technique for multi-label image classification. Although the authors did not report any improvement over the baselines in case of single models, they noted that an ensemble of models trained with mixup achieved better results.

3.3 Method

Our method for the multi-label classification is based on a transformer architecture. We try different strategies to assign labels to the decoder output. In this section, we briefly introduce some transformer concepts, explain our proposed architecture with different losses and, finally, present our different adaptations of the mixup technique.

3.3.1 Transformers

The main component of transformers is a self-attention module [113] which uses a set of weights W to compute the *query* (Q), *key* (K) and *value* (V) vectors with the input vector:

$$Q = W_Q X, \quad K = W_K X, \quad V = W_V X \tag{3.1}$$

Then, these three vectors are combined to compute the output of the self-attention layer:

$$\mathrm{A} = \mathrm{softmax}(\frac{QK^T}{\sqrt{D}})V \tag{3.2}$$

The result of the dot product between the *query* and the *key* is divided by the square root of the dimensionality D to have smoother softmax values. Many encoder layers, consisting of a self-attention and a feed-forward neural network are stacked in the encoder part of the network, and the output of the final encoder layer (A) is passed to the *cross attention module* in every decoder layer. The cross attention module has the same structure as the self attention module, with the only difference that the output of the final encoder layer is used as the key and value vectors for every decoder layer during the forward pass, while the query is obtained from the input of the decoder.

The input of the transformer decoder stack depends on the task and design of the architecture. In case of an NLP task, it might be class embeddings [143] or masked output embeddings [113]. In case of a vision-based task, it might be a set of object queries [11]. In [11], the object queries are added to the query input of each decoder layer (the query input of the first decoder layer consists of zeros as can be seen in Figure 3.1). The fact that object queries are provided to all decoder layers complicates their training: we conjecture that the requirement to be useful at multiple hierarchical levels (and therefore at different semantic levels) is hard to fulfill. In order to verify this, we conduct several experiments where we compare the approach in [11] (denominated as DETR) with inputting unique sets of object queries to each decoder layer (denominated as DETR*). The first two rows of Table 3.1, show the performance of the two models with different number of decoder layers. DETR* obtains significantly better results than DETR which confirms our assumption that inputting the same set of object queries to different decoder layers is not optimal. However, although learning a separate set of object queries for each decoder layer improves the results, it might be computationally redundant and not feasible when the number of decoder layers increases.

3.3.2 Overview of Architecture

Our architecture consists of two parts: a backbone that processes the input image, and a transformer, which can be further divided into encoder and decoder (see Figure 3.2).

Table 3.1 – Comparison of the performances of DETR, DETR* and T-POQ models with different number of decoder layers on MS-COCO.

	# of decoder layers								
	2			3			4		
	C-P	C-R	C-F1	C-P	C-R	C-F1	C-P	C-R	C-F1
DETR	75.9	64.7	69.9	74.9	64.6	69.4	**76.8**	64.2	69.9
DETR*	75.9	65.9	70.5	75.9	66.1	70.7	76.5	65.4	**70.5**
T-POQ	**76.6**	**66.0**	**70.9**	**76.5**	**66.1**	**70.9**	73.8	**67.1**	70.3

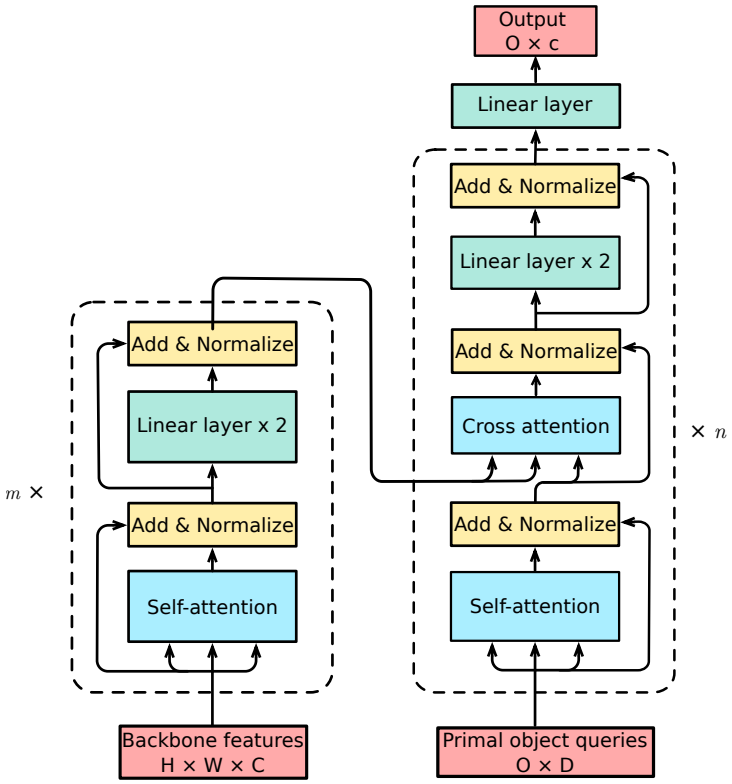


Figure 3.2 – The overall transformer architecture used in this work.

Backbone. We use a convolutional neural network to obtain a feature representation of the input image, but since the transformer requires sequential data, we

linearize the feature map of the last convolutional layer of the backbone network along the spatial dimensions, and use them as the input sequence for the encoder. More precisely, let the last convolutional layer of the CNN output a feature map of size $H \times W \times C$, where H and W are the spatial dimensions, and C the number of channels, we reshape it to $HW \times C$ to obtain a sequence of feature vectors with dimension C.

Transformer encoder and decoder. In the transformer encoder, the linearized feature map is passed through the self-attention and linear layers, and then the result is passed to the cross attention module of every decoder layer. The initial linear layer reduces the dimension to D. We do not add learnable or fixed positional encodings to encoder features since it does not give any improvements. We attribute this to using semantically strong features that do not require additional spatial information for the task of multi-label classification. We quantitatively verified that adding positional encodings is only beneficial when features from earlier layers are used, however, that also leads to lower results.

The input of the first decoder layer are *primal object queries* which differ from the object queries proposed in [11] in the way that they are given to transformer decoder stack. In our model, primal object queries are given directly to the first decoder layer instead of being added as positional encodings in every decoder layer. In this way, we avoid the drawbacks of the standard object queries which were mentioned in the previous section. In Table 3.1, we also added the results of our method (T-POQ) and it can be seen that without the overhead of learning separate object queries for each decoder layer (as does DETR*) our method obtains equal or better results. Interestingly, in the experimental section we will also show that T-POQ obtains a significant speed-up in training when compared to the DETR model. Finally, the number of primal object queries is O while the dimension of the queries is D. Then, a linear layer outputs a tensor whose shape is $O \times c$ where c is the number of classes.

Training losses. Using object queries for multi-label classification is a new concept. Previously, e.g. in CNN-RNN models, the labels are ordered either dynamically, as we do in Chapter 2, or an imposed fixed-order is applied [119]. The forward pass is done in a recursive way, therefore the length of the output tensor of the RNN is bounded by the number of labels of an image that has the most labels. However, in case of transformers, the forward pass is not done recursively which gives the flexibility of determining the size of the output tensor. We will consider two ways to train transformers for multi-label classification.

Firstly, we consider the case where we align each object query with a specific

class. In this case $O = c$ and each object query will specialize in detecting a single object class. In case of non-existent labels, *empty* tokens are assigned to the object queries that are in charge of these labels. We call this the *exhaustive model*. Here is the loss equation for the exhaustive model:

$$\begin{aligned} S_{ji} &= \frac{e^{x_{ji}}}{\sum_{i=1}^{c} e^{x_{ji}}}, \quad \mathscr{L}_{exh} = -\frac{1}{O}\sum_{j=1}^{O}\sum_{i=1}^{c} y_{ji}\log S_{ji} \\ \text{subject to} \quad & y_{ji} \in \{0,1\}, \\ & \textstyle\sum_j y_{ji} = 1 \ \forall\ i \in L, \quad \sum_j y_{ji} = 0 \ \forall\ i \notin L \end{aligned} \tag{3.3}$$

where $y_{ji} \in \mathbb{N}^{O\times c}$, x_{ji} and L are the class labels, output tensor and set of class labels, respectively. A drawback of the exhaustive approach is that it scales linearly with the number of classes, and might be infeasible for datasets with many labels. However, reducing the number of object queries requires one object query to be in charge of multiple labels, which results in an assignment problem. Therefore, we employ an orderless loss inspired by the one proposed in Equation 2.5. We call this model the *aligned model* and it no longer requires to scale linearly with the number of classes in the dataset:

$$\begin{aligned} \mathscr{L}_{align} = \min_{y} \quad & \sum_{i=1}^{c} y_{ji}\log S_{ji} \\ \text{subject to} \quad & y_{ji} \in \{0,1\}, \quad y_{ji} = 1 \text{ if } \hat{l}_j \in L \text{ and } i = \hat{l}_j, \\ & \textstyle\sum_j y_{ji} \geq 1 \ \forall\ i \in L, \quad \sum_j y_{ji} = 0 \ \forall\ i \notin L \end{aligned} \tag{3.4}$$

where $\hat{l}_j$ is the class predicted by the model at object query j. The order of the labels in y is chosen in such a way that it gives the minimum cross entropy loss. This label assignment problem can be solved with the Hungarian algorithm. In addition, we impose the alignment constraint that assigns the class $\hat{l}_j$ to object query j if $\hat{l}_j$ belongs to L. Therefore, same class may be assigned to several object queries.

3.3.3 Mixup

Mixup was proposed in [137] and was found to significantly improve results for image classification, object detection and NLP tasks [108, 137, 139]. We consider three ways of adapting the mixup technique to the training: soft, hard and restricted hard.

To train the model, a dataset with pairs of images and sets of labels is used. Let (I, T) be the pairs in a batch containing N images $I = \{i_1, i_2, ..., i_N\}$ and labels $T = \{t_1, t_2, ..., t_N\}$, $t_i \in \{0,1\}^c$ where c is the number of classes in the dataset. For the *soft mixup*, which is the original mixup as proposed in [137], we sample random

Label1: *bed, book, person*
Label2: *horse*
Mixed up: *bed, book ,person, horse*

Label1: *giraffe*
Label2: *person, surfboard*
Mixed up: *giraffe, person, surfboard*

Label1: *bus, car, traffic light*
Label2: *person, skis*
Mixed up: *bus, car, traffic light, person, skis*

Figure 3.3 – Some mixuped images from MS-COCO dataset.

weights from a beta distribution $\lambda \sim Beta(\alpha, \alpha), \alpha \in (0, \infty)$ to use them to mixup images and their associated labels:

$$\begin{aligned} i_m &= \lambda i_i + (1-\lambda) i_j \\ t_m &= \lambda t_i + (1-\lambda) t_j \end{aligned} \tag{3.5}$$

where i_i and i_j are randomly selected images, and i_m is the mixed up version.

The soft mixup makes sense for single class image classification where the last layer is typically a softmax. However, for multi-label image classification multiple labels can be present in the image. Therefore, we use the mixup proposed in [121] and denominate it as *hard mixup* where the union of labels is taken instead of the average. As proposed by the author, we alternatively enable and disable the application of mixup in every epoch and we use the ratio of $0.5 : 0.5$ for images. Some example images for hard mixup can be seen in Figure 3.3. In order to apply the mixup in all epochs and have both mixed and non-mixed images in a batch we consider the *restricted hard mixup* as a final setup, where we apply the mixup in every epoch and restrict it by applying it only to half of the images in the batch. We use the last half of the batch to mix with the first half. For instance, if we set the batch size to 4, after the mixup the images and labels become $I = \{(i_1 + i_3)/2, (i_2 + i_4)/2, i_3, i_4\}$ and $T = \{t_1 \cup t_3, t_2 \cup t_4, t_3, t_4\}$ respectively.

3.4 Experiments

Datasets and setting. We evaluate our models on MS-COCO [75] and NUS-WIDE[24] datasets. **MS-COCO** consists of 82,081 training and 40,137 test images, with labels for image segmentation, image captioning, and object detection for 80 object categories. By ignoring the bounding box annotations, it can also be used for multi-label classification. **NUS-WIDE** consists of 269,648 images with a total number of 5,018 unique labels. However, annotations for 81 labels are more trustworthy and used for evaluation. After removing the images that do not belong to any of the 81 labels, 209,347 images remain.

Evaluation metrics. We use *per-class* and *overall* precision, recall and F1 scores. The *per-class* metric averages precision and recall scores for each class, and the geometric mean of these averaged values gives the *per-class* F1 score (C-F1). In the *overall* metric, precision and recall scores are computed for all images, and the geometric mean gives the *overall* F1 score (O-F1). For the MS-COCO, we also report the mean average precision (mAP) score.

Network training. We implemented the architecture using the PyTorch framework [96]. The number of object queries is set to 25 for all datasets. The internal dimension of the transformer is 512. The backbone network, which is pre-trained on ImageNet [101], uses an SGD optimizer with learning rate 0.001 and momentum 0.9. The rest of the model is trained with the ADAM optimizer with a learning rate 0.0001 for 40 epochs. The batch size is 32, and random affine transformations and contrast changes are applied as data augmentation. The batch norm layers in the backbone are frozen during the training. The experiments are run on a 12GB GeForce GTX 1080 Ti, and it takes approximately 18 hours for both datasets when the input image size is 448×448.

3.4.1 Ablation

All the ablation studies are done on MS-COCO dataset. All the experiments are repeated three times and the final result is the average of the three experiments. For validation we use 25% of the training data and the rest as train set. The snapshot that gives the highest score on the validation set is evaluated on the test set. The input image size is 288. Unless stated otherwise, the used transformer model is the aligned model.

In Table 3.2, LSTM and transformer models with different encoder and decoder layers are compared. The LSTM model is trained with an orderless loss (see Equation 2.5) and has the same backbone and hidden size as the transformer model. From the results it can be seen that adding encoder layers does not yield any sig-

nificant improvement. When we evaluate the attention maps generated by the self-attention module in the encoder, unlike in [11], the encoder does not separate instances which is not crucial for multi-label classification unlike for object-detection. On the other hand, more decoder layers do lead to improvement in the recall metric.

Table 3.2 – Comparison of LSTM and different transformer models

Model	# of enc.	# of dec.	mAP	C-P	C-R	C-F1	O-P	O-R	O-F1
LSTM	-	-	75.3	76.2	66.7	71.1	78.8	71.3	74.9
Trans.	0	1	78.0	79.6	67.9	73.3	**81.8**	71.9	76.5
Trans.	1	1	77.6	77.9	68.8	73.1	80.2	72.6	76.2
Trans.	1	2	**78.3**	**79.6**	68.2	**73.5**	80.8	72.4	76.3
Trans.	2	2	78.2	78.2	69.3	73.5	80.2	73.1	**76.5**
Trans.	2	3	77.9	77.4	**69.6**	73.3	79.3	**73.6**	76.4

Table 3.3 – Subtraction of LSTM scores from transformer scores on different number of labels per image.

	Number of labels per image										
	1	2	3	4	5	6	7	8	9	+10	Avg
precision	2.1	1.5	1.5	2.4	1.3	1.8	1	-0.5	0.7	0.0	0.8
recall	0.5	1	1.1	1.7	1.2	1.5	1.2	2	2.5	2.1	1.7
F1	1.5	1.3	1.3	2	1.3	1.6	1.1	1.1	1.9	1.5	1.5

In Table 3.3, we compare the performance of the LSTM and transformer model (the one with one encoder and two decoder layers) on images that have different number of labels. The values are the subtraction of average LSTM scores from average transformer scores. When the number of labels increases, the transformer model misses fewer classes that leads to fewer false negatives and higher recall. This phenomenon can be observed in Figure 3.4.

In Table 3.4, aligned and exhaustive models (one encoder and two decoder layers) are compared. The results of the models are comparable. However, for the comparison with the state-of-the-art models, we will use the aligned model. It is a more compact model and the overhead cost of the alignment step (0.9 ms per image) is negligible.

In Table 3.4, different mixup setups are compared. The α value for the soft mixup is 0.4. Mixup improves the results considerably. Among all the mixup setups, the best result is obtained with the restricted hard mixup. Higher precision and mAP

Figure 3.4 – Attention maps, predictions and their respective scores by our model and a LSTM model respectively, from top to bottom. Top-2 predictions of the LSTM model are displayed for each time step. The orange labels in the white box are false negatives, while the red ones are the false positives. The LSTM model stops predicting after it outputs the *end (termination) token* at the fourth time step. Although the model is not very confident about the prediction (considering the relatively low score), the decoding process stops once the *end token* is raised and the not-yet predicted classes remain unattended. Unlike the LSTM model, our transformer-based model attends the image as a whole and is less likely to miss classes, especially when the number of labels grows.

scores show that the restricted hard mixup model has more confident predictions. We attribute the superiority of the restricted hard mixup over the soft mixup to soft labels being detrimental for modelling label correlations; and the superiority over the hard mixup to lower variance during the gradient update due to mixed and non-mixed samples in the same batch.

3.4.2 Object Queries

In Figure 3.5, we compare the performance of our primal object queries with the object queries in DETR [11]. We show the change of C-P, C-R and C-F1 metrics with different number of decoder layers for both approaches. For simplicity, we do not employ the mixup and the image size is 224×224. We set the learning rate of the backbone to 0.0001 for the DETR model to make the model convergence during training. It can be seen that our approach yields significantly higher C-F1 scores in every setup. The more decoder layers the model has, it achieves higher recall. In addition, the model with one decoder layer already performs comparable with the models that have more decoder layers. On the other hand, the model with the DETR

Table 3.4 – Comparison of exhaustive and aligned models and mixup methods.

	mAP	C-P	C-R	C-F1	O-P	O-R	O-F1
Exhaustive	78.3	77.5	69.5	73.2	79.8	73.3	76.4
Aligned	78.1	77.6	69.4	73.3	79.9	73.3	76.4
Aligned + soft mixup	78.6	79.1	**69.8**	74.2	80.9	**73.7**	77.1
Aligned + hard mixup	79.0	79.7	69.4	74.2	81.4	73.3	77.1
Aligned + restr. hard mixup	**79.6**	**80.2**	69.7	**74.6**	**82.1**	73.5	**77.5**

approach requires at least two decoder layers to achieve comparable performance. We attribute this superiority to the residual connection after the cross attention module which enables the propagation of our primal object queries to the next layer. We empirically confirm this by disabling the residual connection and obtaining the same results as the DETR model in the case that the number of decoder layers is one. The same fact causes 79.0% and 38.6% faster convergence on MS-COCO and NUS-WIDE datasets respectively. In order to calculate the convergence, we determine the epochs that the C-F1 curves stop increasing and become steady for both models. Then, we get the percentage increase of the epoch number for setups that have up to three decoder layers. Finally, we average the percentages from different setups to get the final percentage. In Figure 3.6, it can be seen that the DETR model starts to converge much later than our model (the setup with three decoder layers). Moreover, due to the lack of the propagation of the object queries, DETR model starts from a significantly lower point compared to our model. Consequently, when we compare the best models, we achieve improvements over the DETR approach by 1.2%-1.5% in all metrics

Next, we analyze what the primal object queries of the aligned model learn when they must account for multiple classes with the orderless loss. In Figure 3.7 (darker shades indicate higher values), we display the normalized counts of the predicted classes for each object query. Each object query learns to recognize a subset of classes. Since each of them can only make one prediction during one forward pass, the subsets consist of classes that are not likely to exist together. For example, the fourth object query learns to recognize *bear*, *cow*, *suitcase*, and *wine glass*.

3.4.3 Comparison with the SOTA

We compare our results with several recent models: CNN-RNN [119], SR CNN-RNN [76], Chen et al. [15], Li et al. [69] and PLA, our model proposed in Chapter 2, are models that include RNNs either to model label relations or recursively generate

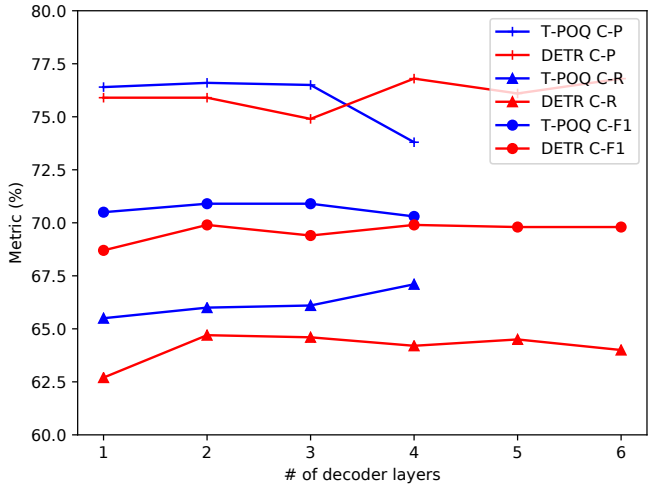


Figure 3.5 – Comparison of the proposed and DETR approaches with different number of decoder layers on MS-COCO.

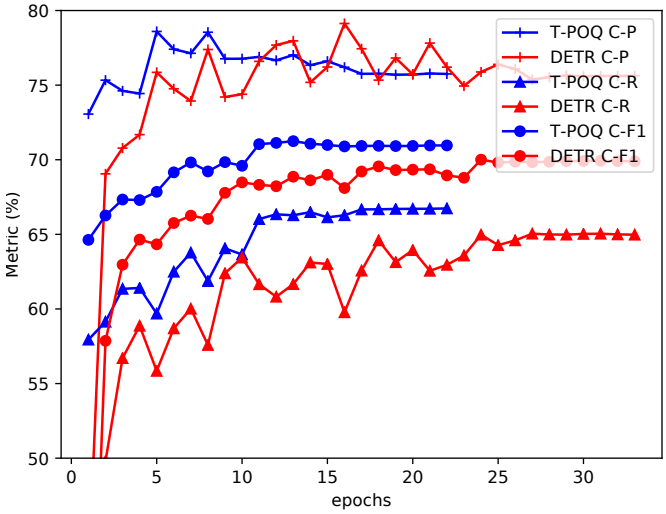


Figure 3.6 – Convergence of the proposed and DETR models on MS-COCO.

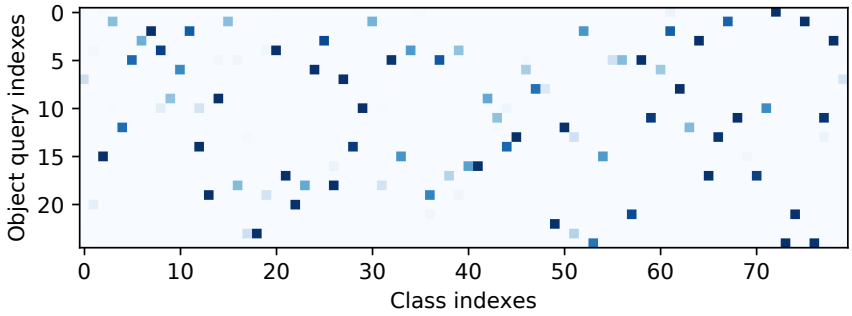


Figure 3.7 – Normalized counts of predicted classes. Each object query recognizes a subset of classes.

attention maps. SRN [141] proposes a Spatial Regularization Network that generates attention maps for all labels. ACfs [44], proposes a two-branch network with an original image and its transformed image as inputs and imposes an additional loss to ensure the consistency between attention maps of both versions. DER [20] proposes to train a detection model in three steps in which it learns class-aware attention maps, models label correlations explicitly and measures similarity between label embeddings. ML-GCN [19], exploits graph convolutional networks to capture label dependencies. C-Tran [65] is the only transformer model that we compare with. It exploits self-attention layers in a transformer encoder to learn label correlations given image features and a set of masked label embeddings and does not include any transformer decoder layer unlike our model. For MS-COCO experiments ResNet-101 architecture is used for the backbone, and for NUS-WIDE we run experiment with both ResNet101 and VGG16. For the comparison with SOTA models, we use the aligned transformer model with one encoder and two decoder layers. We also employ the restricted hard mixup.

The results on MS-COCO (see Table 3.5) show that we outperform all the state-of-the-art models. The performance superiority is more apparent in the recall metrics, since the transformer model is less likely to miss classes. The results on the NUS-WIDE dataset can be seen in Table 3.6. The results on the top part of the table use the split proposed by [58] with a VGG16 backbone, while the ones on the lower part use the original split and a ResNet-101 backbone. For the T-POQ* model, we resize the input image to 448×448 to make the comparison fair with the DER [20]. For the T-POQ model, we resize it to 224×224 to make the comparison fair with SRN [141]. We surpass all the other models, especially in the class-wise metrics which are more relevant, since NUS-WIDE is an unbalanced dataset.

Table 3.5 – Comparison with state-of-the-art on MS-COCO.

Methods	Image size	mAP	C-P	C-R	C-F1	O-P	O-R	O-F1
SRN [141]	224	77.1	81.6	65.4	71.2	82.7	69.9	75.8
ACFs [44]	288	77.5	77.4	68.3	72.2	79.8	73.1	76.3
PLA [133]	288	-	80.4	68.9	74.2	81.5	73.3	77.1
ML-GCN [19]	448	83.0	85.1	72.0	78.0	85.8	75.4	80.3
DER [20]	448	82.9	84.7	71.6	77.6	86.0	74.9	80.0
C-Tran [65]	576	85.1	**86.3**	74.3	79.9	**87.7**	76.5	81.7
T-POQ	224	77.9	79.5	67.4	73.0	81.5	71.3	76.1
T-POQ	288	80.6	80.9	70.9	75.6	82.5	74.4	78.2
T-POQ	448	84.5	82.9	75.8	79.2	84.4	78.4	81.3
T-POQ	576	**86.2**	84.1	**77.9**	**80.9**	85.0	**80.6**	**82.8**

Table 3.6 – Comparison with state-of-the-art on NUS-WIDE.

Methods	C-P	C-R	C-F1	O-P	O-R	O-F1
CNN-RNN [119]	40.5	30.4	34.7	49.9	61.7	55.2
Chen et al. [15]	59.4	50.7	54.7	69.0	71.4	70.2
SR CNN-RNN [76]	55.7	50.2	52.8	70.6	71.4	71.0
Li et al. [69]	44.2	49.3	46.6	53.9	68.7	60.4
LSEP [72]	**66.7**	45.9	54.4	**76.8**	65.7	70.8
PLA [133]	60.7	52.4	56.2	72.0	**72.8**	72.4
T-POQ	66.0	**52.7**	**58.6**	74.7	71.8	**73.2**
SRN [141]	65.2	55.8	58.5	75.5	71.5	73.4
DER [20]	64.2	**57.9**	60.9	75.5	73.0	74.2
T-POQ	66.5	56.0	60.8	75.1	73.2	74.1
T-POQ*	**66.8**	57.8	**62.0**	**75.6**	**74.3**	**74.9**

3.5 Conclusions

We introduced the *primal object queries* that differ from the standard object queries in the way that they are used. We achieved significantly better results and a large speed-up of training convergence for both MS-COCO and NUS-WIDE datasets. Our model is unique in that it achieves to learn long-term dependencies, adapts and integrates the mixup technique for multi-label classification successfully, and obtains state-of-the-art results for multi-label classification on the MS-COCO and NUS-WIDE datasets by a large margin.

Part II

Deep Learning for Fashion Problems

4 Main Product Detection with Graph Networks for Fashion

4.1 Introduction

The e-commerce market is growing every year and it is estimated that by the end of 2021 it will make for almost 18% of the total global retail sales [1]. As a consequence, investment in AI technology for fashion that improves the online consumer experience is also increasing [103]. A common problem that AI services companies operating in the fashion industry have, is accurately parsing the feeds with hundreds of thousands of products that the different clients provide as input. Although this task may seem simple at first glance, different patterns of language usage and search engine optimization (SEO) strategies by the merchants (each client can aggregate tens or hundreds of different merchants), combined with visual ambiguity in the images, make achieving industry-grade accuracy very hard. These product feeds often contain fashion products with multiple images depicting a model wearing a complete outfit, and the associated text data like product title, description or category information.

More precisely, the task of *main product detection* consists in finding all bounding boxes that contain the product being sold for an input which consists of possibly multiple gallery images combined with a product title (see Figure 4.1). Finding the main product is a crucial step in many computer vision-based fashion product processing pipelines, as all information derived from the computer vision models that analyze the images will be inaccurate otherwise. Two examples of downstream consequences are wrong category inference and visual search mismatches (e.g. showing a sweater product page when the query image is a skirt). The problem of multi-modal main product detection was defined in [99], and is related to visual grounding: a text query (i.e. product title) must be associated with corresponding parts (i.e. bounding box) in a set of gallery images. In their work, they use a contrastive loss in order to learn the representation of positive and negative image-text pairs and treat each bounding box independently, discarding the information of other bounding boxes that belong to the same product. Therefore, the model does not take similarities and dissimilarities between the bounding boxes into account neither during training nor during evaluation. In addition, we introduce the more challenging problem of *gallery-only main product detection*, where at inference time

Figure 4.1 – Fashion e-commerce sites usually showcase products with a descriptive title and a gallery of images. However, different merchants have different picture and title styles, making it difficult to define generic rules to determine which of the items displayed in the pictures is the one being sold. Therefore, algorithms that can learn this relation are of utmost interest since they would greatly reduce annotation cost.

the system has no access to the product title and has to detect the main product only based on the visual information. Although not very common, this setting arises in cases of uninformative product titles, different languages or malformed product feeds, and can lead to costly catastrophic failures if the model cannot recover from it.

In our approach, we represent bounding boxes as nodes in a densely connected graph, in which message propagation is realized between all neighbor nodes. In that way, we learn the relation between the images that belong to the same product, exploiting the context provided by all bounding boxes for the prediction (see Figure 4.2). Our model is inspired by the one proposed in [93] for visual question answering. In extensive experiments, we show that taking the context into account leads to improved performance. Especially when considering cross-dataset evaluation where we report a gain of 6-12 points and for the *Gallery-only Main Product Detection* scenario where the text input is missing, where we show that using graphs can result in a gain of up to 50 points when comparing to the same network without graphs.

This Chapter is organized as follows, in Section 4.2, we introduce the related works that focus on main product detection and incorporate graph convolutional networks for fashion applications. In Section 4.3, we explain our approach and the components of the proposed model in detail. In Section 4.4, we describe the experiments that we conduct on the datasets and the results obtained. Finally, in Section 4.5, we summarize our work and draw our main conclusions.

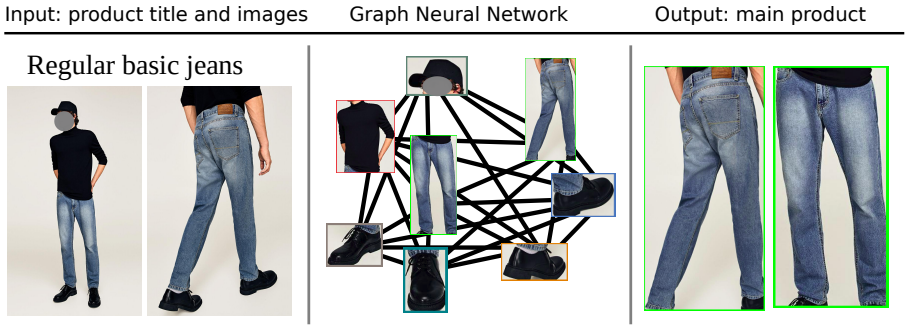


Figure 4.2 – Bounding boxes detected in all images of a product are used as nodes in a graph neural network. In this example, inter-image relations are considered for main product detection (jeans).

4.2 Related Work

The irruption of computer vision and deep learning in the fashion industry has led to many new tasks being proposed to the academic community, such as garment landmark detection [83, 122], fashion attribute recognition [40, 84], exact product retrieval [2, 46, 63] and compatibility prediction [26, 112]. In this section we review some of the works most related to ours, namely the ones that use graph convolutional networks or multi-modal embedding learning for fashion-related tasks.

Graph networks for fashion. The interest in combining convolutional networks with graph structured data became popular with spectral graph networks proposed in [10] and extended by [60] and [29]. Velivckovic et al. [114] proposed graph attention networks to exploit masked self-attentional layers to improve the previous methods. Therefore, after the graph networks became popular, new papers emerged which exploit them for traditional computer vision tasks such as image classification [19, 82], image segmentation [138], action recognition [18, 131], anomaly detection [140] etc. There are also several works using architectures that include graph neural networks for fashion. Cucurull et al. [26] propose an apparel compatibility prediction model where clothing items and their pairwise compatibility are represented as a graph, in which vertices are the clothing items and edges connect the items that are compatible. They exploit a graph neural network to predict edge connections in order to find out whether two items are compatible or not. Cui et al. [27] also propose a model for compatibility prediction with an attention mechanism. In another work [63], the authors use a graph neural network to learn similarities

between a query and catalog image in multiple scales, and the similarities are represented by the nodes of a graph that is densely connected. To the best of our knowledge graph neural networks have not been used for main product detection before.

Visual-semantic joint embedding for fashion. Paired text-image data is very common in the online fashion retail industry, and it has been naturally leveraged to train visual-semantic joint embedding networks. Han et al. [47] propose a concept discovery framework, which automatically identifies attributes derived by jointly modeling image and text. Han et al. [48], employs a bi-LSTM model to jointly learn compatibility relationships among fashion items and a visual-semantic embedding in an end-to-end framework in order to predict compatibility of fashion items and to recommend a fashion item that matches the style of an existing set. Li et al. [71], propose a CNN-RNN model to predict the popularity of a fashion set by fusing text and image features. Liao et al. [74], map fashion features and embeddings of product titles into a joint space in order to obtain meaningful representations and semantic affinities among fashion items. Transformer models have been shown to achieve excellent results in Natural Language Processing, thanks to the abundance of training data. In [4], a large dataset of product title-image pairs is used to train a transformer-based visual semantic embedding, which achieves excellent results at cross-modal retrieval.

Main product detection. As mentioned in the previous section, main product detection is a new computer vision task, proposed in [99]. Their proposed model has three main components which are the contrastive loss, the classification losses, and the word2vec model [89] that extracts the product title embeddings. The contrastive loss is used for positive and negative image-text pairs. Auxiliary classification losses for image and text are used to improve training stability and performance. To train a word2vec model, they concatenate all the available text fields in their feeds, then compute 100-dimensional descriptors for each word appearing more than 5 times. Finally, they average the descriptors to get the product title embeddings. They treat each image independently during training and evaluation which means that they do not take the relation between images that belong to a same product into account. For the rest of the Chapter, we will denominate this work as *Contrastive model*.

4.3 Method

Main product detection deals with associating correct parts of images (bounding boxes) with the given product title. As discussed before, prior work [99] considers the bounding boxes separately to decide on which of them correspondent to the

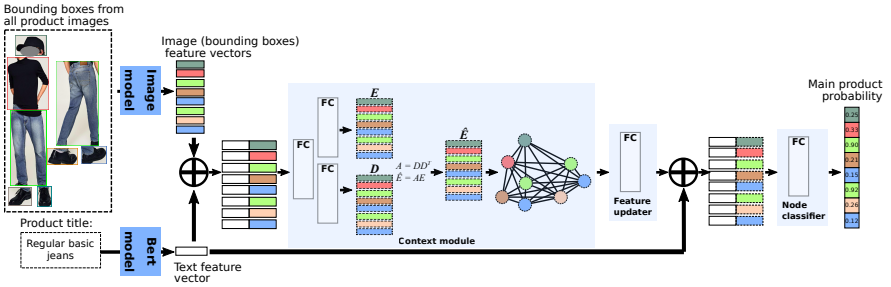


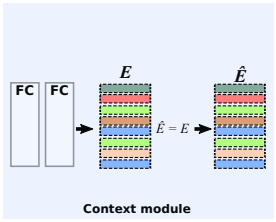
Figure 4.3 – The architecture of the model. The image features for bounding boxes from all product images are concatenated with the product title embedding. These are then used as nodes of the graph. The probability that they are the main product is estimated for each one. We also display the other variants of our model in Figure 4.4.

product title. However, it is likely that a good view of the product in one gallery image should be able to help the algorithm identify the main product in other images where it is featured less prominently. Therefore, we take a more holistic view to the problem and we want the algorithm to consider all parts in all the gallery images simultaneously.

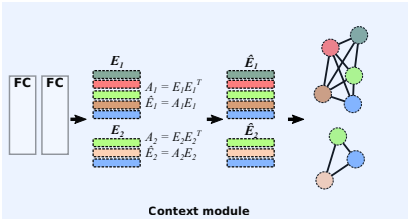
Figure 4.3 shows the architecture of our proposed model, which consists of five parts: image model, BERT (text) model [31], context module, feature updater and node classifier. The input for the BERT model are product titles, while the input for the image model are image crops corresponding to the bounding boxes. The graph in the context module is densely connected, and the nodes represent the bounding boxes found in the product gallery images. Let $G = \{V, \mathscr{E}, A\}$ be an undirected graph with self-loops, where $\mathscr{E}$ and $V \in \mathbb{R}^{N \times d}$ represent the edges and nodes respectively, and $A \in \mathbb{R}^{N \times N}$ the corresponding adjacency matrix. N and d are number of nodes and dimension of node features respectively. The idea is to learn the relations between the nodes (bounding boxes) given the title and help classify them correctly.

Image model. The Image model is a ResNet-34 [49] convolutional neural network that extracts features for each given bounding box. Activations from *layer4* are average pooled (512 dimensions) and fed to the next stage of the architecture. The model is initialized with pre-trained ImageNet weights.

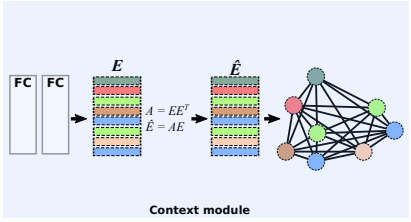
BERT model. In order to extract sentence embeddings for each title, we use a pre-trained BERT model [31]. For the dataset with the product titles in English, we use the *bert-base-uncased* BERT model, and for the one with the product titles in Turkish



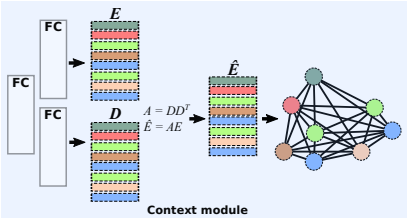
(a) NG



(b) ICFS



(c) PCFS



(d) PDFS

Figure 4.4 – The context modules of baseline and variants of our model. (a) In the no-graph model (NG) there is no graph to represent the bounding boxes as nodes as there is no interaction between the boxes. (b) In the ICFS (Instance Coupled Feature Similarity) we represent each product image as a graph. (c) In the PCFS (Product Coupled Feature Similarity) graph model, the same features are used to get the adjacency matrix and updated features. (d) The PDFS (Product Decoupled Feature Similarity) graph model, decouples the update of node feature and calculation of adjacency matrix.

we use the *bert-multilingual-cased*[*] model. We apply the BERT tokenizer which splits strings in sub-word token strings that convert them to indexes according to mappings in its vocabulary. The model outputs an embedding for each token. To extract the sentence embeddings, we use the average max pooling method (i.e. concatenation of average pooled and max pooled tokens into one vector). Since the dimensionality of the BERT models is 768, after concatenation it doubles in size and becomes 1536, so we add an extra fully connected layer to reduce the dimensionality to 512.

Context module. The main novelty of our work is the introduction of the graph network within the context module. The graph network models the interaction between the various items shown in the image gallery and the product title (see

[*]https://github.com/google-research/bert

Figure 4.4). Since the proposed graph topology is densely connected, the message passing between the nodes cannot be a simple sum of neighbor node features, as it will make all node features equal in the next layer. Therefore, we use the graph learner architecture proposed in [93], that learns the adjacency matrix for the message passing. As mentioned before, one node corresponds to each bounding box, and the edges connect every pair of nodes. We build the node features by concatenating bounding box and title embeddings, represented as $[f_n, t]$ for bounding box feature f_n and the title embedding t, and input them to the graph learner F, which consists of two fully connected layers with ReLU activation:

$$e_n = F([f_n, t]) \tag{4.1}$$

The dimensionality of $[f_n, t]$ is 1024 (512 + 512), but it is reduced back to 512 after the first layer. All N output features e_n are stacked into a matrix $E \in \mathbb{R}^{N \times P}$, where P is the dimension of the concatenated features, we compute the adjacency matrix with the following equation:

$$A = EE^T \tag{4.2}$$

which is defined as a fully connected adjacency matrix. This is not a problem computationally since the number of nodes per product is low in our problem (we will show the statistics in the datasets section). The adjacency matrix is then used for message passing before the node feature update:

$$\hat{E} = AE \tag{4.3}$$

We denominate this model as Coupled Feature Similarity (CFS). In CFS, E is used for obtaining the adjacency matrix and also as input features ($\hat{E}$) for the graph. Therefore, calculation of the adjacency and node feature update are coupled. However, we observed that using the same features E for these two purposes (i.e. pairwise similarity and node representation) may be limiting, so we propose to increase the flexibility of the model by allowing it to decouple them and learn specific representations for each of those purposes. Therefore, we test a variant of our model in which, instead of obtaining the adjacency matrix as a product of E and E^T, an additional fully connected layer (head) after the context module is used to obtain matrix $D \in \mathbb{R}^{N \times D}$ (see Figure 4.3), which is subsequently used for message passing:

$$\begin{aligned} e_n, d_n &= F([f_n, t]) \\ A &= DD^T \\ \hat{E} &= AE \end{aligned} \tag{4.4}$$

As before, all output features d_n are stacked into a matrix D. This formulation allows us to directly learn the adjacency matrix instead of extracting it from the node features. Since this model decouples the update of node feature and calculation of adjacency matrix, we denominate it as Decoupled Feature Similarity (DFS).

The baseline and variants of our model are displayed in Figure 4.4. As can be seen, we consider two setups for the CFS models: Instance Coupled Feature Similarity (ICFS) and Product Coupled Feature Similarity (PCFS). In the ICFS, we represent each product image as a graph. Because of this, and in contrast with the baseline NG model, it is allowed to take into account the context provided by the negative bounding boxes in the same image during training and evaluation. However, it does not fully exploit the relation between all bounding boxes since they are not densely connected as in the PCFS model. We do consider the connections between all bounding boxes in all images in the PCFS model.

Feature updater. The feature updater part consists of one fully connected layer and a leaky ReLU activation. We have also added these layers to the no-graph baseline model (NG) to allow for a fair comparison with the graph-based models (to ensure that they have a comparable capacity as our proposed methods).

Node classifier. The input of the node classifier is the concatenation of the original BERT embeddings and the output node features. It consists of a single fully connected layer to reduce the dimensionality to 2 (node active or inactive), and it is followed by the binary cross entropy loss during training.

4.4 Experiments

4.4.1 Datasets

We evaluated the proposed methods on two datasets whose statistics can be seen in Tables 4.1 & 4.2. We crawled each of the datasets from a different e-commerce website. We collected information related to title, description, attribute information and product images, on which we ran a fashion product detector to get bounding boxes. Finally, we used human annotators to label the ground truth main bounding boxes for each product gallery. We split the datasets and allocate 75%, 5%, 20% for training, validation and test sets respectively. Some example products can be seen in Figure 4.5. All the bounding boxes are computed with a fashion product detector pre-training.

As an extra experiment, we evaluate our models on the main bounding box detection dataset (MBBDD) which was made public by [99]. Due to the significant amount of time has passed by since the dataset was first made public, we were able to recover only a subset of the dataset. Out of total 458,700, we retrieved

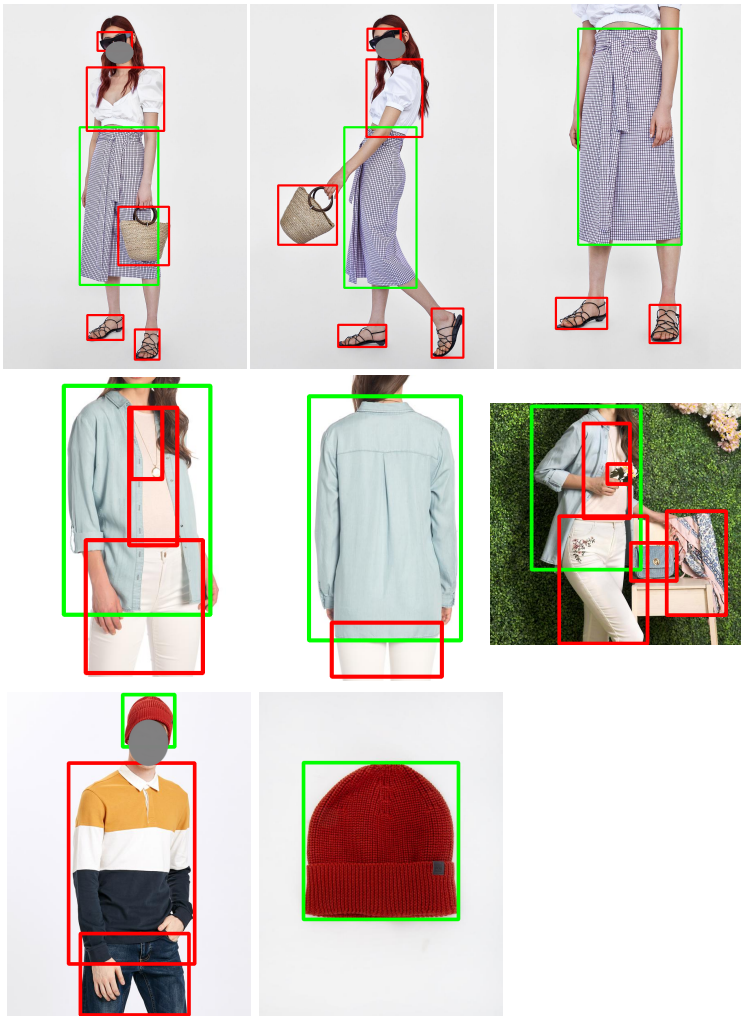


Figure 4.5 – Some multi-language example products from the dataset. The main bounding boxes are drawn in green. The titles of the products are: *Checked wrap skirt, Kadın gömlek*(*Woman shirt*) and *Triko bere*(*Knit beanie*) respectively. All the bounding boxes are computed with a fashion product detector pre-training.

Table 4.1 – Dataset statistics. BBs denotes bounding boxes.

Datasets	Lang.	Categories									Images/product	BBs/image
		accessory	bags	bottom	swim	one-piece	outerwear	shoes	sweaters	top		
1	English	236	440	4711	-	1820	2972	441	2474	6424	4.40	2.40
2	Turkish	2220	556	5183	811	1263	1190	1244	3521	6290	2.46	2.73

Table 4.2 – Number of images with M bounding boxes.

Datasets		M=1	M=2	M=3	M=4	M=5	M=6	M=7	M=8	M=9	M=10	M>10
1	Per image	33747	17590	10945	16111	5750	1410	283	32	8	3	1
	Per product	1118	992	1085	1578	977	790	738	878	974	1038	9350
2	Per image	17334	12170	6955	10360	4841	1985	660	264	92	44	92
	Per product	1980	4849	1507	1439	591	1386	865	2052	1057	1263	5289

91,550 products. The number of images per product is 1 and the average number of bounding boxes per image is 2.37. We use 77,820 products for the training and validation and the rest of them as a test set. Instead of using bounding box proposals, we use the same fashion product detector that we used for our datasets to get bounding boxes. The rest of the details about the dataset can be found in [99].

4.4.2 Evaluation Metrics

We consider the product accuracy for a single product to be 1 if all positive (product being sold) and negative (other parts of the outfit) bounding boxes are classified correctly, and 0 otherwise. Then all scores for all test images are averaged to get the final score. We deem the product accuracy metric to be the most important indicator for a main product detection system. As we explained before, one wrong bounding box classification might cause visual search mismatches in queries related to the product. Therefore, it is crucial to classify all bounding boxes of a product correctly to avoid such problems. We also consider the precision@1, recall@1 and mAP metrics. For the graph based models, we use the classification scores to rank the nodes of a product. For the contrastive model, we use the distances between image features and title embeddings.

4.4.3 Network Training

We implemented our architecture using the PyTorch framework [95] and Deep Graph Library [120]. The Adam optimizer is chosen for the training. We use learning rate 10^{-4} and 3×10^{-6} for the image and BERT models respectively. For the remaining parts of the model, the learning rate is 10^{-4}. The batch size is 6 and each batch

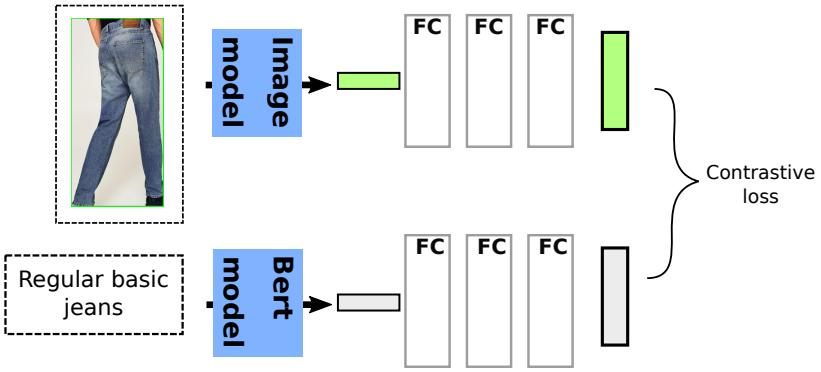


Figure 4.6 – The architecture of the contrastive model.

sample is a graph of nodes that represent the bounding boxes that belong to same products. In all experiments, we train the models for 25 epochs, and the snapshot that yields the best accuracy on the validation set is evaluated on the test set for the reported results. For the contrastive model we use batch size of 32 and train for 35 epochs. This was done to obtain competitive results compared to our methods. In the evaluation, we choose a node as a positive node if the probability of the final score is higher than 0.5. For [99], we set the margin hyper-parameter of the contrastive loss to 0.5 for training. During evaluation, we accept as main product the detections that have a cosine distance lower than 0.1 with the product title embedding. Both values were selected by cross-validation.

4.4.4 Comparison with the Baseline Models

In the initial experiments, we compare the proposed approach with a no-graph (NG) model, which contains the same layers as the proposed model (see Figures 4.3 and 4.4). The only difference is that the adjacency matrix is not used, as there is no node feature update step in the NG model. Therefore, bounding boxes cannot interact, and each decision is computed independently from the others.

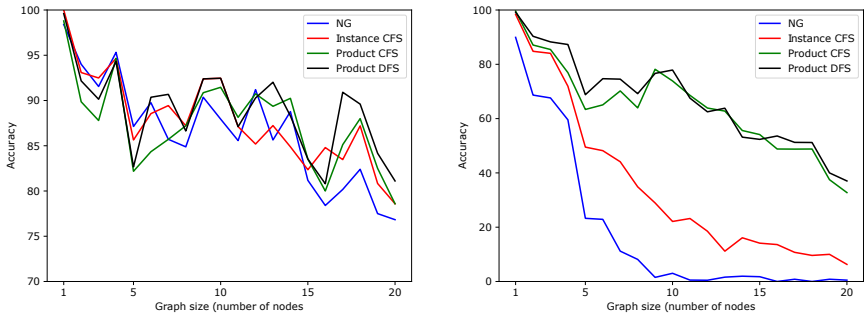
Our second baseline model is the *Contrastive model* [99], where the authors propose to map the image and text embeddings into a common space, and reduce the distances between positive bounding boxes and their titles with a contrastive loss, as well as including additional auxiliary losses for bounding box and text classification. To make the models comparable, we make sure that the image and text branches have the same architectures, we include the extra fully connected layers in the other

Table 4.3 – Performance comparison of the baselines and graph-based approaches.

Train	Test	Models	P@1	R@1	mAP	Prod. acc.
1	1	Contrastive	98.7	32.2	99.1	81.0
		NG	99.3	32.4	99.5	87.1
		ICFS	**99.3**	**32.4**	**99.5**	88.1
		PCFS	98.6	31.6	99.1	87.6
		PDFS	99.1	32.1	99.4	**89.1**
	2	Contrastive	97.1	44.8	98.1	84.3
		NG	**98.1**	**45.2**	98.4	84.7
		ICFS	97.7	45.0	98.1	87.8
		PCFS	96.5	44.7	97.6	87.1
		PDFS	97.7	45.1	**98.5**	**90.3**
2	1	Contrastive	92.7	30.5	93.4	41.1
		NG	95.4	31.0	93.9	43.2
		ICFS	96.0	31.1	94.7	53.7
		PCFS	**96.1**	**31.4**	**95.8**	51.0
		PDFS	93.1	30.4	94.5	**55.7**
	2	Contrastive	99.2	45.8	99.4	92.0
		NG	99.6	46.0	99.6	94.7
		ICFS	**99.7**	**46.1**	**99.6**	94.5
		PCFS	99.5	46.0	99.6	94.9
		PDFS	99.5	46.0	**99.6**	**95.6**

parts of the model, and remove every loss apart from the contrastive loss. Since we cannot concatenate features and embeddings as we do in our proposed model, we create two branches for image features and text embeddings after the image and BERT models (see Figure 4.6). Then, we compare our graph-based approaches: ICFS, PCFS nad PDFS. To make the comparison fair with the other graph-based models, we evaluate the ICFS model by checking the image score (which is 1 if all bounding boxes of an image are classified correctly 0 otherwise) and assigning 1 to product score if all image scores are 1.

The results are summarized in Table 4.3. We first focus on the in-dataset evaluation, referring to the results where train and test set originate form the same dataset. As can be seen the graph-based methods outperform the baselines in the product accuracy metric by a significant margin. Especially our PDFS model manages to obtain good results in the product accuracy metric, outperforming the other graph-based methods and the NG baseline. Since the average graph size is bigger for dataset 1 (see Tables 4.1 and 4.2), the gain with the graph-based



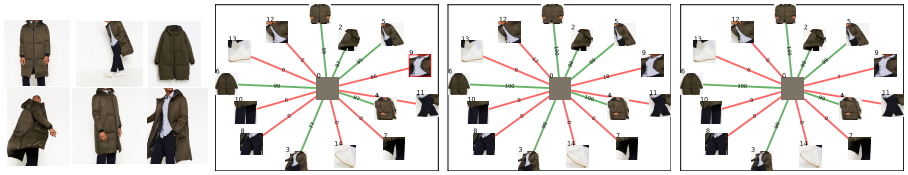
(a) Main Product Detection (b) Gallery-only Main Product Detection

Figure 4.7 – Comparison of accuracies of different models with changing graph sizes on the dataset 1. Our proposed method especially improves results when the gallery of images contains many bounding boxes.

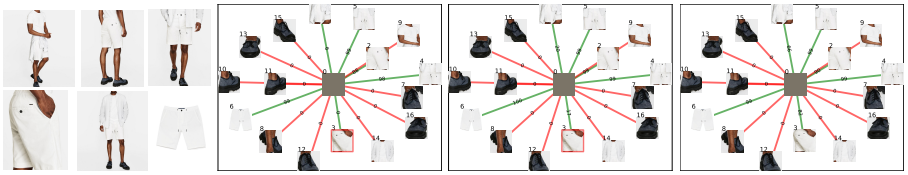
models is higher for the dataset 1. Precision@1 and recall@1 metrics yielded by the graph-based and baseline models are comparable, because it is relatively easy task to sort the bounding boxes by similarity since the number of nodes per product is low. However, in most of the metrics, our graph-based models obtain better scores. The change in performance with the graph size is further analyzed in Figure 4.7a. All the graphs whose size is bigger than 20, are represented as their size is 20 in the figure. As expected, graph-based approaches can handle larger graphs better than the non-graph based approaches since it gets harder to classify all the nodes correctly when the number of nodes increases in the absence of context.

We also do cross-dataset evaluation to assess the generalization ability of the models. For the cross-dataset evaluation, we translate the titles from English to Turkish and from Turkish to English by using a Google Translator API. In this case the gains because of the graph-model are more pronounced, especially when evaluating the model trained on dataset 2 on dataset 1, where results increase from 43.2% (NG) to 55.7% (PDFS), showing that the graph-based methods generalize better to new data.

In Figure 4.8, we display some qualitative results for the NG, PCFS and PDFS models. Moreover, in Figure 4.9 it can be seen that after the node feature update the cosine similarities of node features are getting higher.



(a) *Feather coat*



(b) *Chino bermuda shorts*

Figure 4.8 – Qualitative evaluation of NG, PCFS and PDFS models respectively. The product titles are written under the subfigure. The gray nodes in the middle are the title nodes which are presented for demonstration purposes. The scores on the edges are the classification scores of the connected nodes. The green and red edges represent positive and negative nodes respectively. The nodes that are bounded by a red box are the wrong classifications. The superiority of the graph based models are more apparent in case of larger graph sizes (see also Figure 4.7b) (best viewed in color).

4.4.5 Gallery-only Main Product Detection

In Table 4.4, we evaluate the setup which we call gallery-only main product detection. In this setup, we take the best models from previous experiments and re-evaluate them while setting all input text embeddings to zero. This setup is an important indicator to evaluate models when they are deployed in the wild where product titles or descriptions will not always be available. It can be seen that the failure rate of the baseline approaches is much higher than the graph-based approaches. PCFS and PDFS models also yield better results than the ICSF model. This can be attributed to the fact that the graph-based models are able to enforce consistency between the bounding boxes thanks to the graph formulation, whereas the other methods show more dependency on the text, and fail in the case where no text input is provided. We attribute the relative high performance of the contrastive model to being biased to selecting the biggest bounding boxes as main products. The margin between the proposed and baselines approaches gets larger when the graph size increases, as can be seen in Figure 4.7b.

Finally, as an additional illustration, in Figure 4.10 we show that our method can

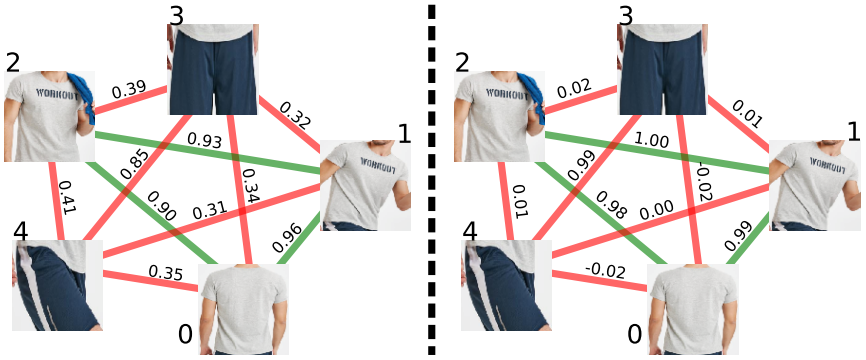
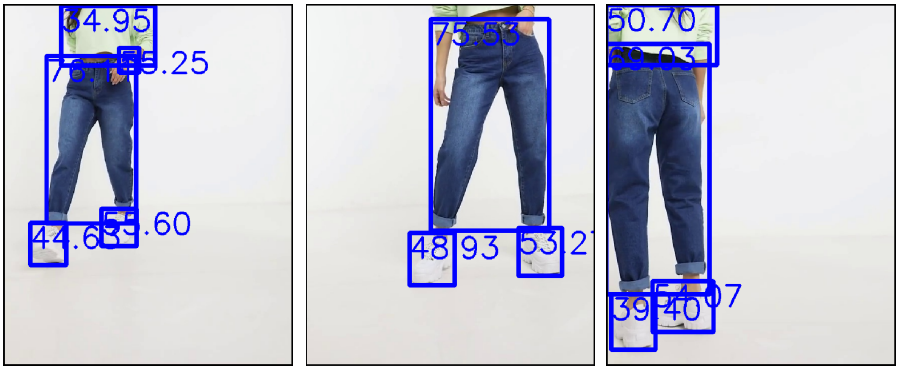


Figure 4.9 – Cosine similarities between the features after the image model and the graph network respectively. After the node feature update, the similar items get closer to each other in the feature space, while the dissimilar items are pushed further away. The main bounding boxes are connected to each other with green edges. (best viewed in color).

be used to detect the main product in videos, by considering several frames from the video. The video frames are taken from products that are being sold in a website of a fashion retailer. In this website, along with the images, titles and descriptions of a product, a video of a model wearing the item is available to customers. We evaluated our PDFS model by randomly sampling 3 frames of a video. After running the fashion detector on these frames, the bounding boxes are input to the main product detection model along with the product title. In the figure, it can be seen that the main product detection model successfully assigns the highest scores to the main items compared to other items. This example shows that the proposed method here for product detection in gallery images can potentially also be used for detection of main products in fashion videos.

4.4.6 MBBDD

We also train and evaluate the baselines and our models on MBBDD. We use the same models and hyperparameters that we used for the previous experiments. The results can be seen in Table 4.5. Since the average number of positive bounding boxes per product is 1.02, the R@1 metric is much higher compared to results on our datasets. Again, especially in the product accuracy metric, graph based models achieve higher results than baselines. We do not display the results of the ICSF model since the products of the dataset are represented by single images. It is also interesting that although images are represented by single images, graph-based



(a) *Mom jeans in blue*

(b) *Dress with lace inserts in white*

(c) *Shacket in black*

Figure 4.10 – Evaluation of the main product detection on video frames. The product titles are written under the subfigure. In all frames, the bounding box that has the highest score is the main bounding box.

Table 4.4 – Performance comparison of the baselines and graph-based approaches in the gallery-only setup, where no text input is provided.

Train	Test	Models	P@1	R@1	mAP	Prod. acc.
1	1	Contrastive	79.8	28.0	89.6	56.9
		NG	89.1	27.4	83.4	21.2
		ICFS	**91.4**	**29.5**	89.1	37.4
		PCFS	88.6	28.1	**91.2**	65.1
		PDFS	83.0	28.4	88.9	**67.9**
	2	Contrastive	81.6	39.3	91.6	65.1
		NG	80.5	38.3	84.2	37.8
		ICFS	**88.4**	**41.4**	**90.4**	51.2
		PCFS	81.3	39.0	89.3	67.6
		PDFS	81.8	39.2	89.6	**71.3**
2	1	Contrastive	54.0	19.5	74.2	20.7
		NG	84.1	26.1	78.5	21.8
		ICFS	**87.8**	**27.2**	79.5	24.8
		PCFS	77.4	24.5	73.4	21.3
		PDFS	80.1	25.0	**80.6**	**30.4**
	2	Contrastive	71.7	35.7	87.4	56.2
		NG	89.9	42.0	91.8	57.1
		ICFS	89.9	42.1	90.9	44.9
		PCFS	84.4	39.5	83.0	54.3
		PDFS	**93.3**	**42.9**	**93.7**	**72.4**

Table 4.5 – Performance comparison of the baselines and graph-based approaches on MBBDD.

Models	P@1	R@1	mAP	Prod. acc.
Contrastive	95.5	94.5	97.7	88.4
NG	96.3	95.3	98.1	91.1
PCFS	**96.5**	**95.5**	**98.2**	94.8
PDFS	96.4	95.4	98.1	**94.8**

models surpass the baselines by a significant margin. This shows the importance taking the context of negative images into account for the final decision on the MBBDD dataset.

4.5 Conclusions

In this work, we propose a new approach for main product detection that incorporates a graph neural network to capture the relationships between all the detected products in a fashion product image gallery. We empirically demonstrate that the graph-based approaches surpass the baselines which do not take the context of product images into account with gains of 6-12 points. If we consider the more challenging *Gallery-only Main Product Detection* we show that using graphs can result in gains of up to 50 points when comparing to the same network without graphs. Finally, we put a focus on the main product detection, a crucial but often overlooked task, that has received less attention from the research community due to its more application oriented structure.

5 Color Naming for Multi-color Fashion Items*

5.1 Introduction

Computer vision offers great potential to develop tools to improve interaction between buyers and sellers in the fashion industry [12, 104, 130]. Color attributes (in this work referred to as color names) are among the essential properties of fashion items and their understanding is therefore crucial for efficient interaction with users. Therefore, in this work we focus on the automatic estimation of color names of images of fashion items. We will focus on extracting the colors of the fashion items in real-world images with background clutter and without available segmentation masks or bounding boxes which indicate the exact location of the fashion item. The task therefore is twofold, automatic detection of the fashion item, and estimation of its colors.

Color naming is a challenging task due to several reasons, including discrepancies between the physical nature of color and human perception (which is also affected by the cultural context), or external factors like varying illumination and complex backgrounds. Moreover complex background, human skin, or human hair act as clutter that deteriorates the accuracy of models. It is important to minimize the effects of this type of clutter in order to improve accuracy. A further difficulty of color naming in fashion, which is the focus of this work, is that many of the objects that we see in the real world have several colors, which complicates the decision making process for algorithms.

Computational color naming has primarily focused on the 11 basic colors of the English language [5, 111]. Those 11 basic colors are defined in the seminal work of Berlin and Kay [6] in which they researched the usage of color names in various different languages. Color names have been successfully used in a number of computer vision applications, including action recognition, visual tracking and image classification; see [109] for an overview. In the field of fashion image understanding, Liu et al. [77] do color naming using Markov Random Fields to infer category and color labels for each pixel in fashion images. To the best of our knowledge, all existing work on color naming focuses either on single colored objects or pixel-wise

*This chapter is based on a publication in the World Conference on Information Systems and Technologies (WorldCIST, 2018) [134]

predictions.

Therefore, we address the problem of color name assignment to multi-color fashion items. We design several neural network architectures and experiment with various loss functions. We collect our own multi-label color dataset by crawling data from Internet sources. We show that a network with an additional classification head that explicitly estimates the number of color names improves performance. In addition, we show in a human annotation experiment that multi-color naming is an ambiguous task and human annotation results are only a few percent higher than results obtained by our best network.

The rest of this Chapter is organized as follows. Related work is discussed in Section 5.2. Section 5.3 describes details of the dataset that we use for the experiments. Section 5.4 elaborates the proposed approach. Experiments are presented in Section 5.5. Finally, we conclude the Chapter in Section 5.6.

5.2 Related Work

Research papers for fashion firstly focused on the segmentation of fashion products in images. Yamaguchi et al. [130], propose the Fashionista dataset consisting of 158,235 fashion photos with associated text annotations. They use a Conditional Random Field Model (CRF) in order to parse fashion clothes pixel-wise. However, their algorithms require fashion tags during the test time to get good accuracies. Simo-Serra et al. [104] address this issue and also propose a CRF model that exploits different image features such as appearance, figure/ground segmentation, shape and location priors for cloth parsing. They manage to obtain state-of-the-art performance on the Fashionista dataset. Liu et al. [85] propose a novel dataset which consists of 800,000 images with 50 categories, 1,000 descriptive attributes, bounding boxes and clothing landmarks. Moreover, they also propose a novel neural network architecture which is called FashionNet. The network learns clothing features by jointly predicting clothing attributes and landmarks. They do pooling and gating of feature maps upon estimated landmark locations to alleviate the effect of clothing deformation and occlusion. Recently, Cervantes et al. [12] propose a hierarchical method for the detection of fashion items in images.

For color, Cheng et al. [21] use a modified version of VGG for pixel-wise prediction out of 11 color labels (which are blue, brown, gray, white, red, green, pink, black, yellow, purple and orange) and a CRF to smooth the prediction. Although their model is robust to background clutter, and can produce pixel-wise prediction, it is not robust to other clutter such as skin and hair color. Van de Weijer et al. [111], use probabilistic latent semantic analysis (PLSA) on Lab histograms to learn color names. Benavente et al. [5], present a model for pixel-wise color name prediction

Figure 5.1 – Sample images from the dataset with varying content and background clutter. Note that we do not provide segmentation and therefore to estimate the colors, the algorithm needs to implicitly segment the main fashion item.

by using chromaticity distribution. Wang et al. [124], propose an algorithm which has two stages: in the first stage, which they name self supervised training, they train a shallow network with color histograms of random patches from the dataset. In the second stage, they fine-tune the same network to predict 11 basic colors. Mylonas et al. [91], use a mixture of Gaussian distributions. Schuerte and Fink [102] propose a randomized hue-saturation-lightness (HSL) transformation to get more natural color distributions; secondly, they used probabilistic ranking to remove the outliers. They claim that these steps helps color models accommodate to the variances seen in real-world images. In none of the before mentioned works to task of color naming multi-color objects is addressed.

5.3 Multi-color Name Dataset

There are several datasets for color name learning. Van de Weijer et al. [111] introduced two datasets, constituted of images of objects retrieved from Google and EBAY respectively, and labeled with the 11 basic color names. Liu et al. [77] introduce another dataset which consists of 2682 images with pixel-level color annotations of the 11 basic colors plus a "background" class. However, almost every image in the dataset has a single color. To the best of our knowledge there is no dataset which explicitly considers multi-color objects.

We therefore collect a new dataset for this work, composed of images of fashion objects with one to nine colors (see Table 5.1). Single colored fashion images are crawled from various online shopping sites, and most of the multicolor labeled images are obtained by querying the Google images search engine with a query term containing a pair of color names and a fashion keyword (e.g. red and blue skirt) and downloading the 100 first images. There are 67 fashion keywords that we use and 55 color pairs that can be obtained with combinations of 11 basic colors. At

Table 5.1 – The number of images for each color category.

	1	2	3	4	5	6	7	8	9	Total
Train	5556	5431	2178	1203	476	131	19	4	3	15001
Test	50	50	50	50	0	0	0	0	0	200

the end, we remove irrelevant and noisy data and crop the fashion item to prepare the dataset.

This process allows us to obtain images with two colors and more colors, as sometimes the search engine also returns images with additional colors not included in the query. Unfortunately, this leads to an imbalance between the number of 2-colored images and multi-colored images. Directly crawling for products with more than two colors using Google Images produces unsatisfactory results.

The dataset includes different types of images of varying complexity: catalog shots with smooth or complex background, images with plain background without any person or images taken by social media users; all labeled with the color names of the main fashion item. Sample images from the dataset can be seen in Figure 5.1. It should be noted that we do not use segmentation for the images, and naming the multiple colors of the fashion items includes dealing with clutter from the background, occlusions, and skin and hair of the person. However, if there is more than one fashion item in an image, to avoid any confusion, we provide a bounding box for the correspondent fashion item. In any case, the network has to implicitly segment the fashion item from occlusions and clutters.

5.4 Networks for Multi-color Name Prediction

Methods on color naming focus on single colored objects. In this work we aim to propose a method for multi-colored fashion items. We evaluate several network architectures and losses for this task.

5.4.1 Network Design

In principle we believe the mapping from RGB to color names not to be highly complicated and only several layers are required. However, differentiating background from foreground is a highly complex process that requires many layers and should be implicitly done by the network.

First, we propose a shallow network; the truncated version of Alexnet [62]. We keep the first five convolution layers of the architecture and remove the fully connected layers of 4096 dimension. At the end we add a fully connected layer

which maps features to the eleven basic color names. As a second network we use the full Alexnet architecture. Both nets are initialized with pretrained weights from ILSVRC 2012 dataset [30]. We think that finetuning from this model can alleviate noise caused by clutter such as complex backgrounds, hair or skin.

5.4.2 Loss Functions

We consider two loss functions for the purpose of color naming for multi-color fashion items. The first loss we consider to train the network is the softmax cross-entropy loss (SCE). The softmax cross-entropy can be seen in Equation 5.1:

$$L_{sce} = -\frac{1}{N}\sum_{i}^{N} P(i)\log Q(i) \tag{5.1}$$

where Q the predicted color distribution, P is the true color distribution, and N is the number of images. Q is obtained by applying a softmax normalization to the output of the last fully connected layer of the network, and the ground truth P is computed by assigning a uniform probability to all color names annotated for the fashion item (e.g. in case of three annotated color names, P would contain three elements with value 0.33).

While the softmax cross-entropy loss teaches a network to compute color probability distributions for an input fashion item, no decision is made on the actual number of colors. To remedy this, a threshold on the computed probabilities Q, learned from an independent validation set, is used to discard the colors unlikely to be really present.

The second loss we consider is the binary cross-entropy loss (BCE), which inherently supports multi-label classification. This loss is commonly used for attribute detection [85, 86] because it models the presence of multiple labels simultaneously. Therefore it is expected to obtain better results than the softmax-cross entropy. Unlike with the softmax cross-entropy loss, the computed probability for a color name is independent of the others. For example the probability of both 'green' and 'orange' can be one simultaneously, something which is impossible for the softmax cross-entropy loss. Therefore, the loss trains 11 binary classifiers for each color. In Equation 5.2, the binary cross-entropy can be seen.

$$L_{bce} = -\frac{1}{N}\sum_{i}^{N} P_i\log Q_i + (1-P_i)\log(1-Q_i) \tag{5.2}$$

Similarly as the softmax-cross entropy loss we determine a threshold on a validation set to decide on the colors which are present in the fashion item. We found this to yield better results than choosing the natural threshold of 0.5.

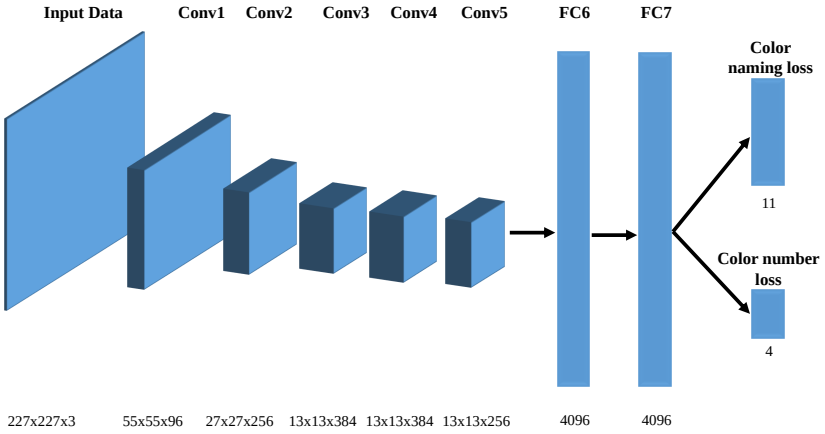


Figure 5.2 – The architecture of the deep network with the extra head.

5.4.3 Extra Head to Explicitly Estimate Number of Colors

In the previous section we consider two losses to estimate the color names. In principle the binary cross-entropy loss which implements the multi-label softmax cross-entropy loss is more suitable for the estimation of multiple colors. However, the probabilities which are the outcome of these networks both encode information of the number of color names as well as the confidence of the network in its estimation of the color names. Considering a single colored object which the system is not sure to label with either 'orange' or 'red', the algorithm that is based on binary cross-entropy might give both colors a probability of 0.6. Based on this we might conclude that the object is a multi-color object which is both 'orange' and 'red'. However, looking at the object it might be obvious that it only has a single color.

Therefore, we experiment with adding an extra classification head to the network, which explicitly estimates the number of colors in the main object. We model this objective as a classification task, and define four possible classes: one, two, three and four or more colors. A natural choice for this objective is the softmax cross-entropy loss layer, typically used for classification. In the experiments, we add this additional objective both to the networks which use softmax-cross entropy loss and the binary cross-entropy loss. The architecture of the network can be seen in Figure 5.2.

5.4.4 Training Procedure

To train the network, we finetune from an Alexnet model which is trained on ILSVRC 2012 [30] using the Caffe framework [55]. The batch size is 64, the optimization method is SGD with momentum, set to 0.99, and we decrease the learning rate after every 5000 iterations. The initial learning rate is 0.0001 and the maximum iteration number is 20000. We also use data augmentation techniques in order to increase the accuracy of the models. The data augmentation techniques that we use are changing contrast, rescaling image and cropping random parts from images. Rescaling basically consists on changing the resolution of the image before resizing to the required network input size. The probability that any augmentation technique is applied to an image is 50%. We never keep both the original and the augmented image in the same batch, as we have observed that it may negatively impact the accuracy of the learned model. Finally, to avoid aspect ratio distortions caused by the resizing process, we use a padding function in order to make all images square.

5.5 Experiments

To evaluate the performance we use label based metric methods. We calculate the micro-precision, micro-recall, micro-F1, macro-precision, macro-recall and macro-F1. In the micro methods we sum up true positive, false positive and true negative for each label in order to get micro-recall and micro-precision. In the macro methods, we calculate precision and recall of each label and average them in order to get the macro-recall and macro-precision. The main difference is that the macro metrics do not take the label imbalance into account. To clarify the difference between the micro and macro methods, here we give the micro-precision and macro-precision:

$$P_{micro} = \frac{\sum_{j=i}^{L} tp_j}{\sum_{j=0}^{L} tp_j + \sum_{j=0}^{L} fp_j} \qquad P_{macro} = \sum_{j=0}^{L} \frac{tp_j}{tp_j + fp_j} \tag{5.3}$$

L is the number of classes, tp_j and fp_j is the true positive and false positive of class j. All of the results can be seen in Table 5.2. We focus on the F1-score which is a fair metric to compare methods. We first evaluate the two network architectures, namely the shallow and deep network. Both of the models have the extra head which forces them to learn the number of colors on a fashion item. The deep model

(a) **GT**: **DE**: **SH**: (b) **GT**: **DE**: **SH**: (c) **GT**: **DE**: **SH**:

Figure 5.3 – Qualitative results of the shallow and deep networks. GT, DE, SH denote the ground truth, the predictions of the deep and shallow model respectively. Note that the networks should estimate the colors of the fashion item while ignoring the non-relevant colors present in the background.

Table 5.2 – Results of our models and the human annotators.

		Shallow BCE w/ extra head	SCE w/o extra head	SCE w/ extra head	BCE w/o extra head	BCE w/ extra head	Human Score
micro	precision	77.2	85.6	80.3	82.3	83.6	81.4
	recall	67.2	63.2	70.8	69.8	71.2	81.9
	F1	71.9	72.7	75.2	75.5	**76.9**	81.3
macro	precision	78.4	84.7	79.8	81.4	83.1	81.6
	recall	65.2	62.1	69.8	67.4	69.3	80.9
	F1	69.3	69.4	73.5	72.5	**74.2**	80.2

clearly outperforms the shallow model. We attribute this to the fact that the shallow model is not able to segment the fashion items implicitly, and therefore fails for the more cluttered cases as can be seen in Figure 5.3.

Next we evaluate the different losses, and we verify if the additional head which explicitly predicts the number of colors contributes to a performance gain. It can be seen that adding the additional objective improves both the softmax cross-entropy loss and the binary cross-entropy loss; it forces the network to learn the number of colors on a fashion item, and also contributes to name them as can be seen when comparing the columns 4-5 and 6-7 of Table 5.2. During the inference, the extra head can predict maximum 4 colors. In case the networks without extra head predicts more than 4 colors, we get the first 4 with the highest scores.

In the last column of Table 5.2, we show the average performance obtained

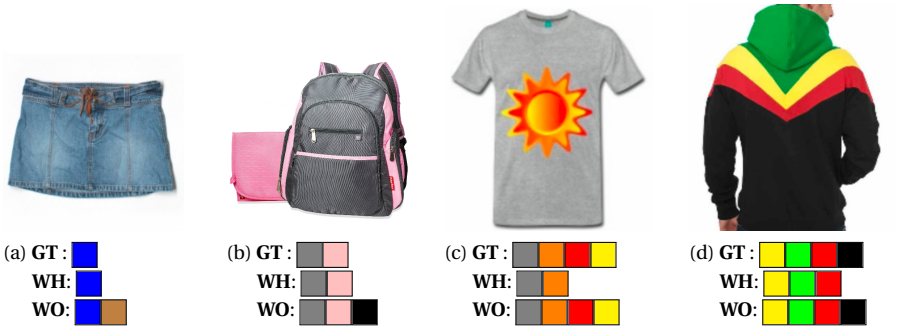


Figure 5.4 – Qualitative results of the BCE model with (WH) and without (WO) the extra head.

by humans for the same task. We asked seven annotators of different ages and backgrounds to provide labels for the images in the test set. The obtained scores for humans show that multi-color labelling is an ambiguous task, and for many objects humans do not agree on the labels. This score can be considered to be an upper bound for computational methods.

The contribution of adding an extra head is shown in Figure 5.4. From the ground truth and prediction of the cross entropy models it can be seen that the extra head provides robustness if the color distribution is not uniform in the image. However, it makes the model more conservative and biases it towards predicting a lower number of colors in the image (last two examples on the right).

5.6 Conclusions

In this work we address the problem of color name estimation in multi-colored objects that, to the best of our knowledge, we are the first to address. We collect a dataset of over 15.000 images with a varying number of colors per object and we evaluate several network architectures for the purpose of multi-color estimation. Preliminary results show that adding an additional objective to explicitly estimate the number of colors in the object improves results. Following recent work we are interested in extending the set of color names to include a wider range of colors [90, 135]. We hope that this work further motivates researchers to investigate the more realistic setting of color naming for multi-colored objects.

6 Conclusions and Future Work

6.1 Conclusions

The commercial impact of deep learning has been tremendous. Every year, companies propose new computer vision problems to the research community. This has led to significant advances in many industries, including autonomous driving, robotics, visual search for online retailers, medical imaging, etc. Fashion is one of the industries that has been eager to incorporate novel functionalities based on the new possibilities offered by big data and machine learning. Thanks to the close collaboration between the industry and academy, advances develop at a rapid pace. This thesis has aimed to contribute to the development of smart fashion applications. In doing so, we have also contributed to the more general application of multi-label classification that is also used in many other industries.

In this thesis, we addressed several problems that are common in the fashion industry. Firstly, in Chapter 2, we addressed the problem of imposing fixed orders to recurrent models for multi-label image classification. In Chapter 3, we tackled the problem of long term dependencies caused by recurrent models due to their sequential nature for orderless set prediction problem. In Chapter 4, we addressed the problem of main product detection for fashion product parsing. In Chapter 5, we addressed the problem of color naming for multicolored fashion items.

The methods proposed and the results obtained in this thesis are:

- **Chapter 2: Orderless Recurrent Models for Multi-label Classification:**. We showed that imposing fixed orders does not always correspond the natural order of objects in images. We proposed novel orderless losses which align the ground truth sequence with predicted labels in a way that the minimum loss is achieved. We empirically showed that our models do not suffer from duplicate predictions, something very common for recurrent models that are trained with fixed order methods. We also surpassed the state-of-the-results with our orderless recurrent models.

- **Chapter 3: Visual Transformers with Primal Object Queries for Multi-label Image Classification:** We introduced the *primal object queries* that differ from the standard object queries in the way that they are used. We achieved

significant speed-up of training convergence and obtained state-of-the-art-results on MS-COCO and NUS-WIDE datasets. We proposed a visual transformer model that is unique in that it manages to learn long-term dependencies and integrates the mixup technique for multi-label classification successfully.

- **Chapter 4: Main Product Detection with Graph Networks for Fashion:** We collected a new dataset with fashion products represented by multiple images and provided the main product annotations. We exploited graph convolutional networks to represent bounding boxes of a fashion product as nodes in a fully connected graph. We empirically showed that learning the relation between the bounding boxes during the training and taking the entire context into account for the final decision improve the main product detection performance significantly, especially in harder setups such as when we miss the title-input at inference time.

- **Chapter 5: Color Naming for Multi-color Fashion Items:** We proposed a novel architecture with an additional head that explicitly estimates the number of colors in multicolored fashion items. We show that it alleviated the problem of ambiguity caused by fine lines between the classes in color spaces and resulted in better color naming performance.

6.2 Future Work

For multi-label image classification, we are interested in the idea of target-aware data augmentations. For example, improving the hard mixup in a way that the generated targets reflect the statistics of the training set or imposing more constraints such as restricting the mixup between pairs whose labels do not co-occur in the training set. Moreover, we would like to tackle the order problem by employing different methods other than alignment with Hungarian algorithm, since the complexity ($\mathcal{O}(n^3)$) increases significantly when the number of labels per image is higher.

For main product detection, we would like to employ transformers instead of graph convolutional networks for future training. Since our fashion graph is fully connected, we can exploit self-attention layers in transformers to learn the relation between the bounding boxes that belong to the same product. This would also make the training faster since the parallelization is easier with transformers. Moreover, the cross-attention module, which takes the *keys* and *values* from a convolutional backbone or a stack of encoder layers, would act as natural visual grounding (locating the most relevant object or region in an image) module which

is very relevant for main product detection.

For color naming, features of colors that occupy lower number of pixels in an image tend to disappear due to multiple pooling layers. Instead of increasing the image size and computational cost due to larger features, we are interested in imposing extra losses in intermediate features to improve the recognition of minor colors in fashion items.

Summary of published works

1. **Vacit Oguz Yazici**, Abel Gonzalez-Garcia, Arnau Ramisa, Bartlomiej Twardowski, and Joost van de Weijer. "Orderless recurrent models for multi-label classification." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13440-13449. 2020.

2. **Vacit Oguz Yazici**, Joost van de Weijer, and Arnau Ramisa. "Color naming for multi-color fashion items." World Conference on Information Systems and Technologies. pages 64-73. Springer, Cham, 2018.

3. **Vacit Oguz Yazici**, Longlong Yu, Arnau Ramisa and Joost van de Weijer. "Main product detection with graph networks in fashion." In IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop on Computer Vision for Fashion, Art and Design, 2020.

4. Lu Yu, **Vacit Oguz Yazici**, Xialei Liu, Joost van de Weijer, Yongmei Cheng, and Arnau Ramisa. Learning metrics from teachers: Compact networks for image embedding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2907–2916, 2019.

Bibliography

[1] Adrien. Future of ecommerce: 10 international growth trends. beeketing. com/blog/future-ecommerce-2019, 2020. 2020 update.

[2] Kenan E Ak, Joo Hwee Lim, Jo Yew Tham, and Ashraf A Kassim. Which shirt for my first date? towards a flexible attribute-based fashion query system. *Pattern Recognition Letters*, 112:212–218, 2018.

[3] S. Bai. A survey on automatic image caption generation. *Neurocomputing*, 311:291 – 304, 2018.

[4] M. Bastan, A. Ramisa, and M. Tek. T-vse: Transformer-based visual semantic embedding. CVPR Workshop on Computer Vision for Fashion, Art, and Design, 2020.

[5] Robert Benavente, Maria Vanrell, and Ramon Baldrich. Parametric fuzzy sets for automatic color naming. *JOSA A*, 25(10):2582–2593, 2008.

[6] Brent Berlin and Paul Kay. *Basic color terms: Their universality and evolution*. Univ of California Press, 1991.

[7] Wei Bi and James Kwok. Efficient multi-label classification with many labels. In *International Conference on Machine Learning*, pages 405–413. PMLR, 2013.

[8] Lukas Bossard, Matthias Dantone, Christian Leistner, Christian Wengert, Till Quack, and Luc Van Gool. Apparel classification with style. In *Asian conference on computer vision*, pages 321–335. Springer, 2012.

[9] Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Describing people: A poselet-based approach to attribute classification. In *2011 International Conference on Computer Vision*, pages 1543–1550. IEEE, 2011.

[10] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[11] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[12] Esteve Cervantes, Long Long Yu, Andrew D Bagdanov, Marc Masana, and Joost van de Weijer. Hierarchical part detection with deep neural networks. In *ICIP*, pages 1933–1937, 2016.

[13] Huizhong Chen, Andrew Gallagher, and Bernd Girod. Describing clothing by semantic attributes. In *European conference on computer vision*, pages 609–623. Springer, 2012.

[14] Liang-Chieh Chen, Alexander Schwing, Alan Yuille, and Raquel Urtasun. Learning deep structured models. In *International Conference on Machine Learning (ICML)*, pages 1785–1794, 2015.

[15] Shang-Fu Chen, Yi-Chen Chen, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. Order-free rnn with visual attention for multi-label classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[16] Tianshui Chen, Zhouxia Wang, Guanbin Li, and Liang Lin. Recurrent attentional reinforcement learning for multi-label image recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[17] X. Chen and C. Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*, 2015.

[18] Yuxin Chen, Gaoqun Ma, Chunfeng Yuan, Bing Li, Hui Zhang, Fangshi Wang, and Weiming Hu. Graph convolutional network with structure pooling and joint-wise channel attention for action recognition. *Pattern Recognition*, page 107321, 2020.

[19] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5177–5186, 2019.

[20] Zhaomin Chen, Quan Cui, Xiu-Shen Wei, Xin Jin, and Yanwen Guo. Disentangling, embedding and ranking label cues for multi-label image recognition. *IEEE Transactions on Multimedia*, 2020.

[21] Zhiyi Cheng, Xiaoxiao Li, and Chen Change Loy. Pedestrian color naming via convolutional neural network. In *ACCV*, pages 35–51. Springer, 2016.

[22] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*, 2014.

[23] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yan-Tao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)*, Santorini, Greece., July 8-10, 2009.

[24] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, pages 1–9, 2009.

[25] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. Meshed-memory transformer for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10578–10587, 2020.

[26] G. Cucurull, P. Taslakian, and D. Vazquez. Context-aware visual compatibility prediction. In *CVPR*, pages 12617–12626, 2019.

[27] Z. Cui, Z. Li, S. Wu, X. Zhang, and L. Wang. Dressing as a whole: Outfit compatibility learning based on node-wise graph neural networks. In *The World Wide Web Conference*, pages 307–317, 2019.

[28] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[29] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[30] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[31] J. Devlin, M. Chang, Kenton Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[33] Wei Di, Catherine Wah, Anurag Bhardwaj, Robinson Piramuthu, and Neel Sundaresan. Style finder: Fine-grained clothing style detection and retrieval. In *Proceedings of the IEEE Conference on computer vision and pattern recognition workshops*, pages 8–13, 2013.

[34] J. Donahue, L. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, . Saenko, and T.Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[35] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[36] H. Fang, S. Gupta, F. Iandola, R. Srivastava, L. Deng, and P. Dollár others. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

[37] Jianlong Fu, Jinqiao Wang, Zechao Li, Min Xu, and Hanqing Lu. Efficient clothing retrieval with semantic-preserving visual phrases. In *Asian conference on computer vision*, pages 420–431. Springer, 2012.

[38] Andrew C Gallagher and Tsuhan Chen. Clothing cosegmentation for recognizing people. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[39] Weifeng Ge, Sibei Yang, and Yizhou Yu. Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1277–1286, 2018.

[40] Yuying Ge, Ruimao Zhang, Xiaogang Wang, Xiaoou Tang, and Ping Luo. Deepfashion2: A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5337–5345, 2019.

[41] Nadia Ghamrawi and Andrew McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 195–200. ACM, 2005.

[42] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013.

[43] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *IEEE workshop on automatic speech recognition and understanding*, pages 273–278, 2013.

[44] Hao Guo, Kang Zheng, Xiaochuan Fan, Hongkai Yu, and Song Wang. Visual attention consistency under image transforms for multi-label image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 729–739, 2019.

[45] Yuhong Guo and Suicheng Gu. Multi-label classification using conditional dependency networks. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[46] M Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C Berg, and Tamara L Berg. Where to buy it: Matching street clothing photos in online shops. In *Proceedings of the IEEE international conference on computer vision*, pages 3343–3351, 2015.

[47] Xintong Han, Zuxuan Wu, Phoenix X Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S Davis. Automatic spatially-aware fashion concept discovery. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1463–1471, 2017.

[48] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. Learning fashion compatibility with bidirectional lstms. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1078–1086, 2017.

[49] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[50] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[51] Md. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A Comprehensive Survey of Deep Learning for Image Captioning. *ACM Computing Surveys*, 51(6):1–36, feb 2019.

[52] Hexiang Hu, Guang-Tong Zhou, Zhiwei Deng, Zicheng Liao, and Greg Mori. Learning structured inference neural networks with label relations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[53] Md Amirul Islam, Matthew Kowal, Konstantinos G Derpanis, and Neil DB Bruce. Feature binding with category-dependant mixup for semantic segmentation and adversarial robustness. In *British Machine Vision Conference*, 2020.

[54] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[55] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM int. conf. on Multimedia*, pages 675–678. ACM, 2014.

[56] Jiren Jin and Hideki Nakayama. Annotation order matters: Recurrent image annotator for arbitrary length image tagging. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2452–2457. IEEE, 2016.

[57] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. *ICCV*, 2015.

[58] Justin Johnson, Lamberto Ballan, and Li Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4624–4632, 2015.

[59] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[60] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[61] R. Kiros, R. Salakhutdinov, and R. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *TACL*, 2015.

[62] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[63] Z. Kuang, Y. Gao, G. Li, P. Luo, Y. Chen, L. Lin, and W. Zhang. Fashion retrieval via graph reasoning networks on a similarity pyramid. In *ICCV*, pages 3066–3075, 2019.

[64] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[65] Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. General multi-label image classification with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16478–16488, 2021.

[66] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[67] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[68] Dangwei Li, Xiaotang Chen, and Kaiqi Huang. Multi-attribute learning for pedestrian attribute recognition in surveillance scenarios. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 111–115. IEEE, 2015.

[69] Liang Li, Shuhui Wang, Shuqiang Jiang, and Qingming Huang. Attentive recurrent neural network for weak-supervised multi-label image classification. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1092–1100. ACM, 2018.

[70] Yining Li, Chen Huang, Chen Change Loy, and Xiaoou Tang. Human attribute recognition by deep hierarchical contexts. In *European Conference on Computer Vision (ECCV)*, pages 684–700. Springer, 2016.

[71] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. Mining fashion outfit composition using an end-to-end deep learning approach on set data. *IEEE Transactions on Multimedia*, 19(8):1946–1955, 2017.

[72] Yuncheng Li, Yale Song, and Jiebo Luo. Improving pairwise ranking for multi-label image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3617–3625, 2017.

[73] Xiaodan Liang, Liang Lin, Wei Yang, Ping Luo, Junshi Huang, and Shuicheng Yan. Clothes co-parsing via joint image segmentation and labeling with application to clothing retrieval. *IEEE Transactions on Multimedia*, 18(6):1175–1186, 2016.

[74] Lizi Liao, Xiangnan He, Bo Zhao, Chong-Wah Ngo, and Tat-Seng Chua. Interpretable multimodal retrieval for fashion products. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1571–1579, 2018.

[75] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[76] Feng Liu, Tao Xiang, Timothy M Hospedales, Wankou Yang, and Changyin Sun. Semantic regularisation for recurrent image annotation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2872–2880, 2017.

[77] S. Liu, J. Feng, C. Domokos, H. Xu, J. Huang, Z. Hu, and S. Yan. Fashion parsing with weak color-category labels. *IEEE Transactions on Multimedia*, 16(1), 2014.

[78] Si Liu, Jiashi Feng, Csaba Domokos, Hui Xu, Junshi Huang, Zhenzhen Hu, and Shuicheng Yan. Fashion parsing with weak color-category labels. *IEEE Transactions on Multimedia*, 16(1):253–265, 2013.

[79] Si Liu, Zheng Song, Guangcan Liu, Changsheng Xu, Hanqing Lu, and Shuicheng Yan. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3330–3337. IEEE, 2012.

[80] Xihui Liu, Haiyu Zhao, Maoqing Tian, Lu Sheng, Jing Shao, Shuai Yi, Junjie Yan, and Xiaogang Wang. Hydraplus-net: Attentive deep features for pedestrian analysis. In *IEEE International Conference on Computer Vision (ICCV)*, pages 350–359, 2017.

[81] Yongcheng Liu, Lu Sheng, Jing Shao, Junjie Yan, Shiming Xiang, and Chunhong Pan. Multi-label image classification via knowledge distillation from weakly-supervised detection. In *ACM International Conference on Multimedia*, pages 700–708, 2018.

[82] Yongsheng Liu, Wenyu Chen, Hong Qu, SM Hasan Mahmud, and Kebin Miao. Weakly supervised image classification and pointwise localization with graph convolutional networks. *Pattern Recognition*, page 107596, 2020.

[83] Z. Liu, S. Yan, P. Luo, X. Wang, and X. Tang. Fashion landmark detection in the wild. In *ECCV*, pages 229–245. Springer, 2016.

[84] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1096–1104, 2016.

[85] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, June 2016.

[86] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738, 2015.

[87] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[88] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*, 2015.

[89] T. Mikolov, I. Sutskever, K. Chen, G. S Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[90] Dimitris Mylonas and Lindsay MacDonald. Augmenting basic colour terms in english. *Color Research & Application*, 2015.

[91] Dimitris Mylonas, Lindsay MacDonald, and Sophie Wuerger. Towards an online color naming model. In *Color and Imaging Conference*, number 1, pages 140–144. Society for Imaging Science and Technology, 2010.

[92] Yulei Niu, Zhiwu Lu, Ji-Rong Wen, Tao Xiang, and Shih-Fu Chang. Multi-modal multi-scale deep learning for large-scale image annotation. *IEEE Transactions on Image Processing*, 28(4):1720–1731, 2019.

[93] W. Norcliffe-Brown, S. Vafeias, and S. Parisot. Learning conditioned graph structures for interpretable visual question answering. In *NIPS*, pages 8334–8343, 2018.

[94] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[95] A Paszke, S. Gross, et al. Pytorch: An imperative style, high-performance deep learning library. In *In NIPS*, pages 8024–8035. 2019.

[96] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[97] Luis Pineda, Amaia Salvador, Michal Drozdzal, and Adriana Romero. Elucidating image-to-set prediction: An analysis of models, losses and datasets. *arXiv preprint arXiv:1904.05709*, 2019.

[98] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333, 2011.

[99] Antonio Rubio, LongLong Yu, Edgar Simo-Serra, and Francesc Moreno-Noguer. Multi-modal embedding for main product detection in fashion. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2236–2242, 2017.

[100] D.E. Rumelhart, G.E. Hintont, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[101] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[102] Boris Schauerte and Gernot A Fink. Web-based learning of naturalized color models for human-machine interaction. In *DICTA*, pages 498–503. IEEE, 2010.

[103] Ron Schmelzer. The fashion industry is getting more intelligent with ai. www.forbes.com/sites/cognitiveworld/2019/07/16/the-fashion-industry-is-getting-more-intelligent-with-ai, 2019. 2019-07-16.

[104] Edgar Simo-Serra, Sanja Fidler, Francesc Moreno-Noguer, and Raquel Urtasun. A High Performance CRF Model for Clothes Parsing. In *ACCV*, 2014.

[105] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[106] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning (ICML)*, pages 843–852, 2015.

[107] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[108] Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

[109] Joost Van De Weijer and Fahad Shahbaz Khan. An overview of color name applications in computer vision. In *CCIW*, pages 16–22, 2015.

[110] Joost Van de Weijer and Cordelia Schmid. Applying color names to image description. In *ICIP*, volume 3, pages III–493. IEEE, 2007.

[111] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1523, 2009.

[112] M. I Vasileva, B. A Plummer, K. Dusad, S. Rajpal, R. Kumar, and D. Forsyth. Learning type-aware embeddings for fashion compatibility. In *ECCV*, pages 390–405, 2018.

[113] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[114] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[115] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR, 2019.

[116] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[117] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

[118] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.

[119] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

[120] M. Wang, L. Yu, et al. Deep graph library: Towards efficient and scalable deep learning on graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[121] Qian Wang, Ning Jia, and Toby P Breckon. A baseline for multi-label image classification using an ensemble of deep convolutional neural networks. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 644–648. IEEE, 2019.

[122] W. Wang, Y. Xu, J. Shen, and S. Zhu. Attentive fashion grammar network for fashion landmark detection and clothing category classification. In *CVPR*, pages 4271–4280, 2018.

[123] Xianwang Wang and Tong Zhang. Clothes search in consumer photos via color matching and attribute learning. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1353–1356, 2011.

[124] Yuhang Wang, Jing Liu, Jinqiao Wang, Yong Li, and Hanqing Lu. Color names learning using convolutional neural networks. In *ICIP*, pages 217–221. IEEE, 2015.

[125] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. Hierarchical multi-label classification networks. In *International Conference on Machine Learning*, pages 5075–5084. PMLR, 2018.

[126] Yunchao Wei, Wei Xia, Junshi Huang, Bingbing Ni, Jian Dong, Yao Zhao, and Shuicheng Yan. Cnn: Single-label to multi-label. *arXiv preprint arXiv:1406.5726*, 2014.

[127] Hui Wu, Yupeng Gao, Xiaoxiao Guo, Ziad Al-Halah, Steven Rennie, Kristen Grauman, and Rogerio Feris. Fashion iq: A new dataset towards retrieving images by natural language feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11307–11317, 2021.

[128] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[129] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[130] Kota Yamaguchi, M Hadi Kiapour, Luis E Ortiz, and Tamara L Berg. Parsing clothing in fashion photographs. In *CVPR*, pages 3570–3577. IEEE, 2012.

[131] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. *arXiv preprint arXiv:1801.07455*, 2018.

[132] Ming Yang and Kai Yu. Real-time clothing recognition in surveillance videos. In *2011 18th IEEE International Conference on Image Processing*, pages 2937–2940. IEEE, 2011.

[133] Vacit Oguz Yazici, Abel Gonzalez-Garcia, Arnau Ramisa, Bartlomiej Twardowski, and Joost van de Weijer. Orderless recurrent models for multi-label classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13440–13449, 2020.

[134] Vacit Oguz Yazici, Joost van de Weijer, and Arnau Ramisa. Color naming for multi-color fashion items. In *World Conference on Information Systems and Technologies*, pages 64–73. Springer, 2018.

[135] Lu Yu, Lichao Zhang, Joost van de Weijer, Fahad Shahbaz Khan, Yongmei Cheng, and C Alejandro Parraga. Beyond eleven color names for image understanding. *Machine Vision and Applications*, 2018.

[136] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[137] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[138] Li Zhang, Xiangtai Li, Anurag Arnab, Kuiyuan Yang, Yunhai Tong, and Philip HS Torr. Dual graph convolutional network for semantic segmentation. *arXiv preprint arXiv:1909.06121*, 2019.

[139] Zhi Zhang, Tong He, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of freebies for training object detection neural networks. *arXiv preprint arXiv:1902.04103*, 2019.

[140] Jia-Xing Zhong, Nannan Li, Weijie Kong, Shan Liu, Thomas H Li, and Ge Li. Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1237–1246, 2019.

[141] Feng Zhu, Hongsheng Li, Wanli Ouyang, Nenghai Yu, and Xiaogang Wang. Learning spatial regularization with image-level supervisions for multi-label image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5513–5522, 2017.

[142] Shizhan Zhu, Raquel Urtasun, Sanja Fidler, Dahua Lin, and Chen Change Loy. Be your own prada: Fashion synthesis with structural coherence. In *Proceedings of the IEEE international conference on computer vision*, pages 1680–1688, 2017.

[143] Xinxin Zhu, Lixiang Li, Jing Liu, Haipeng Peng, and Xinxin Niu. Captioning transformer with stacked attention modules. *Applied Sciences*, 8(5):739, 2018.