UNIVERSITAT DE BARCELONA

Deep Learning and Uncertainty Modeling in Visual Food Analysis

Eduardo Aguilar

UNIVERSITAT DE BARCELONA

DOCTORAL THESIS

Deep Learning and Uncertainty Modeling in Visual Food Analysis

Author:
Eduardo AGUILAR

Supervisor:
Dr. Petia RADEVA

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

"Computer Vision and Machine Learning at the University of Barcelona" (CVMLUB) Consolidated Research Group,
Departament de Matemàtiques i Informàtica

May 21, 2020

Abstract

Several computer vision approaches have been proposed for tackling food analysis problems, due to the challenging problem it poses, the ease collection of food images, and its numerous applications to health and leisure. However, high food ambiguity, inter-class variability and intra-class similarity define a real challenge for the Deep learning and Computer Vision algorithms. With the advent of Convolutional Neural Networks, the complex problem of visual food analysis has experienced significant improvement. Despite this, for real applications, where thousands of foods must be analyzed and recognized, it is necessary to better understand what the model learns and, from this, guide its learning on more discriminatives features to improve its accurate and robustness.

In this thesis we address the problem of analyzing food images through methods based on deep learning algorithms. There are two distinguishable parts. In the first part, we focus on the food recognition task and delve into uncertainty modeling. First, we propose a new multi-task model that is able to simultaneously predict different food-related tasks. Here, we extend the homoscedastic uncertainty modeling to allow single-label and multi-label classification and propose a regularization term, which jointly weighs the tasks as well as their correlations. Second, we propose a novel prediction scheme based on a class hierarchy that considers local classifiers, in addition to a flat classifier. For this, we define criteria based on the Epistemic Uncertainty estimated from the 'children' classifiers and the prediction from the 'parent' classifier to decide the approach to use. And third, we propose three new data augmentation strategies that analysis class-level or sample-level epistemic uncertainty to guide the model training.

In the second part we contribute to the design of new methods for food detection (food/non-food classification), for ensemble of food classifiers and for semantic food detection. First, we proposes an overview of the last advances on food/non-food classification and an optimal model based on the GoogLeNet architecture, Principal Component Analysis, and a Support Vector Machine. Second, we propose a combination of multiple classifiers for food recognition based on two different Convolutional models that complement each other and thus, achieve an improvement in performance. And third, we address the problem of automatic food tray analysis in canteens and restaurants environment through a new approach that integrates in the same framework food localization, recognition and segmentation for semantic food detection.

All the methods designed in this thesis are validated and contrasted over relevant public food datasets and the results obtained are reported in detail.

Acknowledgements

First of all, I want to express my gratitude to my advisor Dr. Petia Radeva for believing in me and giving me the opportunity to live this PhD experience. Moreover, I want to thank her for the invaluable guidance, motivation, patience and encouragement her have given me throughout all these years.

I also want to thank the members of the CVUB group for their willingness and camaraderie. I especially thank Marc for his support, mainly during the first years, and for always being present to discuss and give his opinion on a research issues. To Alejandro, for countless anecdotes and fun moments we share. To Estefania, for her kindness and cheerful companionship. To Bhalaji and Rupali for their willingness and collaboration. To Bea, Maedeh, Mariella and Gabriel for their invaluable companionship. Thanks all.

I also acknowledge the people on the LogMeal team for the good time I spent on this business adventure. Thanks to Pedro, Juan Luis, Albert and Marta. Wish you all the best.

I am very grateful to the DISC academics at the Universidad Católica del Norte, specially to Claudio Meneses and Luis Lobos, who trusted me and gave me the opportunity to improve my academic training.

Finally, and not least, I want thanks to my family for their unconditional support, patience and love. Without you I wouldn't be here.

This thesis was financially supported by CONICYT Becas Chile and the Universidad Católica del Norte.

Contents

Abstract iii

Acknowledgements v

1 Introduction 1

1.1 Problem statement . . . 1

1.2 Food Analysis Relevance . . . 3

1.2.1 Food for Healthcare . . . 3

1.2.2 Restaurant Industry . . . 3

1.2.3 Recommender Systems . . . 4

1.2.4 Social Networking Tools . . . 4

1.3 Objectives . . . 4

1.4 Research Contributions . . . 5

1.5 Thesis Organization . . . 8

2 Background 9

2.1 Neural Networks . . . 9

2.2 Bayesian Neural Networks . . . 11

2.3 Convolutional Neural Networks . . . 13

2.4 Uncertainty in Deep Learning . . . 16

3 Regularized Uncertainty-based Multi-Task Learning Model for Food Analysis 19

3.1 Introduction . . . 19

3.2 Related Work . . . 21

3.2.1 Multi-Task Learning based on CNNs . . . 22

3.3 Regularized Uncertainty-based Multi-Task Learning model for food analysis 23

3.3.1 Model Design and Activation Functions . . . 24

3.3.2 Multi-task Uncertainty-based Loss . . . 25

3.3.3 Multi-task Class Regularization . . . 27

3.4 Experimental Results . . . 29

3.4.1 Multi-Attribute Food Datasets . . . 29

MAFood-121. . . . 29

VIREO Food-172. . . . 31

3.4.2 Metrics . . . 31
3.4.3 Experimental setup . . . 32
3.4.4 Results . . . 33
3.5 Conclusions . . . 37

4 Uncertainty-aware hybrid classifier for food recognition 39
4.1 Introduction . . . 39
4.2 Related Work . . . 41
4.2.1 Food recognition . . . 41
4.2.2 Uncertainty estimation . . . 42
4.3 Method for Uncertainty-based Hybrid Classification . . . 43
4.3.1 Prediction by integrating local and flat classifiers . . . 43
4.3.2 Epistemic uncertainty estimation . . . 44
4.3.3 CNN Model . . . 45
4.4 Experiments . . . 45
4.4.1 Datasets . . . 45
4.4.2 Experimental setup . . . 46
4.4.3 Results . . . 47
4.5 Conclusions . . . 51

5 Uncertainty-aware data augmentation for food recognition 53
5.1 Introduction . . . 53
5.2 Related Work . . . 55
5.2.1 Food Recognition . . . 55
5.2.2 Data Augmentation Strategies . . . 56
5.2.3 Uncertainty Modeling . . . 57
5.3 Uncertainty-based Data Augmentation Methods . . . 58
5.3.1 Class-Conditional Data Augmentation . . . 58
5.3.2 Uncertainty-Aware GAN-augmented Food Recognition . . . 60
5.3.3 Uncertainty-Aware Data Augmentation . . . 61
5.4 Validation . . . 63
5.4.1 Food datasets . . . 63
Food-101 . . . 63
UECFOOD-256 . . . 63
MAFood-121 . . . 63
5.4.2 Evaluation metrics . . . 64
Metrics for CCDA . . . 64
Metric for UAGAN . . . 64
Metrics for UDA . . . 65
5.4.3 Experimental Setup . . . 65

Specific setups for CCDA 65
Specific setups for UAGAN 66
Specific setups for UDA 67
5.4.4 CCDA Results 68
5.4.5 UAGAN Results 69
5.4.6 UDA Results 72
5.5 Conclusions 76

6 Exploring Food Detection using CNNs 77
6.1 Introduction 77
6.2 Methodology for food detection 78
6.2.1 GoogLeNet for feature extraction 79
6.2.2 PCA for feature selection 79
6.2.3 SVM for classification 79
6.3 Experiments 80
6.3.1 Datasets 80
6.3.2 Experimental Setup 81
6.3.3 Metrics 82
6.3.4 Results 82
6.4 Conclusions 84

7 Food Recognition using Fusion of Classifiers based on CNNs 85
7.1 Introduction 85
7.2 Methodology 87
7.2.1 Training of CNN models 87
7.2.2 Decision templates for classifiers fusion 87
7.3 Experiments 88
7.3.1 Datasets 88
7.3.2 Experimental Setup 89
7.3.3 Data Preprocessing and Metrics 90
7.3.4 Experimental Results on Food-11 91
7.3.5 Experimental Results on Food-101 92
7.4 Conclusions 94

8 Grab, Pay and Eat: Semantic Food Detection for Smart Restaurants 97
8.1 Introduction 97
8.2 Related Work 99
8.3 Semantic Food Detection 102
8.3.1 Food Segmentation 102
8.3.2 Food Detection 103

8.3.3 Semantic Food Detection 104
Background Removal 104
Non-Maximum Suppression 105
8.4 Experimental Results 105
8.4.1 Dataset 106
8.4.2 Food Segmentation 106
8.4.3 Semantic Food Detection Performance 108
8.5 Conclusions 113

9 Conclusions and Future Lines **115**
9.1 Future works 117

Bibliography **119**

List of Figures

1.1 Example of high intra-class (first two rows, left-right) variability for *com tam* and *coq au vin* foods; and low inter-class (last two rows, bottom-top) variability for five different foods pairs. 2
1.2 Pipeline diagram illustrating how each food-related problem addressed relates to different approaches designed during thesis work. In addition, the building of a large food dataset is highlighted as a cross-cutting task. 5

2.1 A cartoon drawing of a mathematical model representing a biological neuron. Preprint [79] 10
2.2 Illustration of a feedforward neural network architecture. 10
2.3 Example of Common Neural Network (left) vs Bayesian Neural Network (right). Preprint [19] 12
2.4 LeNet architecture for document recognition. Preprint [96] 13

3.1 Food-related tasks predicted by our model. 20
3.2 Regularized Uncertainty-based Multi-Task Learning model for classification of food-related tasks. 23
3.3 Examples of images from the MAFood-121 dataset divided in the 11 different cuisines considered. 30
3.4 Success (top 3) and failure (bottom) cases of RUMTL on MAFood-121. . . 35
3.5 Success (top 3) and failure (bottom) cases of RUMTL on Vireo-172. 35
3.6 From left to right: Loss values and task uncertainty (σ^2) for the dish, cuisine and food families tasks obtained during the training of the RUMTL model on MAFood-121. 36
3.7 Loss values and task uncertainty (σ^2) for the dish (left) and ingredients (right) tasks obtained during the training of the RUMTL model on VIREO Food-172. 37

4.1 Illustration of a class hierarchy for a Local Classifier per Parent Node and Flat Classifier approaches on MAFood-121 dataset. A, A1-A11 and B denote the classifiers for the nodes shown within each rectangle. 40

4.2 Main scheme of the proposed method, which shows the procedure performed when the predictions in the first level of the hierarchy are equal to 1. The suffix *wod* denotes the prediction with the dropout turned off, *wd* denotes the prediction with the dropout turned on, the terms *SO*, *MSO* and *EU*, denote Softmax Output, Mean Softmax Output and Epistemic Uncertainty. 42
4.3 Samples of the Smallest and Largest EU within the same food class. 48
4.4 CDF for the entropy of the predictive distributions on the out-of-distribution (1st & 2nd row) and in-of-distribution (3rd & 4th row) cuisines on MAFood-121. 49

5.1 Example of high intra-class (left-right) and low inter-class (bottom-top) variability in food images for prime_rib (top) and steak (bottom) classes. . 54
5.2 Flowchart of the proposed class-conditional data augmentation procedure. SDA_model_D denotes the model which incorporate the Dth specific data augmentation technique (eg. color, contrast) into the generic data augmentation process. 58
5.3 Main scheme of the our UAGAN food recognition method. 60
5.4 An example of the resulting image when we apply the minimal (top) and maximal (bottom) values used in the experimentation for each strategy corresponding to the image enhancement methods. 67
5.5 Sample of the synthetic images from the generator applied. 68
5.6 From left to right: classes selected of UECFOOD-256 to apply adjustment of: a) brightness, b) color, c) contrast and d) sharpness. The class identifier (c_{id}) can be obtained for the formula: $c_{id} = 16 * col + row$, where the first position (upper-left corner) corresponds to $(row, col) = (0, 0)$. 69
5.7 Histogram for the entropy of the predicted images. 70
5.8 Training images vs epistemic uncertainty. 71
5.9 Synthetic image generated on the selected images from the training set. . . 72
5.10 Histogram of the entropy after each training cycle when training with UDA (S_4). 73
5.11 Number of synthetic images generated after the third training cycle. . . . 74

6.1 Method overview for our food detection approach. 78
6.2 Example of images contained in the FCD Dataset. Top row shows food images from Food101 and bottom row shows non-food images from Caltech256. 81
6.3 Example of images contained in the RagusaDS Dataset. Top row shows food images from UNICT-FD889, middle row shows food images from Flickr-Food, and bottom row shows non-food images from Flickr-NonFood. 81
6.4 FP (top) and FN (bottom) on RagusaDS dataset. 83

6.5 FP (top) and FN (bottom) on FCD dataset. 84

7.1 General scheme of our CNNs Fusion approach. 86
7.2 Images from the Food-11 dataset. Each image corresponds to a different class. . . 89
7.3 Example images of Food-101 dataset. Each image represents a dish class. 89
7.4 Misclassified Food-11 examples: predicted labels (on the top), and the ground truth (on the bottom). 92
7.5 Misclassified examples for the Food-101 classes that obtained the worst (steak) and best (edamame) classification results by F1 score (groundtruth label - bottom). 94

8.1 Example of images used in traditional approaches to food analysis (left) and food tray analysis (right). 98
8.2 Main tasks of our Semantic Food Detection framework. 98
8.3 Detailed workflow of the proposed Semantic Food Detection method: food segmentation and food detection methods are applied in parallel, before combining them for a final detection on food tray images. 102
8.4 Food segmentation output (left to right): binary prediction by FCN, regions boundaries and food bounding boxes. 103
8.5 A representative sample of the UNIMIB dataset [32]: original image (left) and food annotations (right). 106
8.6 a) Distribution of the trays according to the number of foods that are placed in them. b) Results in terms of *Recall* (blue) and *TA* (orange), for each item of the distribution. 111
8.7 Results of our proposal when varying the background removal (BR) and non-maximum-suppression (NMS) thresholds. 112
8.8 Some samples of the results obtained using the proposed approach, from left to right: food tray with all the objects correctly detected (blue), false detection sample (orange), and two samples with one misclassified object (red). 113

List of Tables

3.1 Results of the different variants of our model compared to the use of single-task models on MAFood-121 dataset. 34
3.2 Results of our proposed model compared to the use of single-task models on VIREO-172 dataset. 34

4.1 Advantages and disadvantages of traditional classification approaches and our proposal. 41
4.2 Cuisine classification results on MAFood-121 in terms of accuracy taking into account only the minimum entropy criterion. 47
4.3 Results on MAFood-121 in terms of Accuracy. 48
4.4 Results on Food-101 in terms of Accuracy. 50
4.5 Accuracy of our proposal for both datasets, according to the criteria applied, using the Deep Ensemble method to estimate uncertainty. 51

5.1 Results on Food-101 in terms of Acc and NEU for the models trained with different data augmentation techniques. For NEU, less is better. 70
5.2 Results on UECFOOD-256 in terms of Acc and NEU for the models trained with different data augmentation techniques. For NEU, less is better. . . . 70
5.3 Results obtained on the test set in term of accuracy with the standard deviation. 71
5.4 Synthetic images added after each training cycle. We mark with * the last images added for each selected model. 72
5.5 Results obtained on the test sets in terms of R_{micro}. 74
5.6 Results obtained on the test sets in terms of R_{macro}. 75
5.7 Results obtained on the test sets in terms of $|R_{micro} - R_{macro}|$. 75

6.1 Results obtained by models based on CNN on RagusaDS and FCD datasets on the food detection task. All results are reported in %. 82
6.2 Results obtained when GoogLeNet + PCA-SVM is trained on both datasets together (RagusaDS+FCD) and evaluated separately and jointly. 82

7.1 Overall test set error rate of Food-11 obtained for each model. The distance measure is shown between parenthesis in the CNNs Fusion models. . . . 91

7.2 Percentage of images well-classified and misclassified on Food-11 using our CNNs Fusion approach, distributed by the results obtained with GoogLeNet (CNN_1) and ResNet50 (CNN_2) models independently evaluated. 92
7.3 Some results obtained on the Food-11 using our CNNs Fusion approach. 93
7.4 Overall test set accuracy of Food-101 obtained for each model. 93
7.5 Percentage of images well-classified and misclassified on Food-101 using our CNNs Fusion approach, distributed by the results obtained with InceptionV3 (CNN_1) and ResNet50 (CNN_2) models independently evaluated. 94
7.6 Top 3 better and worst classification results on Food-101. 95

8.1 Results obtained with our Food Segmentation approach in test set. 108
8.2 Results obtained by YOLOv2 and the proposed approach in training set using different confidence thresholds. 110
8.3 Tray Food Analysis results, from top to bottom: food detection method 1) without segmentation, 2) with segmentation, and 3) with ground-truth segmentation to perform the recognition. The best results per block are in boldface. 110

List of Abbreviations

Acc	Overall **Acc**uracy
ANN	**A**rtificial **N**eural **N**etworks
BNN	**B**ayesian **N**eural **N**etwork
BR	**B**ackground **R**emoval
CCDA	**C**lass-**C**onditional **D**ata **A**ugmentation
CDF	**C**umulative **D**istribution **F**unction
CEU	**C**lass **E**pistemic **U**ncertainty
CO	**Co**vering
CNN	**C**onvolutional **N**eural **N**etwork
DenseNet	**Dense** Convolutional **Net**work
DP	**D**ecision **P**rofile
DT	**D**ecision **T**emplate
EDL	**E**vidential **D**eep **L**earning
EU	**E**pistemic **U**ncertainty
FB	**F**ull **B**ayesian
FC	**F**ully **C**onnected
FCN	**F**ully **C**onvolutional **N**etwork
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
GA	**G**lobal Pixel **A**ccuracy
GAN	**G**enerative **A**dversarial **N**etwork
GDA	**G**eneric **D**ata **A**ugmentation
GNN	**G**raph **N**eural **N**etwork
GT	**G**round **T**ruth
IoU	**I**ntersection **o**ver **U**nion
ILSVRC	**I**mageNet **L**arge **S**cale **V**isual **R**ecognition **C**hallenge
KL	**K**ullback-**L**eibler
LCC	**L**ocal **C**hild **C**lassifier
LCL	**L**ocal **C**lassifier Per **L**evel
LCN	**L**ocal **C**lassifier Per **N**ode
LCPN	**L**ocal **C**lassifier Per **P**arent **N**ode
LPC	**L**ocal **P**arent **C**lassifier
MAA	**M**acro **A**verage **A**ccuracy
MAP	**M**aximum **a** **P**osteriori
MC	**M**onte **C**arlo
ML	**M**ulti-**L**abel
MLE	**M**aximum **L**ikelihood **E**stimation
MLP	**M**ulti-**L**ayer **P**erceptron
MSMVFA	**M**ulti-**S**cale **M**ulti-**V**iew **F**eature **A**ggregation

MTA	**M**ulti-**T**ask **A**ccuracy
NMS	**N**on-**M**aximum-**S**uppression
NEU	**N**ormalized **E**pistemic **U**ncertainty
PCA	**P**rincipal **C**omponent **A**nalysis
Pre	**Pre**cision
Rec	**Rec**all
ResNet	**Res**idual **Net**work
RUMTL	**R**egularized **U**ncertainty-based **M**ulti-**T**ask **L**earning
SA	**S**tandard **A**ccuracy
SDA	**S**pecific **D**ata **A**ugmentation
SGD	**S**tochastic **G**radient **D**escent
SoA	**S**tate-**o**f-the-**A**rt
SL	**S**ingle **L**abel
SLP	**S**ingle-**L**ayer **P**erceptron
SVM	**S**upport **V**ector **M**achine
TA	**T**ray **A**ccuracy
TP	**T**rue **P**ositive
TPr	**T**rue **P**ositive **r**ate
TN	**T**rue **N**egative
TNr	**T**rue **N**egative **r**ate
UAGAN	**U**ncertainty-**A**ware **GAN**-augmented
UDA	**U**ncertainty-Aware **D**ata **A**ugmentation
VI	**V**ariation of **I**nformation
WRN	**W**ide **R**esidual **N**etwork

Dedicated to my wife Carolina Andreu, my son Nicolás Aguilar and my daughter Ágata Aguilar who will soon be born.

Chapter 1

Introduction

In this chapter, we explain the problematic in the automatic food analysis from images, which is a particular fine-grained object recognition case and stands out for having both properties: high intra-class variability and high inter-class similarly, becoming a challenging problem for the Computer Vision community. In addition, we describe the importance of research in this field focusing on possibles applications in the real life. Finally, we present the work objectives and the research contributions.

1.1 Problem statement

Food is an essential substance for people and animals life, consisting of nourishing and nutritive components. It can be simply understood as a substance that provides energy to our vital engine. However it is much more than just a source of energy. Food plays an important role in our life, not only in terms of nutrition, but also became essential in any social activity such as events, meetings, celebrations, etc. In addition, depending on the type of food and the situation in which we eat it can influence our mood and sometimes brings back memories of events that occurred in our lives.

Automatic analysis of food from images has been an emerging field of research that has recently attracted the computer vision community. Visual food analysis includes several tasks mainly with aims of extracting semantic information, from images, at different levels of abstraction. The simplest task, called food detection, claims to determine whether or not food is present in an image. Food recognition is one of the main tasks and aims to classify food images with a single or multiple categorization with respect to the food items or ingredients that compose them. When classification is done pixel by pixel, this task is known as food segmentation. On the other hand, food location is the corresponding task to determine where food items are placed in an image. To name a few of the most important tasks within this field of research.

Independently of the target food-related task, the difficulty in analyzing food images lies in the intrinsic properties of the food appearance. Unlike general objects, most of foods

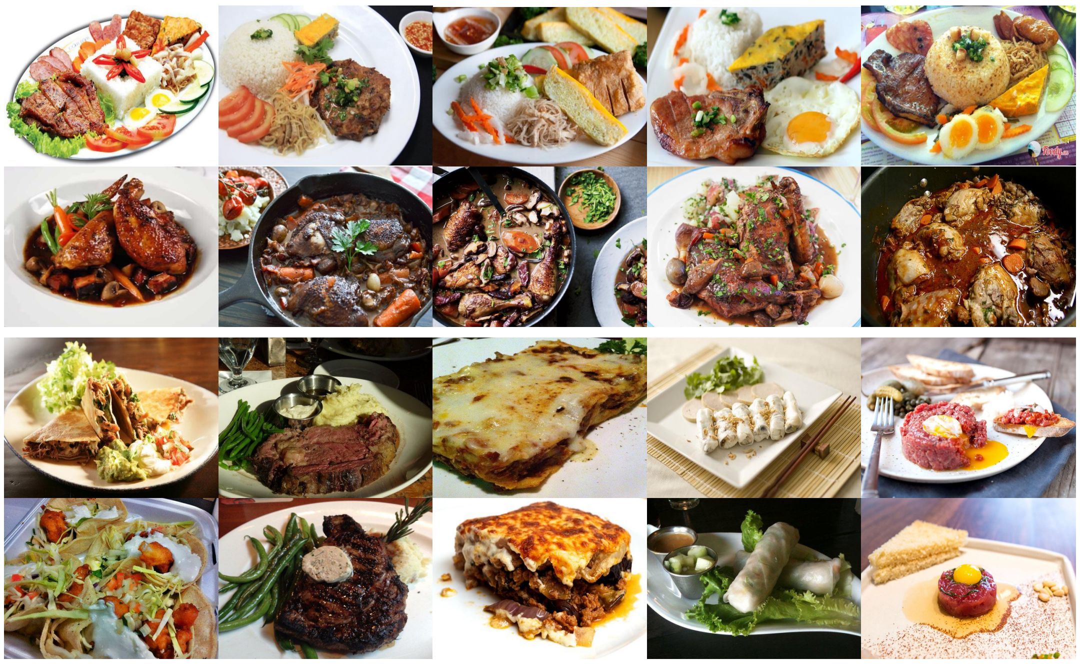


FIGURE 1.1: Example of high intra-class (first two rows, left-right) variability for *com tam* and *coq au vin* foods; and low inter-class (last two rows, bottom-top) variability for five different foods pairs.

are no rigid-objects and therefore do not have a well-defined spatial arrangement and shape. Furthermore, food images are characterized by larger food deformations, high intra-class variability and high inter-class similarity, resulting in a visually sophisticated problem to solve. In Fig. 1.1 the complexity of food images due to its intrinsic properties is illustrated.

With the emergence of deep learning algorithms, and particularly Convolutional Neural Networks (CNNs), there is a "before and after" in the computer vision. There are two main factors that made the emergence of deep learning algorithm possible: improvement in hardware capabilities and availability of large data volumes. The latter is quite relevant because these types of algorithms are prone to over-fitting when the data is not enough. Therefore, to ensure good performance, thousands of data are needed to ease the model learning with respect to the data distribution and thereby better generalize to new data. When the data is limited, incorporation of regularization into the model architecture, transfer learning from a trained model on a large dataset or training with data augmentation strategies have been applied to minimize the risk of over-fitting.

In recent years, promising performance has been demonstrated with deep learning algorithms in a wide range of solutions applicable in real conditions. Some remarkable results have been achieved industrial systems, such as: autonomous vehicles, security and surveillance, healthcare, among others. Visual food analysis follows the same trend, where a huge increase in performance has been shown when comparing traditional, hand-crafted features based methods, with deep learning methods. However, it is still a long way to go to achieve a performance comparable with general objects.

Making a real food analysis solution requires thousands of different foods with thousands of images in order to train a deep learning algorithm for this purpose. Therefore, taking into account the difficulty of data acquisition and the visual complexity present in food images, it is necessary to explore novel strategies to better understand what the model learns and, from this, take steps to guide it. Here, uncertainty modeling can play an important role in studying the behavior of the model to further improve its performance.

In this thesis, our research covers the four food analysis tasks discussed above and delves into uncertainty modeling theory to provide novel methods applicable to various food-related systems in real-life settings. Some of the most relevant fields of application can be seen in the next section.

1.2 Food Analysis Relevance

The computer vision community has highlighted numerous applications in which food analysis from images has a positive impact. Most researchers emphasize its importance in healthcare, but it is also useful in the restaurant industry, for recommendation systems, social media tagging and more.

1.2.1 Food for Healthcare

In such fast-paced world, a fast food goes a long way in saving time. However, frequent consumption can increase the risk of health problems. The fact of ignoring the implication that the ingested food produces in the body, increases the chances that it may give rise to any disease or condition caused by a poor diet, such as: obesity, diabetes, heart problems, kidney problems, among others. To address this, a possible solution based on the visual food analysis would be to generate a system that allows managing and controlling food intake in order to be aware of the nutritional contribution of the food consumed and thus be able to maintain a healthy diet. This system would bring both preventive and corrective benefits for problems related to bad diet. Furthermore, it could also be useful for anyone who must restrict their food intake, such as people with food allergies, celiac disease, food intolerance or after surgery (e.g. organ transplant).

1.2.2 Restaurant Industry

One commercial segment that can take advantage of the visual food analysis is the self-service restaurant. This type of restaurant offers people the possibility of choosing or taking for themselves the food they want to consume, from a variability of food alternatives grouped into main dishes, side dishes, desserts and drinks; and depending on what is served a cashier bills it. Visually determining the food served can speed up the billing process, which means a bottleneck at peak times, in addition to extracting useful information to replace stocks, among other benefits.

1.2.3 Recommender Systems

Interesting applications can be developed for the recommender systems based on food analysis. Basically, this type of systems seeks to predict the preference of a user give for items, based on their own data or on a match with a profile created *a priori* extracted from the data of other users, in order to suggest items that are relevant to him/her. In this sense, an example of a solution could be a visual food-log system, in which the food consumption preference can be determined and, from this, provide recommendations to the user on where to eat or suggest recipes according to ingredients available in the fridge.

1.2.4 Social Networking Tools

As well-known, social networking has became an essential tool for our daily life. In 2020, 84% of people with access to the internet use social media and spend more than 2 hours on them. These tools have made it possible to reach people all over the world and maintain contact with people that they have directly or indirectly shared with use throughout our lives. Among the different functionalities it offers, some social networks have incorporated an automatic tagging function in their systems that allows the user to group or filter their photos in relation to the visual content. Here, an object recognition, and particularly food recognition, plays a key role in carrying out this task. A clear example of the use of food recognition can be seen in Google Plus, which can automatically tag the visual media through Google Photos.

1.3 Objectives

The main objective of this thesis is to develop and implement Deep Learning algorithms based on CNNs to analyze the visual information contained in food images acquired in real-life conditions. To meet this general objective, the specific objectives are defined as follows:

- To review and analyze the state-of-the-art (SoA) in food-related deep learning approaches.
- To develop novel food detection, recognition, localization and segmentation algorithms based on CNN.

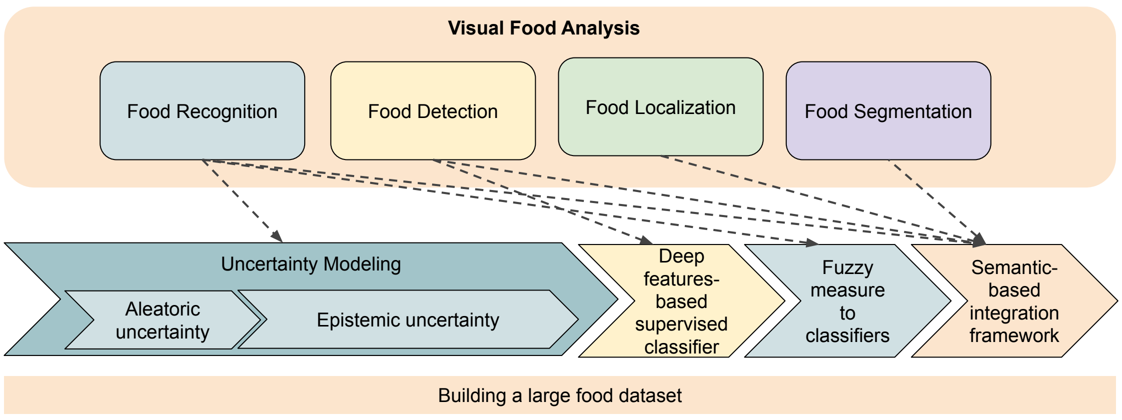


FIGURE 1.2: Pipeline diagram illustrating how each food-related problem addressed relates to different approaches designed during thesis work. In addition, the building of a large food dataset is highlighted as a cross-cutting task.

- To deep dive into Uncertainty modeling theory in Deep Learning and provide several techniques leveraging different aspects of the uncertainty to improve the food analysis performance.
- To build a large database with a wide diversity of food images from all over the world.
- To contrast and validate the results achieved in each step contemplated within the visual food analysis process.

1.4 Research Contributions

In this thesis, we address the problem of visual food analysis mainly taking into account four sub-tasks: food detection, recognition, location and segmentation (see Fig. 1.2); making a huge variety of applications possible. To tackle each one, approaches based on CNN were considered. Specifically in the food recognition case, uncertainty modeling was integrated into the solutions. The main contributions can be summarized as follows:

1. **Modeling aleatoric uncertainty in a multi-task food recognition method** (see chapter 3). We extend the model for Multi-Task Learning with Homoscedastic Uncertainty [83] to allow dynamically weighing single-label and multi-label task losses in the food classification problem. Furthermore, we incorporate a custom regularization term in the loss function that considers negative and positive correlations between classes that belong to different tasks. Moreover, we publish an international food dataset comprised of multiple food-related tasks with more than 20.000 images and multi-task annotation, and a new metric for measuring tasks coherence.

The efficiency of the proposed approach was validate in two public food datasets. The results shown in this chapter are published in the following journal [3]:

E. Aguilar, M. Bolaños, and P. Radeva. "Regularized Uncertainty-based Multi-task Learning Model for Food Analysis." Journal of Visual Communication and Image Representation, 2019. ***(Scopus: Q1, ISI: Q2)***

2. **Hybrid food classifier based on epistemic uncertainty analysis** (see chapter 4). We provide an epistemic uncertainty-based framework, which reduces parent-to-child errors present in the Local Classifier per Parent Node (LCPN) approach, through the proposed criteria to judge when an LCPN or Flat approach is used to give the final prediction. Three different methods were considered to estimate epistemic uncertainty, these are: MC-dropout, Evidential Deep Learning and Deep Ensemble. For all of them, a considerable reduction of parent-to-child errors propagation, as well as improving classification performance with respect to the Flat approach was shown within our proposal. The results shown in this chapter is published in in the PRL journal and a conference proceeding [5]:

   E. Aguilar, and P. Radeva. "Uncertainty-aware Integration of Local and Flat Classifiers for Food Recognition." Pattern Recognition Letters, 2020. (in press) ***(Scopus: Q1, ISI: Q2)***

   E. Aguilar, and P. Radeva. "Food Recognition by Integrating Local and Flat Classifiers." Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA), 2019. ***(CORE ranking: C)***

3. **Uncertainty-aware data augmentation methods for food recognition** (see chapter 5). We explored three different data augmentation strategies, based on uncertainty analysis, conditioned at the class or sample level. In the first, we propose the application of the uncertainty analysis in order to identify the type of data augmentation that can be more beneficial or detrimental for each particular class (eg. some classes may depend on the color feature and changing it could not contribute). In the second, we suggested that some images may be more complex to recognize than others, so we proposed a method that provides more synthetic samples with the same visual content of the complex images, to help the learning of these features. In the last one, we delved into the data augmentation at the sample level, proposing a training procedure following the same principles of an active learning framework. However, instead of selecting new data, the procedure contemplates the generation of new synthetic data and its subsequent incorporation into the training set, after each training cycle, taking into account the images that met the proposed uncertainty-based threshold. The results shown in this chapter are published in the conference proceeding [4][7] and submitted to the ICPR'20:

E. Aguilar, and P. Radeva. "Class-conditional Data Augmentation Applied to Image Classification." International Conference on Computer Analysis of Images and Patterns (CAIP), 2019. ***(CORE ranking: B)***

E. Aguilar, B. Nagarajan, R. Khatun, M. Bolaños, and P. Radeva. "Uncertainty Modeling and Deep Learning Applied to Food Image Analysis." International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC), 2020.

Aguilar, B. Nagarajan, R. Khatun, M. Bolaños, and P. Radeva. "Uncertainty-aware Data Augmentation for Food Recognition." E. International Conference on Pattern Recognition (ICPR), 2020. (Submitted) ***(CORE ranking: B)***

4. **New method to perform food detection from images** (see chapter 6). We contribute in food detection problem, also called food/non-food classification, by establishing the state-of-the-art in public datasets, and further delving into the classification method, rather than simply using a CNN object recognition model previously trained on Imagenet and fine-tuned on new data. Specifically, we propose a method that extracts the features from a CNN model, applies PCA for the feature selection and finally SVM the classification; instead of directly using softmax for classification. Using this approach, previous results in public food/non-food datasets were improved. The results shown in this chapter are published in a conference proceeding [2]:

   E. Aguilar, M. Bolaños, and P. Radeva. "Exploring Food Detection Using CNNs." International Conference on Computer Aided Systems Theory (EUROCAST), 2017.

5. **Fuzzy measures to ensemble food recognition models** (see chapter 7). We analyze how different fuzzy measures of similarity can contribute to the combination of CNN models through a late fusion strategy based on the softmax outputs of each one. Specifically, a Decision Template scheme was adopted in the proposed method for fusing classifiers. In experimentation, we validated our proposal by combining two CNN models based on different architectures, but it is flexible to any number of models. The benefits of this approach were shown in two public food datasets, where the baseline results were outperformed by a wide range. The results shown in this chapter are published in a conference proceeding [1]:

   E. Aguilar, M. Bolaños, and P. Radeva. "Food Recognition Using Fusion of Classifiers Based on CNNs." International Conference on Image Analysis and Processing (ICIAP), 2017. ***(CORE ranking: B)***

6. **Novel framework for semantic food detection in food tray images** (see chapter

8). We propose a novel approach that unifies the problems of food detection, localization, recognition and segmentation into a new framework that we call Semantic Food Detection. The method integrates food/non-food semantic segmentation with simultaneous food location and classification techniques through a probabilistic approach and a custom non-maximum suppression procedure. This framework was validated in a dataset containing images of canteen stock food trays, where we evidenced a large increase in performance and reduction of false positives with respect to the state-of-the-art methods. The results shown in this chapter are published in a journal [6]:

E. Aguilar, B. Remeseiro, M. Bolaños, and P. Radeva. "Grab, Pay, and Eat: Semantic Food Detection for Smart Restaurants." IEEE Transactions on Multimedia, 2018. ***(Scopus: Q1, ISI: Q1)***

1.5 Thesis Organization

This thesis begins in the next chapter by providing an overview to familiarize yourself and better understand some basic concepts that support the developed methods. In Chapter 3, we are entering the uncertainty modeling and describing a novel multi-task recognition method. In Chapter 4 and Chapter 5, we deep dive into uncertainty modeling theory and describe new food recognition methods based on it. Chapter 6 to Chapter 8 are devoted to the problems of food detection, ensemble of food recognition methods and semantic food detection, where we detail the approach proposed for each one. Finally, Chapter 9 highlights the final conclusions and possible future research lines.

Chapter 2

Background

In this chapter, we briefly describe the basic concepts necessary to better understand the fundamental foundation on which the proposed models are supported. The objective is to familiarize with these concepts to facilitate the reading of the rest of the thesis.

2.1 Neural Networks

Inspired by the structure and performance of our biological neural networks, Artificial Neural Networks (ANN) is a computational model consisting of basic computing units can mimic some of the feature of the biological networks [177]. The fundamental cellular unit of the brain is a neuron. A neuron is comprised mainly of three parts: a cell body, an axon, and dendrites [163, 177]. The dendrites receive signal from other neurons. The cell body contains the nucleus and cytoplasm, and is the place where signals from other neurons are received and combine through the dendrites by synaptic junctions (synapse). The axon transmits the resulting neural activity to other nerve cells or muscle fibers. The matching of the different parts of a biological neuron with respect to an artificial neuron is illustrated in Fig. 2.1. Let us consider the input features x and a weight matrix w, then in this case, the multiplicative interaction of a signal transmitted from the axons (e.g. x_0) to the dendrites of the other neuron by synapse (e.g. w_0) correspond to the weighting of the i-th feature x_i with respect to the i-th weight w_i (e.g. $w_0 x_0$). All received signals ($\{w_i x_i\}$) are combined in the cell body ($\sum_i w_i x_i + b$) and then sent to the output axon through an activation function ($f(\sum_i w_i x_i + b)$) representing the propagation frequency of the signal along this axon.

A single artificial neuron (unit) is capable of learning linearly separable data and can be applied as a binary classifier algorithm. Specifically, this is the case of the perceptron algorithm [139] that behaves as a threshold function; the output of the axon passes through a binary step activation function that gives a positive response when the value is greater than 0 and a negative response, otherwise.

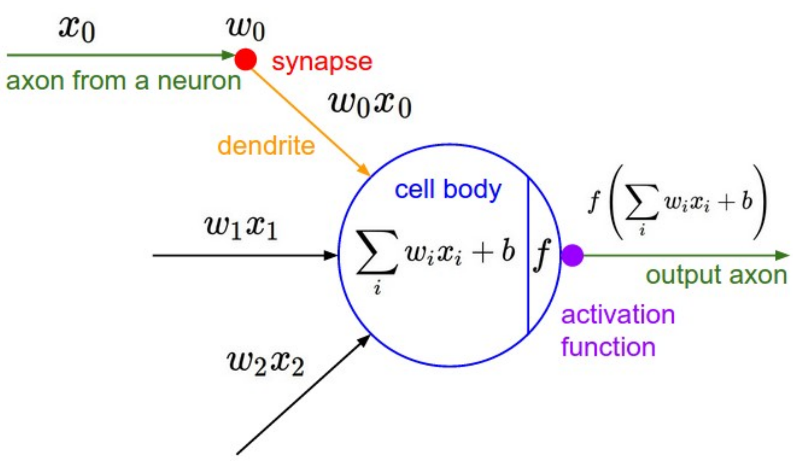


FIGURE 2.1: A cartoon drawing of a mathematical model representing a biological neuron. Preprint [79]

ANN are typically arranged into three types of computing units blocks (layers): input layer, which contains the patterns that are passed to the next layer; hidden layers, where intermediate processing of signals passed from the previous layer to feed the next layer is performed; and output layers, containing the model response in the desired format for the target problem. The simplest type of ANN is the feedforward neural network (see Fig. 2.2). It consists of an ANN with one or more layers that contain one or more neurons for each one. These layers are fully-connected in which the units from adjacent layers are fully pairwise connected through the respective weights. The signal goes from the input to the output layer in one direction and there is no cycle in the network. When the feedforward neural network has only one layer (the input layer is not counted) it is called single-layer perceptron (SLP); otherwise, it is called a multi-layer perceptron (MLP). The major difference regarding learning ability is that, unlike SLP, MLP is able to learn a nonlinear data distribution.

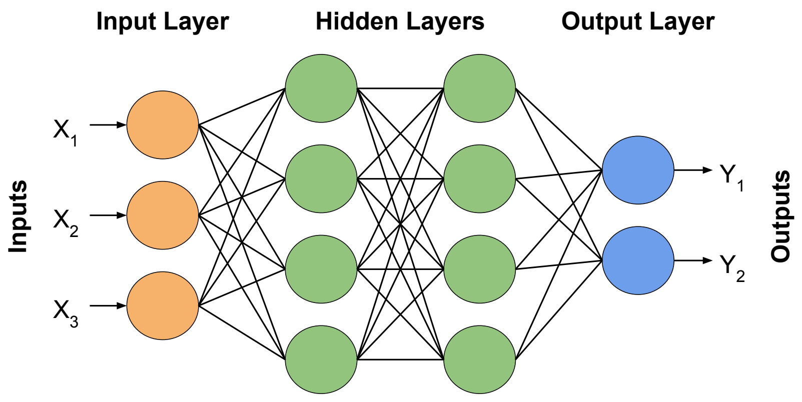


FIGURE 2.2: Illustration of a feedforward neural network architecture.

The supervised learning process in a MLP occurs after each iteration through a forward and backward pass. During the forward pass, the new input pattern is processed through all units from the first to the last layer. After that, a loss function is calculated taking into account the resulting values in the output layer. Then, a backward pass (backpropagation) is made from the last to the first layer propagating the error and adjusting the weights. Therefore, the training process can be interpreted as an optimization problem that seeks to minimize the values of the loss function after each iteration. For this purpose, the most dominant method for optimizing the loss function is the Stochastic Gradient Descent (SGD) [137], a stochastic (random) approximation of the Gradient Descent optimization method, which updates the ANN weights and bias following steps proportional to the negative of the loss function gradient at the current training iteration from a randomly selected subset of data, rather than the entire data set. In ANN, the steps are known as the learning rate and in the SGD case they remain fixed for all hyper-parameters to be optimized. Formally, the update of the hyper-parameters (θ) considering the loss function $L(\theta)$ and the learning rate α is given as:

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial}{\partial \theta_t} L(\theta_t).$$

In ANN, the loss function has a non-convex shape that, in most cases, the error surface in the weight space corresponds to a complex hyperparaboloid with numerous valleys and hills. For this reason, often the solution reached when the loss function converges, corresponds to a local minimum and not the best possible general solution. This is easy to see when training multiple ANN by initializing random weights and analyzing their results, in the same data. In most cases, both overall performance and some misclassified data will be different. Providing a single prediction of an ANN that reached a local minimum, which is often susceptible to overfitting, may not be reliable enough for critical systems. Bayesian Neural Network can help us provide a confidence value for our prediction and also allows us to better understand the model behavior.

2.2 Bayesian Neural Networks

The Bayesian Neural Network (BNN) is a type of ANN that learns a probability distribution in the weights of each computing units using Bayesian inference, rather than learning a single value (see Fig. 2.3). This type of network combines the strength of the probabilistic (stochastic) and ANN models, and thus produces probabilistic guarantees about their predictions and also generates the distribution of the parameters that it has learned from the input data [123]. The main differences of BNN with respect to the standard ANN are: 1) the objective of the training is to learn, for each computing units, a distribution overs weights and not to find a function that learns the data distribution on a single weight, 2) the output corresponds to a distribution and not a single value from the best model, and 3) it models uncertainty and can provide a confidence value in addition to the prediction.

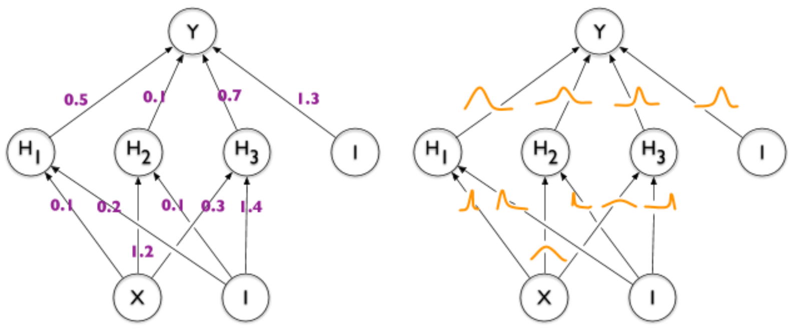


FIGURE 2.3: Example of Common Neural Network (left) vs Bayesian Neural Network (right). Preprint [19]

From the probabilistic modeling side, the optimization process in a standard ANN is equivalent to Maximum Likelihood Estimation (MLE) for the weights, where the aims is to find the parameters (w) that maximize the probability ($p(y|w)$) of the observed data. However, in probabilistic models, it assumes that the model parameters have some distribution according to a prior belief ($p(w)$). The posterior probability ($p(w|y)$) corresponds to the beliefs about the parameters after observing the data, which according to Bayesian rules is proportional to multiplying the likelihood observation ($p(y|w)$) with $p(w)$. An approximation of Bayesian inference can be given by the maximum a posteriori (MAP) estimate of w, which becomes the same that the MLE with an additional prior term.

In ANN, the prior term can be incorporated into the loss function as $L2$ or $L1$ regularization that is equivalent to performing the MAP using a prior Gaussian or Laplace distribution respectively. To illustrate the described above, let us consider an ANN with a softmax activation on the output layer to perform a classification of the data $D = \{x_i, y_i\}$. Then, the posterior probability on the dataset is defined as:

$$p(w, D) = \prod_i p(y_i|x_i, w)p(w),$$

where $p(y_i|x_i, w)$ corresponds to the probability that the softmax output y_i belongs to the data x_i given the weights w. Let us assume that prior distribution $p(w)$ is a Gaussian distribution with mean equal to 0 and variance equal to $\frac{1}{2\sigma^2}$, then:

$$p(w, D) = \prod_i p(y_i|x_i, w)\mathcal{N}(w, 0, \frac{1}{2\sigma^2}).$$

Taking negative log probability, then:

$$-log(p(w, D)) = -\sum_i log(p(y_i|x_i, w)) + \sum_i w_i^2 + c.$$

Finally, we can drop the constant c because it does not affect the optimization and with it we obtain the categorical cross entropy loss plus $L2$ regularization.

Despite being a direct solution that can easily incorporate the probabilistic approach within an ANN, with MAP we cannot know the prediction uncertainty. To tackle with this, the Full Bayesian (FB) approach can be used. However, in the ANN the resolution of a high-dimensional integral associated with the posterior predictive distribution ($p(y|x,D) = \int p(w|D)p(y|x,w)dw$) becomes intractable.

Note that the standard ANN and BNN are not applicable or not accurate for image analysis. Therefore, in Section 2.3 we will describe a CNN, which is a specified network for this type of problem. In addition, in Section 2.4, we will comment how we can model uncertainty on CNN by applying some of the most popular methods to approximate a FB approach using variational inference techniques and other novel approaches.

2.3 Convolutional Neural Networks

Convolutional Neural Networks is class of Deep ANN specialized to analyze images. Early works based on it were developed in the 1990s for handwritten digit recognition [95, 96], however it was not until 2012, after offering impressive results in recognizing 1000 objects different [91], which became popular and therefore widely used in the computer vision community. In the same way as ANN, the architecture of CNN is inspired by biology, specifically, in the analysis carried out in [67] regarding the behavior of the visual cortex of cats. These networks are mainly made up of three architectural ideas [95]: local receptive fields, to extract elemental visual features such as oriented edges, end points, corners; shared weights, which optimize the computational resource usage; and spatial sub-sampling, to reduce the feature map resolution and the sensitivity of the output to shifts and distortions.

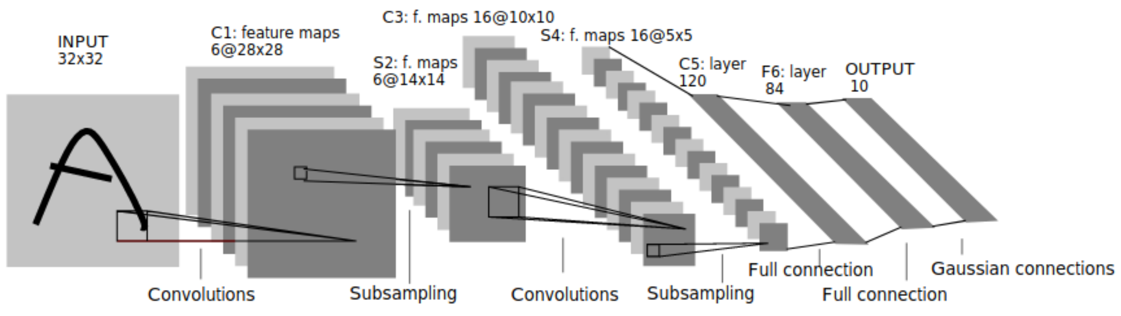


FIGURE 2.4: LeNet architecture for document recognition. Preprint [96]

The basic structure of a CNN is illustrated in Fig. 2.4. Generally, a CNN contains an input layer, one or more convolutional blocks, and ends with an MLP. In this case, the convolutional block consists of a convolutional layer followed by a subsampling layer. The convolutional layer is comprised of multiple filters (convolution kernels), with nxn receptive fields (kernel size), and weights independently for each one. As for the subsampling layer, it acts directly on to the feature maps generated by the convolutional layer, reducing their resolution through of a pooling operation like max or mean pooling. During the forward pass, the process consists of the features extraction and combining (in higher layers) by local convolution operations, frequently with a kernel size of $3x3$,$5x5$ or $7x7$, throughout the image using a sliding window technique. After that, the generated feature maps (one for each filter) are passed to a non-linear activation as a ReLU function, increasing the nonlinearity in the data, enhancing the features distinction and avoiding gradient issues (during optimization) that they occur when the input value is too high or low. Next, the feature maps are reduced by applying a pooling method to feed the follow layer, which can be an additional convolutional block or the MLP. In the case of MLP, the output of the subsampling layer is flattened and fully connected to the first hidden layer. Finally, the results are given by the output layer using an activation according to the target problem.

Like any ANN, for CNN training, a loss function must be designed or chosen according to the problem to be treated. For example, in the supervised image classification case, the loss function evaluate the candidate solution (output prediction) with respect to the real solution (ground-truth) and, from them, quantify the error (loss). Common single-label (SL) and multi-label (ML) image classification losses are categorial-cross entropy (L_{cce}) and binary-cross entropy (L_{bce}), respectively:

$$L_{cce}(\hat{y}, f^W(x)) = -\sum_{i}^{C} \hat{y}_i log(f^W(x)_i),$$

$$L_{bce}(\hat{y}, f^W(x)) = -\sum_{i}^{C} (\hat{y}_i log(f^W(x)_i) + (1-\hat{y}_i) log(1 - log(f^W(x)_i))),$$

where C is the number of classes or labels, $\hat{y}$ is a binary vector that represent the ground-truth and $f^W(x)$ corresponds to the prediction from the output layer after applying a softmax (for SL) or sigmoid (for ML) activation for the input image x given the weights W. During training, the W is adjusted to provide minimal loss and learn better data distribution. For this purpose, SGD and ADAM [85] are two of the most popular optimizer used to minimize the loss function in Deep Learning based methods. SGD stands out for its simplicity and generalization capacity and ADAM for its fast training speed [169]. Unlike SGD, ADAM is an adaptive optimizer that maintains, and automatically updates an individual learning rate for each hyper-parameter though the estimation of both the first (mean) and second raw (uncentered variance) moment of the gradients. For Adam, the equation that defines the hyper-parameters update is as follows:

$$\theta_{t+1} = \theta_i - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where $\hat{m}_t$ and $\hat{v}_t$ denotes the first and second raw moment corrected by an exponential decay rate.

Sometimes when the data is not enough, CNN can model a complex function that models the distribution of data providing a near-perfect result in the training set but poor in the test set. Applying a regularization to weights can help us simplify this function and improve generalizability. Here, the L2 and L1 regularization has been frequently adopted. In practical terms, regularization is an additional term that updates the result of the loss function, which forces weights to decay towards zero. That is the reason why L2 regularization is commonly referred to as *weight decay*. To illustrate this, consider a SL image classification problem, so the loss function with L2 regularization can be defined as:

$$L = L_{cce} + \lambda \sum_i w_i^2,$$

where λ is a trainable hyperparameter to control the strength of the regularization.

Modern CNN architecture maintains almost the same layer arrangement than LeNet [96], but are deeper, wider, adding new type of layers and/or residual connections. For example, in [91] a dropout layer was proposed before each fully connected layer to avoid over-fitting, randomly disabling and activating some connections between two consecutive layers during training. In the [159] case, an inception layer was proposed, where several filters with different kernel sizes are convoluted in parallel along the input (image or feature map) in order to better handle the feature extraction for objects present in multiple scales. Also, residual blocks have been incorporated into the CNN architectures (e.g. [59],[66],[181]) enabling the generation of a deeper network without loss of information. On the other hand, instead of simply making the model deeper and/or wider, in [161] an efficient approach is proposed to scale the model depending on the size of the input data.

In general terms, it is interesting to highlight the fact that CNN-based models have been successfully applied in computer vision problems, and became a better alternative than traditional ANNs, since they are characterized by preserving the spatial relationship between pixels and optimize the computing resource by sharing the parameters and learning just the weights of each filter instead of all possibles computing units connections. Particularly for image analysis, these properties allow CNN-based models to scale well to high-resolution images. Methods for modeling and quantifying uncertainty in Deep CNN are described in section 2.4.

2.4 Uncertainty in Deep Learning

Deep learning has revolutionized all kinds of industries, making it possible to create various real-world applications based on it. The models are often assumed to be accurate. However, despite providing good performances in multiple disciplines, the systems have sometimes not worked as expected. An example of this is the classification algorithm used in Google Photos to automatically tag images, which in 2015 mistakenly classified two African friends as gorillas. Another example is the fatal event occurred in 2016 by a self-driving Tesla car, where a man from Ohio was killed crashed into a tractor-trailer, because the autopilot system, based on computer vision, misinterpreted the image like a lighted sky and for this reason the brakes were not applied. Moreover, numerous researches have shown the ease of breaking the performance of the CNN models simply by applying small changes to the input data. For example, self-driven car can be bamboozled by adding some stickers on the road signs, producing an incorrect interpretation of them [45]; image classification systems can dramatically change the prediction when linear perturbation (noise) is added to the input image, despite being imperceptible to human vision [54]; among others. Therefore, accurate and robust models are required.

Uncertainty modelling into a CNN model will help us better understand what it have learned from the data and allow us to make better decisions against an unexpected input data, considering how uncertain the given predictions are. There are two main types of uncertainty one can model [82]: aleatory uncertainty, which captures noise inherent in the observation; and epistemic uncertainty, which captures our ignorance about the data distribution learned by the model from training data. Epistemic uncertainty can be reduced with enough data, while aleatory uncertainty cannot, yet it can be learned. To model uncertainty, a probabilistic approach can be applied as in BNN. However, directly applying an FB approach on Deep CNN is intractable. Variational inference provides a good alternative approach to approximate the posterior probability ($p(w|D)$) for FB models. This approach consists of finding, during the optimization, a probability density function over the weights ($q_\theta(w)$) parametrized on θ that minimizes the Kullback-Leibler (KL) divergence with respect to the exact $p(w|D)$. The resulting loss function is:

$$L(\theta) = KL(q_\theta(w)||p(w|D)),$$

then, applying Bayesian's rule in p(w | D),

$$L(\theta) = KL(q_\theta(w)||p(w)) - \int q_\theta(w) logp(D|w)dw,$$

finally, using Monte Carlo sampling, the loss for a single simulation is approximated as:

$$L(\theta) \approx \log q_\theta(w) - \log p(w) - logp(D|w).$$

Therefore, minimizing the objective ($L(\theta)$) is equivalent to finding a density function ($q_\theta(w)$) that compensates the loss with respect to our a priori knowledge about the weights distribution ($p(w)$) and the data likelihood ($p(D|w)$).

Novel approaches have been proposed to efficiently approximate Bayesian inference through variational inference for Deep Learning models [19, 86, 136, 50, 107, 121]. Most of them select a flexible distribution (e.g. Gaussian) to capture the true posterior probability, but simple enough for efficient optimization [19, 86, 121]. Others prioritize to choosing the approximate posterior distribution closest to the real distribution, rather than efficient computation, building complex distributions based on the normalization flow strategy [136, 107]. However, the simplest and efficient approach to easily incorporate to any CNN is MC-dropout [50]. The idea behind this method is that the common dropout technique frequently used to avoid over-fitting is equivalent to an approximation to the probabilistic deep Gaussian process, when dropout is applied to each weight layer. The authors define $q_\theta(w)$ as a Bernoulli distribution and show that minimizing the KL divergence is equal to minimizing the negative log-likelihood loss plus a regularization. Note that this approach only requires to keep the dropout layer active during the training and prediction phases, but does not require any other change regarding the model setting.

Non-Bayesian inference methods have also been proposed to efficiently quantify uncertainty in Deep Learning models such as: Deep Ensemble [94] or Evidential Deep Learning [146]. Both approaches are used and described in Chapter 4.

In recent years, diverse and successful works have been published in computer vision that take uncertainty into account in their deep learning based proposal. For example, in the image classification case, uncertainty has been modeled into Large-Margin Softmax Loss [106] improving the classification in unbalanced datasets [84] and also modeled on input labels improving the robustness of the model against adverse attacks [129]. For image segmentation, MC-dropout Bayesian method was adopted to weights the predictions from a multi-view inputs generated from different transformation on a 3D data [171] and an uncertainty-aware metric and annotation were proposed to solve the night-time semantic segmentation through a curriculum model adaptation from the correspondence of day-night images content [144]. As for object detection, modeling bounding boxes with a Gaussian distribution was proposed in [30, 60] and an aleatory and epistemic uncertainty has been modeled through a Laplace-based [82] and variational dropout [86] loss function, respectively, to improve the 3D pedestrian localization[17]. In addition, they have been published for multi-modal learning [40, 156], for person re-identification [180], for video-based face recognition [191], for image processing [176, 188], to name a few more. Although they are all interesting applications, we have no evidence of uncertainty-based methods applied to food analysis before us.

Chapter 3

Regularized Uncertainty-based Multi-Task Learning Model for Food Analysis

3.1 Introduction

Being aware of our daily life, diet is an important issue if we want to follow balanced and healthy lifestyle. The amount of people that suffer from nutrition-related health conditions like obesity or cardiovascular diseases is increasing among our society [168]. Moreover, people that have any of these medical conditions, tend to spend more money on medical care and usually have a shorter lifespan. Thus, the need for increasing the awareness of the importance of following a healthy diet is vital.

Considering the importance of food in our daily life, together with the proliferation of social networks, new trends arose. A rather extended trend related to nutrition consists in taking pictures of food in restaurants or other special events of the daily life. Several users that follow this tendency, are used to share pictures on social networks like Instagram or Pinterest. Considering the huge amount of food-related pictures that are already available *online* due to these routinary actions, it is natural to think of using them for developing automatic methods for food recognition and analysis.

Given the importance of food and the challenging image analysis problem, the computer vision community proposed different food analysis methods like food detection [2], recognition [1, 111], localization [21], ingredients recognition [27], multi-attributes recognition [184], recipes retrieval [145], calorie counting [42], food-tray recognition [6], or portion estimation [39], among others. Some of the purposes of these methods could include: 1) easing the tracking of our daily nutrition intake, which would lead to offering recommendations for improvement; 2) providing alerts when recognizing a product

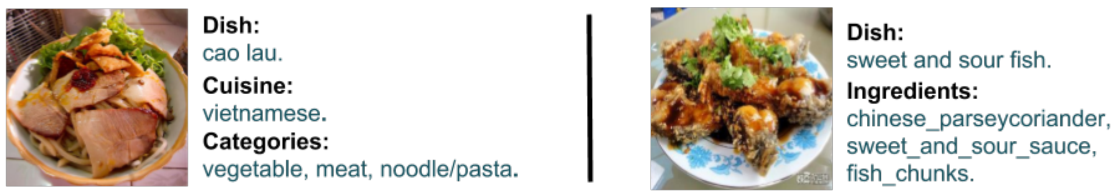


FIGURE 3.1: Food-related tasks predicted by our model.

that could contain potential allergens; 3) provide cooking recommendations; 4) automatic billing in self-service restaurants; just to mention a few.

Although the potential of food analysis systems is clear, several challenges need to be solved. In particular, if we compare food images to other visual analysis problems like object recognition, food classes have much higher inter-class similarity and intra-class variability, making any food analysis problem very difficult to solve. Considering the particular problems of food-related tasks, ingredients recognition can be of high difficulty considering that some ingredients can be present in several textures and shapes, and others can be invisible.

In this chapter, we argue that most tasks related to food analysis can be interconnected (e.g. swiss cuisine, fondue and cheese). To this aim, we explore the problem of food multi-task learning and propose a Regularized Uncertainty based Multi-Task Learning model in which different food-related characteristics that are correlated can be predicted from an input image: a) dish, b) cuisine, c) categories and d) ingredients (see Fig. 3.1). One of the problems of most multi-task methods is that they do not necessarily consider the correlations between tasks. Note that for multi-label classification problems, the relationship between labels have been studied extensively [100, 174, 105, 99]. Instead, as far as we know, there are no proposals for multi-task learning case, where the relationship of the labels between the different tasks being single-label or multi-label are used to regularize the predictions. Furthermore, we argue that different tasks should influence in different degree to the overall model according to their uncertainty. To this purpose, we propose in this chapter the following contributions:

1. We extend the model for Multi-Task Learning with Homoscedastic Uncertainty [83] to allow single-label and multi-label classification.

2. We propose a Multi-Task regularization term that weighs existent negative and positive correlations between classes belonging to different tasks.

3. We publish a new Multi-Attribute Food dataset (MAFood-121) with more than 20.000 images and multi-task annotations, including: dish, cuisine and food categories.

4. We propose a new metric called Multi-Task Accuracy (MTA), which measures the prediction agreement and coherence between tasks.

We prove that our model outperforms the state-of-the-art techniques on MAFood-121 and VIREO Food-172 not only using traditional evaluation metrics, but also on our newly proposed metric that ensures the coherence of the different tasks.

The chapter is organized as follows: in the following section, Sec. 3.2, we review the state-of-the-art related to the most relevant food analysis problems on the computer vision field. In Sec. 3.3, we present our proposal for multi-task learning, detailing both our model and the loss functions that we introduce for training.Then, in Sec. 3.4, we introduce the multi-attribute food datasets, the MTA metric and compare the results obtained by different methods. Finally, in Sec. 3.5, we present the concluding remarks.

3.2 Related Work

In recent years, there have been several studies focused on recognition of food categories.

Food Recognition is one of the most active topics on food-related problems. Most of the initial studies tackling this task proposed the use of hand-crafted features based on color, texture and shape [29, 73, 22, 112, 80]; with the exception of [175], that exploited relationships between ingredients, using statistics of pairwise local features to perform the recognition. All of them were evaluated mainly on datasets with too specific, or a limited number of images. Moreover, most of them were taken in restricted conditions [22, 175] or with up to 50 different dishes [22, 73, 80].

Lately, the emergence of Convolutional Neural Networks made possible to tackle the problem on more challenging food datasets with a large number of images and a wide diversity of dishes [23, 81]. A significant increase of the performance was shown compared to hand-crafted features [23, 173]. Most of the results on CNNs have been obtained using fine-tuned models based on winners of the ImageNet challenges [141], being [58, 104, 173] notable examples that improved the results by a great margin.

Instead of just fine-tuning, a different approach is proposed by [1], which fused several CNN models and explored different fuzzy similarity measures, evidencing better results than using a single network. Finally, Martinel et al. [111] proposed a wide-slide residual network, which provided the best performance in two benchmark datasets, 90.27% on Food-101, and 83.15% on UECFood-256.

Food Group Recognition can serve as a way to represent a group of dishes instead of a particular dish. Food recognition is a complex problem due to its fine-grained nature, which has the particularity of being a problem with a high intra-class variability and high inter-class similarity. In [46], a food dataset with 8 categories related to restaurant menus was proposed. In their dataset, images depicting mixed food (e.g. second course and side dish) were labeled with multiple labels. Instead, the work in [153, 170] adopted a semantic categorization of food, in [170] the authors proposed a three-leveled hierarchy of food in order to make better mistakes on the food recognition problem. The authors of [153] proposed a new dataset with 11 categories considering the major food groups defined by the United States Department of Agriculture. Regarding the results on Food-11, Aguilar et at. [1] achieved the best performance by fusing three classifiers based on CNNs.

Ingredients Recognition can be a possible solution to the high diversity of dishes existent in the food recognition problem. Right now, there are more than 8,000 dishes (according to Wikipedia) [20]. On the other hand, certain ingredients are not visible or are indistinguishable in the image. Taking into account all these challenges, few works considered the ingredients recognition problem so far. Chen et al. in [27] proposed a model that simultaneously recognizes the dish and the visible ingredients. Bolaños et al. [20] tackled the ingredients recognition problem by considering any present ingredient either visible or not. In addition, they opted for a model with a single output to avoid the loss of generalization capabilities on unseen recipes/dishes.

Cuisine Recognition can be characterized as capturing the set of ingredients, cooking methods and presentation of a certain region in the world. Sajadmanesh et al. [143] showed that, due to the geographical locality of the ingredients, some of them are representative to a specific cuisine, and thus, they play a main role to classify the cuisine [118, 143, 155, 185]. From the computer vision point of view, an early model is proposed by [185], which is based on high-level features detecting 16 ingredients characteristic of certain cuisines. A more recent work [118] took into account that the information of the ingredients, in some cases is not sufficient to classify the cuisine. Therefore, it proposed a multi-modal framework which simultaneously modeled the visual content and textual ingredients to tackle the problem.

Taking into account that on an image we can perform different recognition tasks, our main goal is to propose a highly performing model that boosts prediction using the relationship between different food recognition tasks.

3.2.1 Multi-Task Learning based on CNNs

Recently, MTL approaches using CNNs have been adopted by several works to jointly learn related tasks. Several works when applied on face or pedestrian appearance analysis tasks (e.g. facial landmark detection, human pose estimation, etc.), have shown a significant improvement in performance [98, 108, 118, 178, 179, 187, 52]. A novel architecture that utilizes the hierarchical features from different tasks was proposed in [52]. The authors suggest that features of different CNN levels from multiple tasks are not

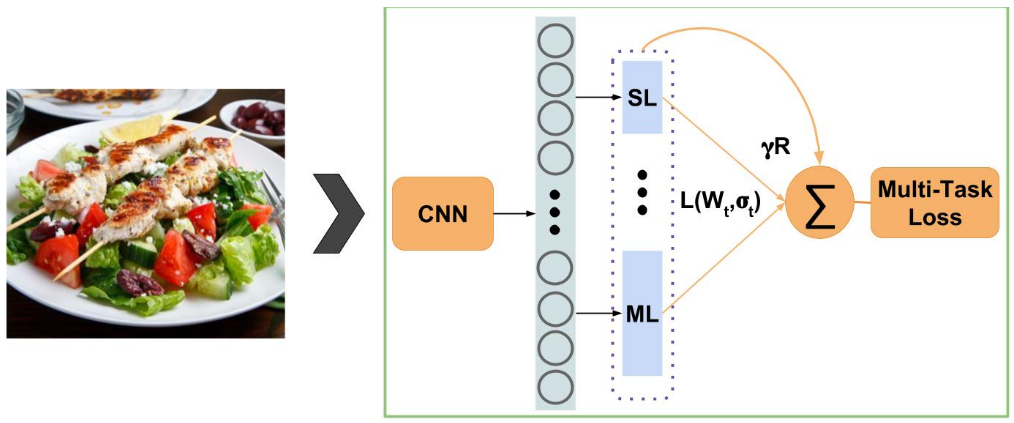


FIGURE 3.2: Regularized Uncertainty-based Multi-Task Learning model for classification of food-related tasks.

identical when the tasks are loosely related. They tackle this problem by concatenating the features of different tasks at different CNN levels. Later, they apply a discriminative dimensionality reduction in order to learn a discriminative feature embedding that satisfies the channel size of the following CNN layers. Instead, in our case we deal with highly related tasks. Additionally, we force the joint learning of the features from the different tasks within the loss function by weighing tasks according to their uncertainty and applying a class regularization term. The good results evidenced in the literature has inspired the emergence of new food analysis works that deal with the simultaneous prediction of two or more tasks. Zhang et al. [184] proposed a MTL based on AlexNet CNN, to recognize three tasks: food, cooking method and ingredients. Chen et al. [27] recognized food and ingredients simultaneously, and the authors in [42] applied simultaneous food recognition and calories estimation. An interesting aspect to highlight is that with the exception of [179], previous works weighted uniformly the loss of each individual task [98, 178] or manually tuned them [27, 42, 179, 184]. In contrast, Yin et al. [179] proposed a dynamic-weighting scheme that fixed the main task with a weight equal to 1, and learned the weights for each side-task. Although not in the food analysis field, a completely dynamic-weighting is proposed by [83], where a multi-task loss function was derived based on maximizing the Gaussian likelihood with homoscedastic uncertainty for both regression and classification tasks.

3.3 Regularized Uncertainty-based Multi-Task Learning model for food analysis

The Regularized Uncertainty-based Multi-Task Learning model (RUMTL) that we propose addresses the problem of multi-attribute food prediction. Our goal is to generate coherent and simultaneous recognition of different food-related attributes that are correlated: 1) dish, 2) cuisine, 3) food categories and 4) ingredients. We propose tackling the problem in a MTL deep learning framework.

3.3.1 Model Design and Activation Functions

We adopted the ResNet-50 [59] as a base for our RUMTL architecture due to the good performance demonstrated in computer vision problems related to object detection and, in particular, for food-related problems [1, 111, 145]. For our purpose, we modified the original design of ResNet-50 by removing the last Fully Connected layer (FC), and instead connected as many FC layers, as tasks we have, with a FC shared layer located at the top of the network containing 2048 neurons (see model architecture in Fig. 3.2).

Taking into consideration that we have two different types of outputs: a) single-label for dish and cuisine; and b) multi-label for categories and ingredients, we need to use different activation functions in the outputs layers.

For the Single-Label tasks, we apply the softmax activation function, due to its ability to obtain a probability distribution that enhances the single most probable class, defined as:

$$softmax(f^{W^t}(x)_i) = \frac{\exp(f^{W^t}(x)_i)}{\sum_j \exp(f^{W^t}(x)_j)},$$

where $f^{W^t}(x)_i$ is the final activation of the network for the sample x on the i-th label and the t-th task, $t = 1, 2$; $i = 1, \ldots, |C|^t$; where $|C|^t$ is the number of classes in task t, which has a set of weights W^t. On the other hand, for the Multi-Label tasks, we apply the sigmoid activation function:

$$sigmoid(f^{W^t}(x)_i) = \frac{1}{1 + \exp(-f^{W^t}(x)_i)},$$

which provides an independent probability for each class and enables the prediction of multiple classes [20]. From these activations, we can determine the probability that the output y^t corresponds to the target label(s) $\hat{y^t}$ for the task t given the image x, as follows:

1. SL probability:
$$p(y^t = \hat{y^t}|x) = softmax(f^{W^t}(x)_{\hat{y^t}})$$

2. ML probability:

$$
\begin{aligned}
p(y^t = \hat{y}^t|x) &= \prod_{i}^{|C|^t} p(y^t_i = \hat{y^t}_i|x) \\
&= \prod_{i}^{|C|^t} sigmoid(f^{W^t}(x)_i)^{\hat{y^t}_i} \times (1 - sigmoid(f^{W^t}(x)_i))^{1-\hat{y^t}_i}
\end{aligned}
$$

3.3.2 Multi-task Uncertainty-based Loss

Regarding the loss functions for the model optimization, we apply the categorical cross-entropy for the dish and cuisine tasks, and binary cross-entropy loss for the food categories and ingredients. Note that, for SL and ML tasks the loss function is calculated as: $-\log p(y^t = \hat{y}^t|x)$.

One of the important issues in MTL concerns on how to combine multiple objectives and train them jointly in order to capture the existent relationships between tasks. Using a manually defined weighing procedure is expensive to tune, increasingly difficult and makes the training process both very sensitive to the parameters and also computationally inefficient. To cope with these problems, the authors in [83] proposed to use a weighing procedure for learning and combining different tasks based on probabilistic uncertainty modeling. In Bayesian modeling, there are two main types of uncertainty [82]: epistemic uncertainty, related to the lack of training data, and aleatoric uncertainty related to the missing information that our data cannot explain. The later can be divided into 2 subcategories: heteroscedastic uncertainty (data-dependent) that is predicted as a model output and homoscedastic (task-dependant) that varies between different tasks.

The authors in [83] proposed a MTL loss function maximizing the Gaussian likelihood with homoscedastic uncertainty. Let $f^W(x)$ be the output of a neural network with weights W and input x. In the case of MTL, the likelihood is expressed as: $p(y_1, y_2, \ldots, y_T)$ considering $f^W(x)$ as a sufficient statistics. Assuming a Gaussian distribution, the loss expressed as a negative log likelihood $L(W, \sigma)$ for each SL classification probabilistic output, modeling the homeoscedastic uncertainty σ, can be approximated by [82]:

$$L(W, \sigma) \approx \frac{1}{\sigma^2} L(W) + \log \sigma^2,$$

using the softmax function as an activation function. The authors defined the MTL loss function integrating the uncertainty along tasks as a sum of the individual multi-task losses: $L(W, \sigma_1, ..., \sigma_T) = \sum_{t=1}^{T} L(W, \sigma_t)$.

In contrast to [83] in our case, we have SL as well as ML classification tasks. A straightforward alternative to solve the ML classification is to transform the N possible outputs of the multi-label classification to N independent single-label binary classifiers. With this transformation every output value could become an independent classification task with two labels - relevant or irrelevant (binary relevance method [186]). Then, the loss function with uncertainty based on the categorical cross-entropy might be used for the loss of each binary classification, treating them as independent single-label tasks. However, the binary relevance method assumes independence between outputs, which is not true in our case. Thus, their relationship is not taken into account during the learning. Instead, to avoid suboptimal results due to the ignorance of label correlations, we propose a loss function with uncertainty for the binary cross-entropy. To this purpose, we derive the uncertainty ML classification loss as follows:

$$\begin{aligned} L(W,\sigma) &= -\log p(y^t = \hat{y^t}|x,\sigma) \\ &= -\log \prod_i p(y_i^t = \hat{y_i^t}|x,\sigma) \\ &= -\sum_i \log(sigmoid(f^{W^t}(\frac{x}{\sigma^2})_i)^{\hat{y_i^t}}(1 - sigmoid(f^{W^t}(\frac{x}{\sigma^2})_i))^{1-\hat{y_i^t}}) \\ &= \frac{1}{\sigma^2}L(W) + \sum_i \log(\frac{\exp^{\frac{1}{\sigma^2}f^W(x)_i} + 1}{(\exp^{f^W(x)_i} + 1)^{\frac{1}{\sigma^2}}}) \\ &\approx \frac{1}{\sigma^2}L(W) + K\log\sigma^2 \end{aligned}$$

where L(W) is the sigmoid binary cross-entropy, and K corresponds to the number of labels. Note that in order to simplify the objective function, similar to [83], in the last transition we introduce the assumption that $\frac{1}{\sigma^2}(\exp^{\frac{f^W(x)_i}{\sigma^2}} + 1) \approx (\exp^{f^W(x)_i} + 1)^{\frac{1}{\sigma^2}}$, becomes an equality when $\sigma^2 \rightarrow 1$.

Therefore, we extend the multi-task objective with homoscedastic task uncertainty for SL and ML classification task as:

$$L(W,\sigma_1,\ldots,\sigma_i) = \sum_i \frac{1}{\sigma_i^2}L(W) + K\log\sigma_i^2 \tag{3.1}$$

where for SL classification task, K takes a value of 1 and $L(W)$ corresponds to the categorical cross entropy, which coincides with the SL loss expression in Kendall's work [83].

The advantage of this model is that it learns the relative weights, σ_t in a well-founded way. The loss is smoothly differentiable and prevents the task weights σ_t from converging to 0. The parameters σ_t model the observational noise, that is, they capture how much noise is manifested in the output. During the training process, the log likelihood is maximized with respect to the model parameters W and the observation noise parameters σ_t, $t = 1, ...T$.

3.3.3 Multi-task Class Regularization

In our case of food analysis, it is natural that some of the classes from different tasks can be correlated negatively (for example, *fondue* is not typical for Japanese cuisine).

To this purpose, we create a **task-exclusion matrix**, $T^{kl} = \{t^{kl}_{ij}\}$, $i = 1, ...|C|^k$ and $j = 1, ...|C|^l$ where:

$$t^{kl}_{ij} = \begin{cases} \text{-1}, & \text{if class } i \text{ and class } j \text{ are exclusive,} \\ 1, & \text{otherwise.} \end{cases}$$

To determine the class exclusions, we explored the ground-truth of the training data imposing the condition that if there are no examples of images from class i in task k and class j in task l, then it will be an exclusive relationship.

In our RUMTL model, we penalize when the multi-task classification infers classes of different tasks that are mutually exclusive (e.g. the dish *goulash* and the cuisine *Japanese*). Thus, we define the following penalizing term:

$$R^{k,l}_{-} = \frac{1}{s^{kl}} \sum_{i=1}^{|C|^k} \sum_{j=1}^{|C|^l} \max(0, -t^{kl}_{ij}) \cdot p(y^k_i|x) \cdot p(y^l_j|x)$$

where s^{kl} is a normalization factor defined as follows:

$$s^{kl} = \begin{cases} 1, & k \text{ and } l \text{ are SL,} \\ \sum_{j=1}^{|C|^l} \max(0, -\min_i(t^{kl}_{ij})), & k \text{ is SL and } l \text{ is ML,} \\ \sum_{i=1}^{|C|^k} \max(0, -\min_j(t^{kl}_{ij})), & l \text{ is SL and } k \text{ is ML,} \\ \sum_{i=1}^{|C|^k} \sum_{j=1}^{|C|^l} \max(0, -t^{kl}_{ij}), & k \text{ and } l \text{ are ML.} \end{cases}$$

On the other hand, when the classes are not exclusive (e.g. the category *pasta* and the cuisine *italian*), the optimization process should push the likelihood of both classes to be as high as possible. Note that this case covers as positive correlation as no correlation between classes. To this purpose, we introduce a second term defined as follows:

$$R^{kl}_{+} = \frac{\sum_{i=1}^{|C|^k} \sum_{j=1}^{|C|^l} \hat{y}^k_i \hat{y}^l_j (1 - p(y^k_i|x) \cdot p(y^l_j|x))}{\sum_{i=1}^{|C|^k} \sum_{j=1}^{|C|^l} \hat{y}^k_i \hat{y}^l_j}$$

where:

$$\hat{y}^t_c = \begin{cases} 1, & \text{if } x \text{ belongs to class } c \text{ from task } t, \\ 0, & \text{otherwise.} \end{cases}$$

Finally, we define the regularization term as a weighed sum of R_- and R_+:

$$R = \binom{T}{2}^{-1} \sum_{k=1}^{T-1} \sum_{l=k+1}^{T} (\alpha R_-^{kl} + \beta R_+^{kl})$$

where α and β look for a trade-off between both terms. In our case, without loss of generality, we used $\alpha = 0.5$ and $\beta = 0.5$. Note that due to the regularization term, if two classes from different tasks are not likely to occur together (that is in some way they are exclusive and in this case, $t_{i,j}^{k,l} = -1$), but the classifiers obtain high probabilities ($p(y_i^k|x) \cdot p(y_j^l|x)$ is high), the regularization term will penalize the loss function. On the other hand, if for a given sample x, there is a clear relation between the classes and at the same time there is no exclusive relation between classes ($t_{i,j}^{k,l} = 1$), the regularization term will try to push the likelihood of the most probable classes close to 1. The final definition of the loss function becomes:

$$L(W, \sigma, \gamma) = \sum_{t=1}^{T} (\frac{1}{\sigma_t^2} L_t(W_t) + K \log \sigma_t^2) + \gamma R \tag{3.2}$$

where γ is a parameter to weigh the importance of the regularization term ($\gamma = 1$, in our case).

To sum up, the proposed regularization allows to guide the joint learning of the different tasks considering two factors: the negative (R-) and non-negative (R+) correlation between the classes of different tasks.

- For R-, first of all we must build the task-exclusion matrix from the training data. With this we can determine during the training, if two classes of different tasks are correlated or not. In case the prediction of the model obtains uncorrelated classes, the regularization factor will increase the value of the loss function. As it can be inferred, the model will tend to generate fewer incoherent predictions, but it does not assure us that the prediction really will be correct to the image. For this, we propose the second term of the R+ regularization.
- The term, R+ does not need to use the exclusivity matrix, but considers the ground truth of the image. The aim is to obtain the highest possible probability given by the model to the ground truth of the image for the different tasks, so it will be penalized as long as this probability for the ground truth labels is less than 1.
- When using both terms together we have found a balance between the improvement of the performance and the coherence of the output between the different tasks.

3.4 Experimental Results

In this section, we first present the datasets used, second we describe evaluation measures, third we present the experimental setup and then we describe the results obtained with the proposed RUMTL approach compared to using single-task models.

3.4.1 Multi-Attribute Food Datasets

In order to justify the results and prove the advantages of our newly proposed multi-task framework, we need a challenging datasets with multiple food-related attributes. In [114], different sets of data with annotations of different tasks were used. However, this is not suitable for building a multi-task model, which requires a single set of data with the annotations for all the tasks. The only publicly available multi-task food dataset is VIREO Food-172 [27], which has annotations for two tasks (dish and ingredients). To highlight the benefits of our approach considering at the same time more realistic multi-task problems, we propose a new dataset with three complementary tasks that have not been exploited so far, two single-label tasks (dish and cuisine) and one multi-label task (food group/categories).

MAFood-121.

We built and make public here a dataset of food images comprising 3 different tasks: a) dish, b) cuisine and c) categories. It consists of 21.175 images, distributed as 72.5% for training, 12.5% for validation and 15% for test. We named the resulting dataset as *MAFood-121.* Both dish and cuisine can take only one value per image, while categories have multi-label annotations. In order to choose the images to include, we selected the top 11 most popular cuisines in the world according to Google Trends (`www.google.com/trends`) (see Fig. 3.3). For each cuisine, 11 traditional dishes were chosen. In total, the dataset consists of 121 dishes, where each one belongs to at least one of the following 10 food categories: Bread, Egg, Fried food, Meat, Noodles/Pasta, Rice, Seafood, Soup, Dumpling and Vegetable. 8 of the 10 categories used coincide with those proposed by [153]. Regarding the food classes, the images were collected from 4 different sources, 3 of them were the public datasets Food-101[23], UEC-Food256[81] and Turk-15 [56]. To reduce the bias that could be present due to the different amount of images per dish, we selected a maximum of 250 images for each one. Regarding Food-101, 250 images were collected from the original test set. As for UEC-FOOD256 and Turk-15, the images were randomly chosen.

For our purpose, it was necessary to collect 38 new dishes along 5 different cuisines from Google Search Engine, because the existing public food datasets consist mainly of food images of a specific geographic region [23, 56, 81]. We restricted the search to the country

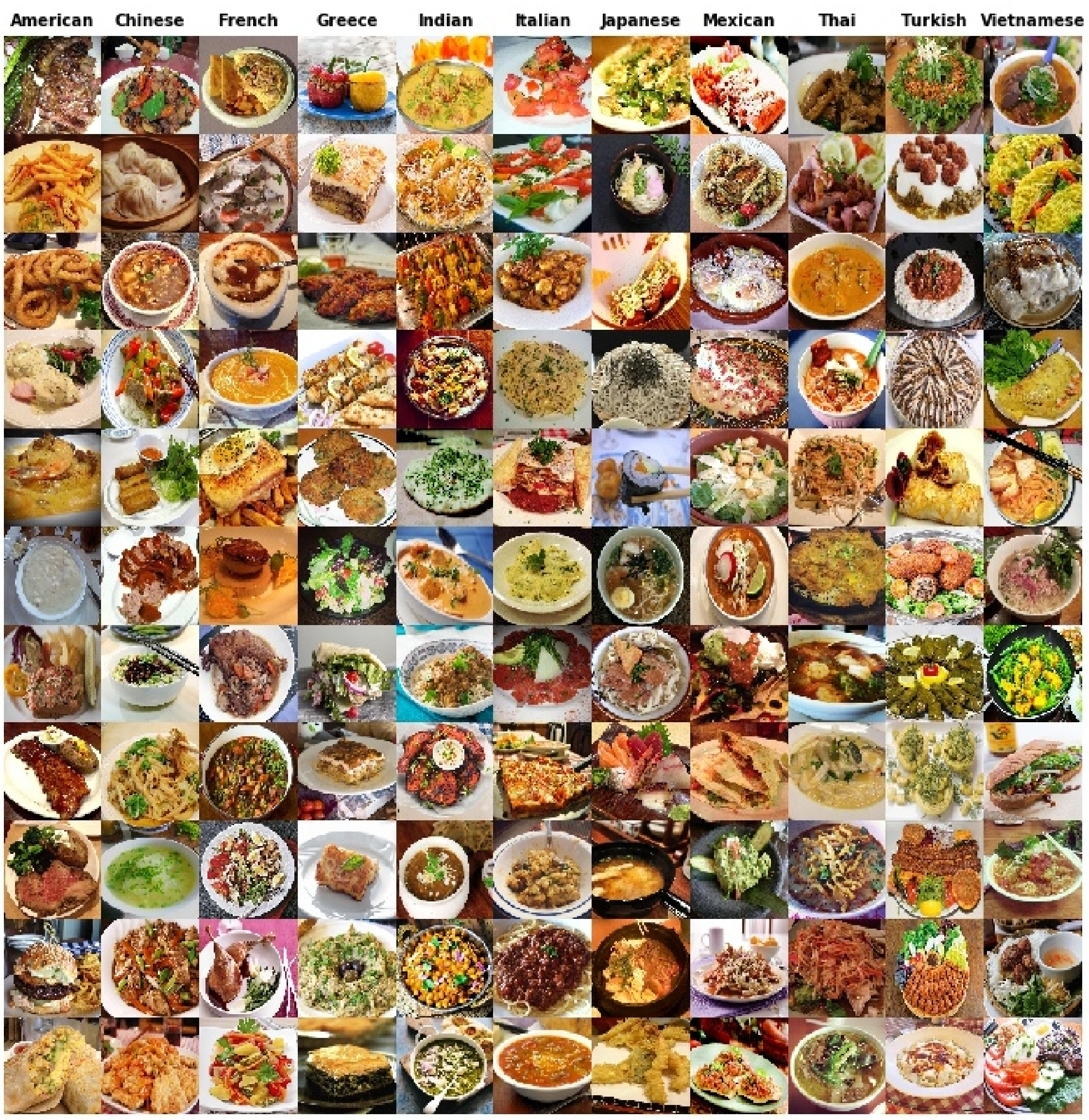


FIGURE 3.3: Examples of images from the MAFood-121 dataset divided in the 11 different cuisines considered.

that represents a specific cuisine and also restricted their minimum sizes to 256 pixels per side. For each dish, 200 images were downloaded and, following the procedure in [23], all images were rescaled to have a maximum size of 512 pixels. The images acquired were reviewed to remove those with explicit copyright, duplicates, and also those that are not representative of the respective dish. To achieve this, we first applied the automatic food/non-food classifier proposed by [2], and then we manually inspected the remaining images. Consequently, an average of 119 images were obtained per dish. After gathering the images from 11 cuisines and their respective 121 different dishes, we manually labeled the food categories for each image considering the visible ingredients.

VIREO Food-172.

This public multi-task dataset consists of 172 popular chinese dishes collected by Baidu and Google image search. VIREO Food-172 has annotations for two tasks: dishes and ingredients. For the last one, the authors only consider the annotation of visible ingredients among 353 ingredients labels, with an average of 3 ingredients per image. In total, the dataset contains $110,241$ images corresponding to 172 dishes and 353 ingredient labels, distributed as 60% for training, 10% for validation and the remaining 30% for testing.

3.4.2 Metrics

In order to evaluate the performance for each individual task, we chose four standard measures frequently used: *Overall Accuracy*, for SL; and *Recall*, *Precision* and *F_1-score (F_1)*, for ML. However, these measures do not reflect the quantity of samples asserted by all the tasks at the same time. To this purpose, we propose a new and more *strict* measure, we name *Multi-Task Accuracy (MTA)*, which represents the ratio of images correctly classified by all tasks at the same time (i.e. measures the prediction agreement and coherence between tasks). We formally define the MTA measure as follows:

$$MTA = \frac{1}{N}\sum_{i=1}^{N}\prod_{t=1}^{T}\frac{|y_{i,t}^{true} \cap y_{i,t}^{pred}|}{|y_{i,t}^{true} \cup y_{i,t}^{pred}|},$$

where $|.|$ stands for the cardinality for both label sets ground-truth (y^{true}) and predictions (y^{pred}), N and T denote the number of images and tasks, respectively. Note that the denominator specially has sense for the ML data, since it allows to obtain 1 for a special i and j, iff all ML predicted and ground-truth labels coincide.

Note that MTA allows us to evaluate the performance of Multi-task classifiers as a whole. If we have different classifiers with the same performance with respect to the individual tasks, it does not ensure that they provide the same results because the errors of each task can come from different images. We believe that it is of interest to have classification that concentrates the tasks errors in the same images in order to get consistent outputs. In this sense, MTA can help to identify better classifiers even when $F1$ and accuracy are similar for each task. We illustrate it with the following example: suppose we have classifiers C_1 and C_2, and images I_1 and I_2, both with the same accuracy for tasks T_1 and T_2, but with different classification results. $C1$ mis-classifies both tasks for $I1$, and C_2 mis-classifies I_1 for T_1 and I_2 for T_2. Thus, C_1 obtains 50% and C_2 a 0%, with respect to the MTA metric. Therefore, despite having the same accuracy, the classifier C_1 produces more coherent results among tasks. Let us consider another example: C_1 (accuracies 0.8 and 0.8) and C_2 (accuracies 1.0 and 0.6). Let us assume that both tasks are single-label and focus on which data/images the error is produced. In this case, $MTA(C_1)$ will be equivalent to

$MTA(C_2)$ only when the 0.2% Acc. error of each task occurs in different images. In the rest of the cases, $MTA(C_1) > MTA(C_2)$, while $MeanAcc(C_1) = MeanAcc(C_2)$. In other words, if we consider that 100 images were predicted by both algorithms, C2 will have inconsistencies in 40 of them, and in the best case C1 will produce only 20 completely misclassified images and all the rest will be correctly classified. Therefore, if we did not consider the data (images) and only looked at the global results, with C1 we would have less misclassified data when considering the coherence between the tasks predictions. Note that the MTA is a lower bound metric of the classical definition of the accuracy of multi-task learning (Mean Acc) i.e. improving MTA leads to improve the accuracy too. In other words, the maximum MTA value achievable is equal to the Mean Acc. When classifications for all tasks are correct for most of the samples: MTA is close to Mean Acc. However in the case that at least one of the tasks is misclassified for most of the samples: MTA tends to 0.

Furthermore, considering the human factor when analyzing the results obtained by a multi-task classifier, MTA allows us to obtain a higher acceptance rate by the end users. To illustrate it, imagine a scenario where we should show the performance of our multi-task model to an end user having two tasks (T1 and T2). On the one hand, if our classifier is wrong on at least one task (MTA=0.0, Mean Acc=0.5), the user usually perceives the outputs as wrong. For instance, a sample with GT dish (T1) = "spaghetti carbonara" and food group (T2) = "pasta" is correctly predicted as dish (T1): "spaghetti carbonara" but misclassified as food group (T2): "meat"; it would be difficult that the end-user perceives the model as well-performing. On the other hand, if our classifier obtains coherent predictions among tasks, both T1 and T2 will be either both correct or both incorrect (MTA=0.5, Mean Acc=0.5). Thus, the user will perceive as correct at least the predictions that provide task coherence, providing a better user acceptance rate of the model.

Summarizing, MTA is a stricter measure of performance that opens room for new research distinguishing data where all classifiers have difficulties to learn from data where some of the classifiers achieve to learn, that could be an indicator for the quality of the data. Hence, providing more specific metrics with respect to the performance of tasks and classifiers (like MTA) will allow to study in the future the different kinds of errors present (epistemic, homoscedastic or heteroscedastic) and thus, ease the improvement of the model.

3.4.3 Experimental setup

We trained end-to-end a CNN architecture (see Fig. 3.3) using the proposed multi-loss function in Eq. (3.2) optimized with Adam. Note that the number and type of output layers of our model depend on the attributes considered for each dataset (MAFood-121, Vireo Food-172). In total, four multi-task configurations applying different σ and γ parameters were evaluated. Additionally, single models focused on a specific task were trained for comparative purpose. The models are named as follows:

1. *Single-task models*: These models trained independently for each task are considered as a baseline.
2. *MTL ($\sigma_i = 1$, $\gamma = 0$)*: "Base" MTL model with the weights of tasks losses uniformly distributed.
3. *RMTL ($\sigma_i = 1$, $\gamma = 1$)*: "Base" MTL model with the weights of tasks losses uniformly distributed and the proposed regularizing term.
4. *UMTL ($\sigma_i = d$, $\gamma = 0$)*: MTL model with uncertainty-modeling weights.
5. *RUMTL ($\sigma_i = d$, $\gamma = 1$)*: MTL model with uncertainty-modeling weights and the proposed regularizing term.

As for training, first all models were pre-trained on the ILSVRC dataset. Then, we retrained the models during 100 epochs with a batch size of 20, and a learning rate of $2e-4$. In addition, we applied a decay of 0.2 every 8 epochs. The training was done using Keras with Theano as backend.

Regarding the weighting of losses, in order to smooth out the difference in the magnitudes of the losses between the SL and ML tasks, we assign an initial weight of 1 for SL tasks and 0.1 for ML task in MAFood-121 and 1 for SL task and 0.02 for ML task in VIREO Food-172. Specifically for the models *UMTL* and *RUMTL*, the loss weights are updated dynamically considering the uncertainty obtained by minimizing the target function $L(W, \sigma_t, \gamma)$ in each epoch using Adam optimization . Note that the σ_t were initialized to 1.

3.4.4 Results

In order to evaluate the performance of our approach in MaFood-121 dataset, we trained four variants of our model by changing the σ and γ values. In addition, we trained a set of single-task models for each task as baseline models. When σ takes the value 1, it implies a naive weighted sum of losses. On the other hand, when $\sigma = d$, it implies the use of uncertainty weighted of the losses. As for γ, it can take the values 1 or 0, whether we apply the proposed regularizing term or not. The results obtained for all the models for each task are shown in Table 3.1. Different measures were used depending on the nature of the tasks. In the case of ML, we show the results in terms of *Pre*, *Rec* and F_1 score. However, these measures mean the same in the case of SL multi-class approaches [128]. Therefore instead, we show the results with overall accuracy measure. In addition, we show the coherence of the results obtained among the individual tasks based on the

TABLE 3.1: Results of the different variants of our model compared to the use of single-task models on MAFood-121 dataset.

	Dish	**Cuisine**	**Categories**			
	Acc	*Acc*	F_1	*Pre*	*Rec*	*MTA*
Single-task	82.50%	85.71%	82.98%	84.15%	81.83%	62.45%
MTL	83.73%	88.23%	83.74%	86.38%	81.26%	67.05%
$R_{-}MTL$	83.76%	88.79%	83.81%	87.29%	80.59%	67.22%
$R_{+}MTL$	82.85%	88.01%	84.09%	86.16%	82.11%	67.24%
RMTL	83.35%	88.86%	84.27%	86.57%	82.09%	67.65%
UMTL	83.47%	88.92%	84.55%	86.61%	82.58%	68.22%
RUMTL	83.82%	88.35%	85.02%	86.40%	83.69%	**68.85%**

TABLE 3.2: Results of our proposed model compared to the use of single-task models on VIREO-172 dataset.

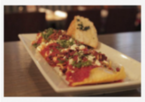
	Dish	**Ingredients**			
	Acc	F_1	*Prec*	Rec	*MTA*
VGG [27]	80.41%	60.81%	-	-	-
ResNet-50	84.67%	76.62%	81.35%	72.40%	62.96%
Arch-D [27]	82.06%	67.17%	-	-	-
MTL	84.34%	70.79%	79.28%	63.94%	56.69%
RUMTL	85.19%	76.87%	81.30%	72.89%	**64.99%**

MTA measure. The latter is applied on the multi-task models and also on all single-task models considering them as a unified model. In general terms, the variants of multi-task models were able to improve the performance on all tasks. Furthermore, according to the MTA metric, the application of the regularization term proposed together with the uncertainty weights allows us to jointly guide the learning of the tasks, with which we obtained better and more coherent results, achieving a total improvement of around 7% with respect to the single task models. Note that, the influence of the both terms of our proposed regularization can be easily seen from $R_{-}MTL$ and $R_{+}MTL$ when we compare the results obtained with respect to MTL in the food categories task for precision and recall metrics. In the case of $R_{-}MTL$, the precision increases, but the recall decreases, on the contrary, for $R_{+}MTL$ the recall increases, but the precision decreases. Integrating both terms ($RMTL$ model) we are able to balance their contributions achieving an improvement with respect to the MTL of 0.6% considering the MTA score. The benefit of the regularization also is evidenced when the uncertainty is included ($UMTL$ vs $RUMTL$), in this case the performance is improved by 0.63%.

GT	RUMTL	Single-task
Dish: tacos	**Dish**: tacos	**Dish**: prime_rib
Cuisine: mexican	**Cuisine**: mexican	**Cuisine**: american
Categories: vegetable, meat, bread	**Categories**: vegetable, bread	**Categories**: vegetable, meat

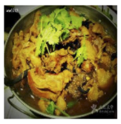
GT	RUMTL	Single-task
Dish: eggs_benedict	**Dish**: eggs_benedict	**Dish**: ravioli
Cuisine: american	**Cuisine**: american	**Cuisine**: italian
Categories: vegetable, bread, egg	**Categories**: vegetable, bread, egg	**Categories**: vegetable, egg

GT	RUMTL	Single-task
Dish: sushi	**Dish**: sushi	**Dish**: cha_ca
Cuisine: japanese	**Cuisine**: japanese	**Cuisine**: japanese
Categories: vegetable, seafood, rice	**Categories**: seafood, rice	**Categories**: fried_food



GT	RUMTL	Single-task
Dish: ravioli	**Dish**: bruschetta	**Dish**: lobster_roll_sandwich
Cuisine: italian	**Cuisine**: italian	**Cuisine**: italian
Categories: dumpling	**Categories**: vegetable, bread	**Categories**: vegetable, meat, bread

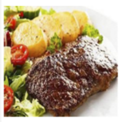
FIGURE 3.4: Success (top 3) and failure (bottom) cases of RUMTL on MAFood-121.



GT	RUMTL	Single-task
Dish: stewed chicken with mushroom	**Dish**: stewed chicken with mushroom	**Dish**: beef curry
Ingredients: chinese_parsleycoriander, chicken_chunks, dried_mushroom	**Ingredients**: chinese_parsleycoriander, chicken_chunks	**Ingredients**: chinese_parsleycoriander, chicken_chunks, dried_mushroom

GT	RUMTL	Single-task
Dish: beef noodles	**Dish**: beef noodles	**Dish**: beef noodles
Ingredients: chinese_parsleycoriander, beef_chunks, water, noodles	**Ingredients**: chinese_parsleycoriander, beef_chunks, water, noodles	**Ingredients**: minced_green_onion, chinese_parsleycoriander, water, noodles, beef_slices

GT	RUMTL	Single-task
Dish: deep fried lotus root	**Dish**: deep fried lotus root	**Dish**: deep fried chicken wings
Ingredients: fried_flour, lotus_root_box	**Ingredients**: fried_flour, lotus_root_box	**Ingredients**: fried_flour, lotus_root_box



GT	RUMTL	Single-task
Dish: beefsteak	**Dish**: beefsteak	**Dish**: beefsteak
Ingredients: hob_blocks_of_potato, lettuce, steak, cherry_tomato_slices	**Ingredients**: hob_blocks_of_potato, lettuce, broccoli, steak, tomato_sclices	**Ingredients**: lettuce, steak, cherry_tomato_slices, batonnet_potato

FIGURE 3.5: Success (top 3) and failure (bottom) cases of RUMTL on Vireo-172.

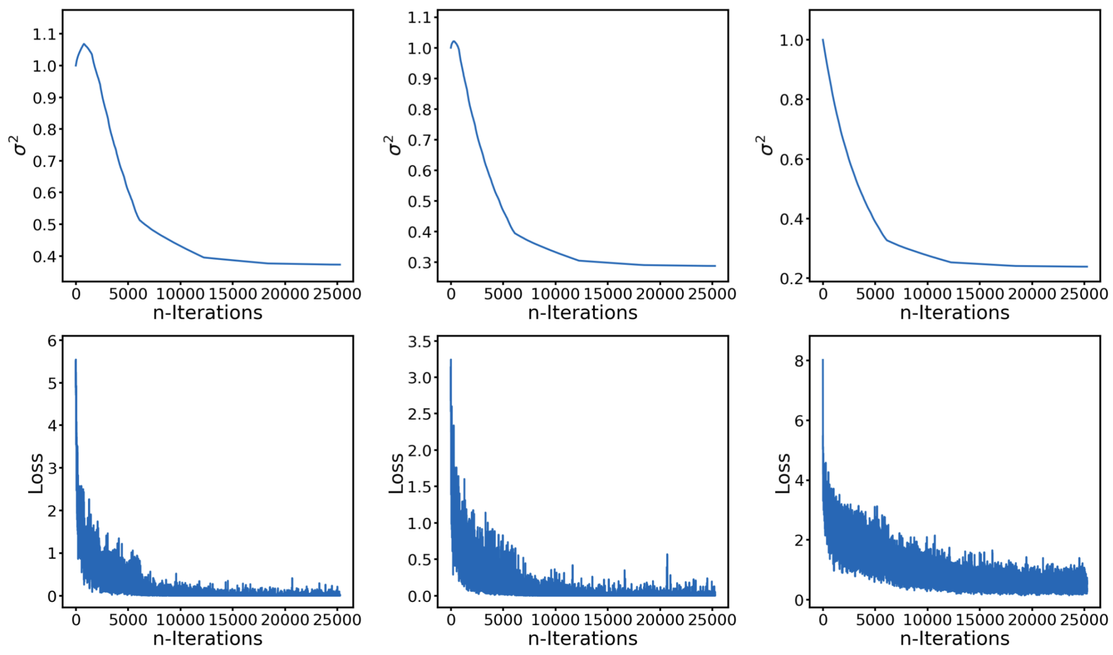


FIGURE 3.6: From left to right: Loss values and task uncertainty (σ^2) for the dish, cuisine and food families tasks obtained during the training of the RUMTL model on MAFood-121.

The results obtained with our proposed model RUMTL in VIREO Food-172 dataset can be seen in Table 3.2. In the same way to the results obtained in MaFood-121, our proposed model shows a better performance when it is evaluated for each task independently. Furthermore, despite the slight difference in the performance of the individual tasks of our proposal compared to single models based on ResNet-50 for each tasks, when we evaluate the joint classification (MTA metric), the result is about 2% better for our model RUMTL. Keep in mind that the MTA metric allows us to know what the behavior of the models is considering the performance achieved for all tasks in each image. Therefore, despite of single tasks are slightly worse, it is possible to have a large difference in MTA when the errors for each task are present in different images. This claims that if we want a system to simultaneously predict more than one task at once, RUMTL provides much better results than using single-task models.

We illustrate the results obtained for our proposed model on MAFood-121 (see Fig. 3.4) and VIREO Food-172 (see Fig. 3.5) datasets. In both cases, the good performance achieved (top 3 dishes) is observed on very diverse dishes. Furthermore, when our method fails, the results of the different tasks maintain a coherence (bottom image on MaFood-121) or are wrong but provide coherent predictions considering the image at hand (bottom image on VIREO Food-172). Finally, in Fig. 3.6 and Fig. 3.7, we show the loss functions evolution and uncertainty obtained for each task during training of the RUMTL model on MAFood-121 and VIREO Food-172 datasets respectively.

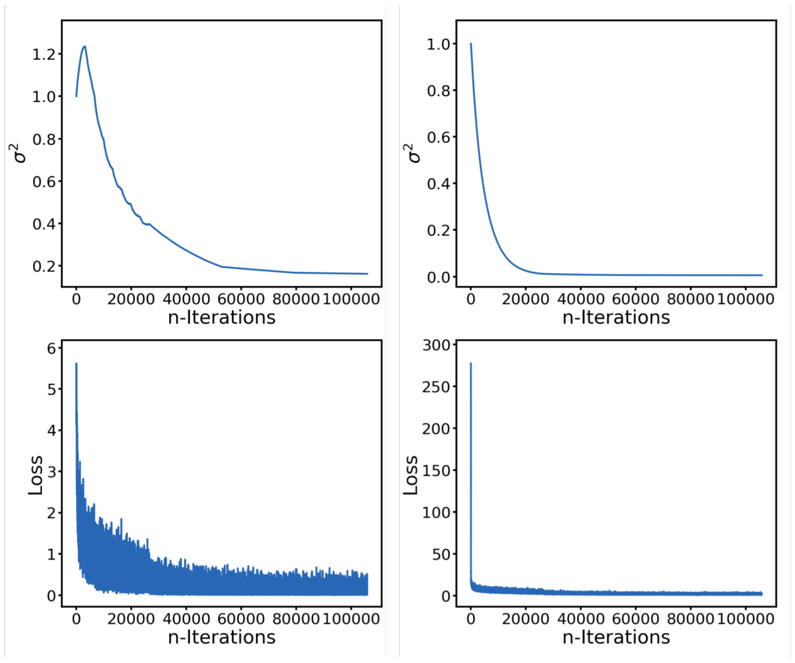


FIGURE 3.7: Loss values and task uncertainty (σ^2) for the dish (left) and ingredients (right) tasks obtained during the training of the RUMTL model on VIREO Food-172.

3.5 Conclusions

In this chapter, we proposed a new model for MTL that improves the coherence of the outputs compared to the single-task models by testing it on two food datasets. The improvement is achieved by modeling the uncertainty in the proposed MTL method, extending the proposal of Kendall [83] to allow SL and ML classification, as well as the incorporation of a MTL regularization term in the loss function. RUMTL allows an end-to-end training with automatic optimization of tasks weights. Furthermore, we publish a new dataset for multi-attribute food analysis and a new metric for measuring tasks coherence to serve as a basis for future works in MTL and food analysis fields.

Chapter 4

Uncertainty-aware hybrid classifier for food recognition

4.1 Introduction

Nowadays, there exists in the literature a large number of food image datasets to perform food recognition [81, 23, 56]. However, if we take into account the most common foods in all over the world, the amount provided by these datasets remains low compared to those needed for a real food recognition application, where thousands of different food classes are required. On the other hand, Convolutional Neural Networks have shown promising results in several Machine learning and Computer Vision tasks, and particularly in large-scale object recognition problems. Despite this, the model performance is affected by two factors [109]: 1) the number of classes that will be recognized and 2) the number of images that belong to each class. Some samples of the aforementioned classes can be seen in [181], where the proposed model decreases the performance in about 10% when compared the results of cifar-10 [90] with cifar-100 [90]. The same effect can be seen in the food domain, where a reduction of 7% is shown in [111] when compared the results on 100 food classes recognition in UECfood-100 [112] with respect to the recognition of 256 classes in UECfood-256 [81]. This suggests that single CNN models are not easily scalable, being not able to recognize a large number of food classes well enough.

In general, the strategies solving classification problems can be grouped in two ways [152]: 1) flat classification approach, where a single model is trained for all classes to be predicted (see the right part of Fig. 4.1), and 2) hierarchical classification approach, where the classes to be predicted are organized into a class hierarchy (see the left part of Fig. 4.1). As for the latter, there are three types of local classifiers to perform the predictions [152]: 2.1) Local Classifier Per Node Approach (LCN), where a binary classifier for each node of the class hierarchy must be trained; 2.2) Local Classifier Per Parent Node Approach (LCPN), where multi-class classifier for each parent node in the class hierarchy must be trained to distinguish between its child nodes; and 2.3) Local Classifier Per

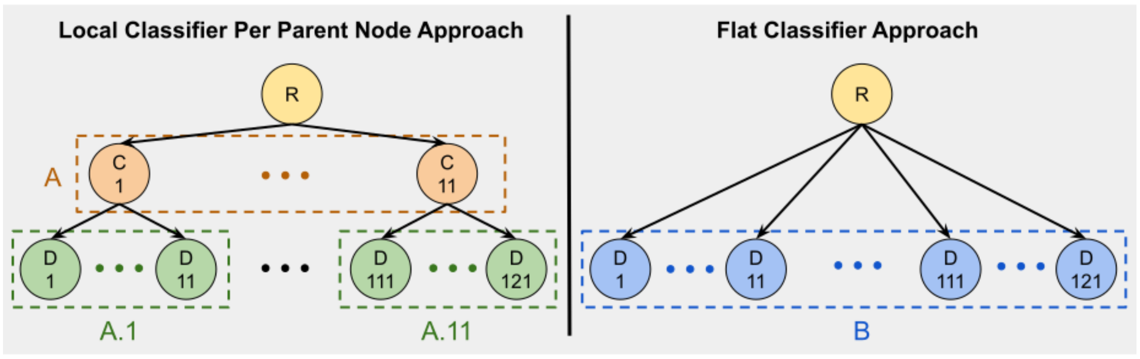


FIGURE 4.1: Illustration of a class hierarchy for a Local Classifier per Parent Node and Flat Classifier approaches on MAFood-121 dataset. A, A1-A11 and B denote the classifiers for the nodes shown within each rectangle.

Level Approach (LCL), where a multi-class classifier is trained for each level of the class hierarchy. In the food image domain, few approaches based on hierarchical classification have been proposed. One of them proposed a hierarchical approach incorporating an LCL strategy to ensure error closer to the real class when there is miss-classification [170]. The other one proposed hierarchically organizing several combinations of features (Color, HoG, LBP) with aims to find, for each class, the optimal combination for classification [132]. One of the main problems of hierarchical classification approaches is being prone to inconsistency (see Table 4.1). In the LCPN case, parent-child error propagation affect directly to the final prediction. To illustrate it, we can see the left part of the Fig.4.1. Let us consider A as the cuisine classifier, and A.1 to A.11 are local dish classifiers. Given an input image x, if classifier A miss-classifies the cuisine type of x like C11, the dish recognition (A.11) automatically will be wrong because the only labels provided by this classifier are from D111 to D121. Aiming to reduce the error propagated for the LCPN approach, we propose to classify with this strategy those predictions that are highly confident, and the remainder ones to classify with a flat classification approach. The likely good predictions may be identified by complementing the decision of the local classifier for the parent node with the most probable child node obtained from the analysis of Epistemic Uncertainty (EU). The EU captures the ignorance about which model generated the collected data [82], and therefore, the analyses of the EU can be a key clue to identify the correct local classifier for the child node. This is due to the fact that it is highly likely that we get a low uncertainty in the cases where the image to be predicted belongs to some class for which it was trained.

Our main contributions of this work are as follows: 1) we provide an epistemic uncertainty-based method, which integrates and takes advantage the benefits of two classification approaches: LCPN and flat; 2) we propose criteria to decide when to apply a local or flat classifier; and 3) we demonstrate the effectiveness of the proposed approach with three

TABLE 4.1: Advantages and disadvantages of traditional classification approaches and our proposal.

Characteristics	**Flat**	**LCN**	**LCPN**	**LCL**	**Our**
Ignore the class hierarchy	yes	no	no	yes	no
Number of classifiers	1	high	medium	low	medium
Prone to inconsistency	no	yes	yes	yes	LT LCPN
Uncertainty-aware method	no	no	no	no	yes

different methods of estimating EU, minimizing in all cases the propagation of parent-to-child errors produced in an LCPN approach, as well as improving classification performance with respect to the flat approach.

4.2 Related Work

Our proposal fits in the middle of two research fields, namely: 1) Food recognition and 2) Uncertainty estimation.

4.2.1 Food recognition

Food recognition is a challenging problem of object recognition that is characterized by its high intra-class variance and low inter-class variance among the food classes. To tackle this problem, deep learning features have proven to be much more effective than hand-crafted features [23, 173]. Consequently, researchers in recent years have addressed food recognition through classifiers based on CNNs, primarily by fine-tuning models pre-trained on ImageNet [104, 58, 1, 111, 89, 161]. Instead of just fine-tuning, more sophisticated approaches have been recently proposed in the best performing models of the state-of-the-art [111, 71, 189]. The WISeR architecture that combines two network branches was proposed in [111]. One branch is a Wide Residual Networks (WRNs) to extract the traditional food traits and the another one a custom slice network to capture the vertical structure of food images. On the other hand, in [71], a Multi-Scale Multi-View Feature Aggregation (MSMVFA) scheme was presented to capture semantic information from food images, composed by three different types of features: 1) high-level semantic features, from the output layer of a categorical network; 2) mid-level attribute features, from the output layer of ingredients network; and 3) deep visual features, from the layer before the output layer of a categorical network. Lastly, an efficient and accurate method has been proposed in [189], which consists of a compact network trained jointly, based on the knowledge distillation [62], to generate a high food recognition method applicable to mobile devices. Although the aforementioned methods have provided promising results on popular food benchmark datasets [23, 81, 27], these datasets are not representative enough to a real food recognition application, where we would need thousands of different foods that are eaten in all over the world.

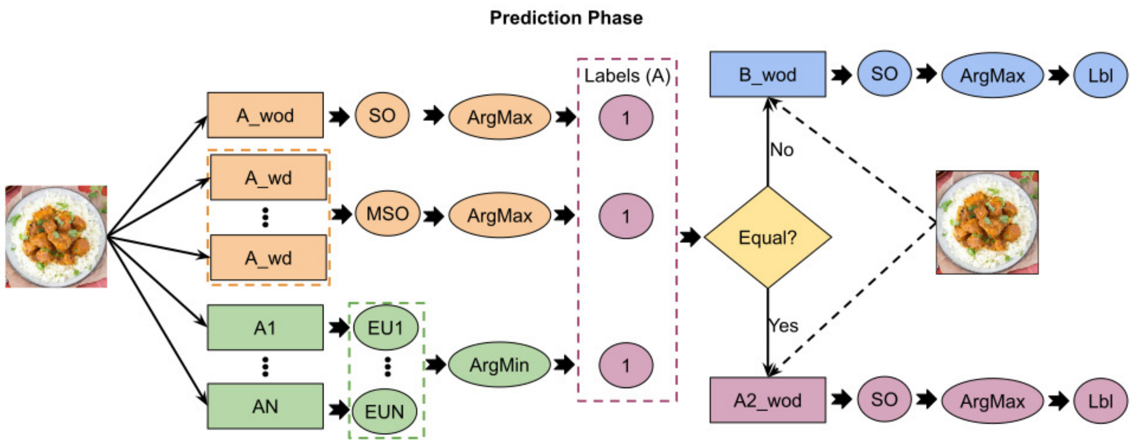


FIGURE 4.2: Main scheme of the proposed method, which shows the procedure performed when the predictions in the first level of the hierarchy are equal to 1. The suffix *wod* denotes the prediction with the dropout turned off, *wd* denotes the prediction with the dropout turned on, the terms *SO*, *MSO* and *EU*, denote Softmax Output, Mean Softmax Output and Epistemic Uncertainty.

4.2.2 Uncertainty estimation

In Bayesian modeling, the uncertainty can be divided in two types [82]: aleatoric (data-related) uncertainty, stemming from the missing information that our data cannot explain and epistemic (model-related) uncertainty, due to the lack of enough training data. The model uncertainty can be captured from a Bayesian Neural Network formulation. However, the exact Bayesian inference on the weights of a deep neural network is intractable due to the non-linear activations between consecutive layers [19, 50, 146]. Instead, approximation techniques such as variational Bayesian methods [19, 50, 121, 107] can be used. From these techniques, MC-dropout [50] is the most popular one due to its simple implementation and minimum adjustment required in the network design. An alternative of BNN is the method proposed by [94] that, motivated by the fact that dropout may be interpreted as an ensemble model combination [154], also proposed a simple method providing comparable or even better results than BNN, by applying a deep ensemble to uncertainty estimates. Evidential Deep Learning (EDL) [146] is another powerful method, based on the Theory of Evidence [37, 74], where a Dirichlet distribution on the class probabilities is employed to quantify belief masses and uncertainty in classification tasks.

4.3 Method for Uncertainty-based Hybrid Classification

In this section, we explain in detail the steps involved in the proposed approach, which considers local and flat classifiers to provide the final image prediction. In the following subsections, we first explain the pipeline of our proposed prediction framework, second we describe the quantification of EU by different considered methods and finally we comment on some considerations that must be kept in mind in CNN models.

4.3.1 Prediction by integrating local and flat classifiers

Let us consider the LCPN approach represented as a tree structure, where each non-leaf node (parent node) has an associated local classifier c_p to recognize its nc_i child nodes. The final (leaf node) image prediction is chosen following a pathway of the most probable parent nodes, according to the classifications given by all c_p. As well-known, one of the main problems with this approach is the parent-to-child error propagation, that is, if the parent node is incorrectly classified, the prediction of the leaf node will be wrong. To reduce this problem, we propose criteria, based on the EU analysis, to decide whether the input image will be classified by a local or flat classifier. Our aim is to perform the classification by the LCPN approach only when the prediction given for all local parent classifiers is likely to be correct. Otherwise, the classification should be performed by a flat classifier. Our criteria are defined as follows:

- The 1st criterion consists of comparing the predicted label of the parent classifier c_p^l (with dropout turned off) with respect to the label given through the analysis of the EU along the child classifiers c_p^{l+1}, where l denotes the level of the hierarchy and takes values between 0 to $L-1$. Regarding the EU analysis, the process consists of estimating the EU of the input image using each c_p^{l+1} and, from this, determine the image label according to the label of the parent node corresponding to the c_p^{l+1} which provides the least uncertainty. We use the LCPN to perform the classification if both predictions are the same.

$$\textbf{C1}: pred(softmax(f^W(x_t))) = pred(\{\dots, EU_i(x_t), \dots\})$$

- The 2nd criterion is similar to **C1**, but this criterion considers the average of several softmax outputs, using the c_p^l with dropout turned on, to determine the prediction. The idea is to focus on those predictions that remain stable despite disabling any connections between neurons, and then compare them to the predicted label from the EU analysis.

$$\textbf{C2}: pred(\overline{softmax(f^W(x_t))}) = pred(\{\dots, EU_i(x_t), \dots\})$$

- The 3rd criterion is the strictest of our proposed criteria, where both **C1** and **C2** must be met to perform the classification with LCPN.

$$\mathbf{C3}: \mathbf{C1} \wedge \mathbf{C2}$$

- The 4th criterion is the most flexible of all criteria, where **C1** or **C2** must be met to perform the LCPN classification.

$$\mathbf{C4}: \mathbf{C1} \vee \mathbf{C2}$$

An example of the proposed prediction scheme can be seen in the Fig.4.2. To perform the prediction first, we must train all the models required for both approaches (LCPN and flat) and also the models used to estimate the EU, which may not necessarily be the same as those used for the LCPN predictions. Then, the final prediction is determined using one of the proposed criteria (in this case, **C3**). Since all predictions provide the same label, the final prediction is given by the respective c_p^{l+1} with dropout turned off. Note that we illustrate the scheme for two levels of hierarchy, but it can be easily extended to more levels.

4.3.2 Epistemic uncertainty estimation

The analysis of EU corresponds to a key element of our scheme (see green part of the Fig. 4.2). For this purpose, we consider the application of three simple but efficient methods: MC-dropout, Deep Ensemble and EDL. All of them quantify the EU through the entropy measure of the output probabilities, which is defined as follows:

$$EU(x_t) = -\sum_{c=1}^{C} p(y_c = \hat{y}_c|x_t) \ln(p(y_c = \hat{y}_c|x_t)), \tag{4.1}$$

where x_t denotes the input image, C is the number of classes, y_c is the predicted class and $\hat{y}_c$ is the ground-truth.

Note that $p(y_c = \hat{y}_c|x_t)$ used to estimate the EU depends on the method chosen:

- In the case of MC-dropout sampling, $p(y_c = \hat{y}_c|x_t)$ is obtained by performing K times predictions to the input image using the same classifier with the dropout turned on. Then, the average of the K softmax outputs provides the predictive probability for each class (see Equation (4.2)):

$$p(y_c = \hat{y}_c|x_t) = \frac{1}{K}\sum_{k=1}^{K} p_k(y_c = \hat{y}_c|x_t). \tag{4.2}$$

- As for the Deep Ensemble method, $p(y_c = \hat{y}_c|x_t)$ is calculated from K predictions on the input image, where K corresponds to the number of classifiers used in the ensemble. The $p_k(y_c = \hat{y}_c|x)$ given by all classifiers are averaged and then, in the same way as MC-dropout, the predictive probability is given by Equation (4.2).
- Finally, with respect to the EDL, unlike MC-dropout or Deep Ensemble, we need just one prediction to estimate $p(y_c = \hat{y}_c|x_t)$. Different from the previous methods, EDL classifiers do not directly provide prediction as probability because their architecture does not allow a softmax activation in the output. Instead, the logistic output ($f^W(x_t)$) for a sample x_t represents the evidence vector predicted by the network. Then, the estimation of the class probability is given by the following equation:
$$p(y_c = \hat{y}_c|x_t) = \frac{\alpha_i}{\sum_{i=1}^{C} \alpha_i}, \tag{4.3}$$
where $\alpha_i = f^W(x_t) + 1$ corresponds to the parameters of the Dirichlet distribution.

4.3.3 CNN Model

In order to use the proposed scheme, we must consider two aspects in the architecture of the classifiers. The first applies to all the classifiers to be used in the LCPN approach, which consists of using CNN models with a dropout layer after each hidden fully connected layer. The second one, required to quantify the EU based on EDL method, corresponds to the application of an activation to ascertain non-negative output [146]. For the latter, we use a RELU activation.

4.4 Experiments

In this section, we first describe the datasets, second we show the experimental setup and finally we describe the results.

4.4.1 Datasets

The proposed method is validate in two public food datasets:

MAFood-121 [3] is a multi-task food image dataset comprising 3 related tasks: a) dish, b) cuisine, and c) categories/food groups. The dataset consists of 121 dishes from the top 11 most popular cuisines. In total, 21.175 images are gathered from 4 different sources, 3 public food datasets [23, 81, 56], and Google Search Engine. As for the distribution, 72.5% of the images are used for training, 12.5% for validation and 15% for test. Regarding the experiments, we consider all the images with their dish and cuisine annotations, respectively.

Food-101 [23] is one of the most popular food datasets, with 101 international food classes and 1,000 images each. The dataset is distributed with 75% of the images for training and the rest for testing. Unlike MAFood-121, this dataset only has the annotations for the class name. Since our method requires a hierarchical annotation architecture, we added a new one that groups dishes into a high-level attribute called food group (see Sec.4.4.2).

4.4.2 Experimental setup

In order to evaluate our method, we adopt ResNet-50 [59] as the base CNN architecture for all the classifiers. This model was modified by removing the output layer, and instead, we added one hidden fully connected layer of 2048 neurons, followed by a dropout layer with a probability of 0.5 and finally an output layer with softmax activation.

For our experiments, since the tree structure for the LCPN in the target datasets has three levels (see Fig. 4.1), we denoted three types of classifiers: 1) Local Parent Classifier (LPC), corresponding to the classifier trained for the first level of the hierarchy; 2) Local Child Classifiers (LCCs), corresponding to the classifiers trained for the second level; and 3) Flat, the classifier trained on all leaf nodes. Specifically, these types of classifiers were trained for the following purposes:

1. *LPC*: CNN Model to perform the cuisine (MAFood-121) or food group (Food-101) classification.
2. *LCC*: CNN Model to perform the local dish classification. In the MAFood-121 case, for the cuisines: American (am), Japanese (ja), Italian (it), Greek (gr), Turkish (tu), Chinese (ch), Mexican (me), Indian (in), Thai (th), Vietnamese (vi) and French (fr). In the Food-101 case, for the food groups: Egg, Soup and Pasta (fg1); Seafood, Rice, Fried food (fg2); Bread, Dip and Dumpling (fg3); Meat, Vegetable and Dairy products (fg4); and Dessert (fg5).
3. *Flat*: CNN Model to perform the flat classification on all the dishes.

Regarding the neurons used in the output layer, they were assigned by the number of child nodes of the respective parent node. In the MAFood-121 case, there were 121 neurons for the Flat model and 11 for the LPC and LCC models. As for Food-101, there were 101 neurons for the Flat model, 5 for the LPC and 20 for the LCC models from fg1 to fg4 and 21 for fg5.

With respect to the models used to estimate the EU, the same LCC models trained for the LCPN were used for the MC-dropout method (with $K = 100$). For the Deep Ensemble, we trained the LCC models five times with the same training parameters, but changing the initialization of the weight and shuffling the input images. For the EDL case, we

TABLE 4.2: Cuisine classification results on MAFood-121 in terms of accuracy taking into account only the minimum entropy criterion.

Cuisine	MC-dropout	Deep Ensemble	EDL
am	55.30%	71.46%	74.49%
ch	49.81%	48.25%	78.72%
fr	43.39%	80.34%	62.80%
gr	40.31%	71.20%	68.06%
in	50.00%	61.58%	85.18%
it	42.42%	62.09%	76.27%
ja	62.77%	70.57%	68.05%
me	64.54%	75.72%	64.21%
th	46.77%	66.67%	59.68%
tu	80.09%	86.06%	67.89%
vi	59.59%	71.50%	83.73%
all	54.09%	69.59%	**71.73**%

replaced the base architecture output layer with a RELU output layer, and then trained once the LCC models like in Deep Ensemble.

Concerning the loss function, we used the categorical cross-entropy loss, for the majority of the models, with the exception of the models trained to estimate EU by EDL. In this case, Equation (5) defined in [146] was applied. For both losses, the Adam optimizer was selected.

As for the training process, the Flat model was re-trained from a pre-trained model on ImageNet [91], during 50 epochs with a batch size of 32, and an initial learning rate of $1e-4$. With respect to the LPC and LCC models, they were re-trained just on the layers after the last convolutional layer from the pre-trained Flat model, during 32 epochs with a batch size of 32, and an initial learning rate of $1e-5$. In the case of the EDL models, we did not apply decay. Otherwise, we applied a decay of 0.5 every 8 epochs. Regarding the data augmentation process, traditional strategies like random crops and horizontal flips were applied. The training was done using Keras with Tensorflow as backend.

Finally, the standard metric for object recognition, overall Accuracy was used to evaluate and compare the proposed method with respect to both approaches: LCPN and Flat.

4.4.3 Results

In this section, we present the results obtained on the MAFood-121 and Food-101 by the LCPN and Flat classification approaches individually, and also by our proposed method, which integrates both approaches during image prediction.

Regarding MAFood-121, to determine the cuisine to which the image belongs, our approach contemplates the EU analysis of a particular image when it is sent to numerous

TABLE 4.3: Results on MAFood-121 in terms of Accuracy.

		Cuisine		**Local**		**Flat**		**Overall**
Approach	**EU method**	#Imgs	*Acc*	#Imgs	*Acc*	#Imgs	*Acc*	*Acc*
GT+Local	-	3177	100.00%	3177	89.61%	-	-	89.61%
LCPN	-	3177	87.19%	2770	91.71%	-	-	79.96%
Flat	-	-	-	-	-	3177	81.37%	81.37%
our	MC-dropout	1579	96.96%	1531	96.08%	1598	70.21%	**81.62%**
our	Deep Ensemble	2039	96.34%	1967	**96.68%**	1138	62.89%	**81.62%**
our	EDL	2095	96.24%	2019	96.50%	1082	61.74%	81.52%

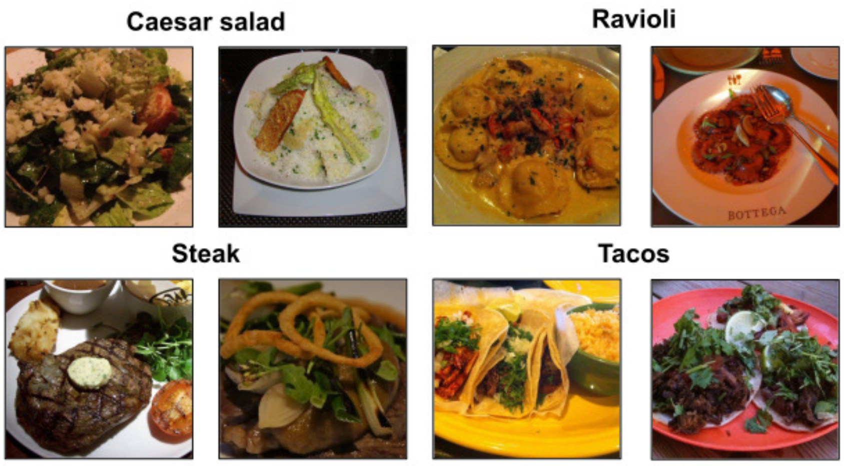


FIGURE 4.3: Samples of the Smallest and Largest EU within the same food class.

local classifiers, one for each type of cuisine. The fact that the EU is explained with sufficient data suggests that if the features extracted from an input image are similar to those learned by a model, EU should be small, implying that this image most likely belongs to one of its classes.

Just analyzing the EU, we can achieve impressive results in cuisine prediction (see Table 4.2), but far from what a local classifier provides. The cause is because the shared features among different cuisines harm the prediction based on the EU. For this reason, we consider EU as a complement to the cuisine recognition classifier, rather than directly applying it to determine the cuisine of the image. The same occurs in Food-101, but for the food group classification.

The cumulative distribution function (CDF) for the entropy of the predictive out-of-distributions and in-distribution for six local cuisine classifiers can be seen in the top and bottom rows of Fig. 4.4 respectively. The estimated entropy considers all images in the test set that belong to a different (out-of-distributions) or equal (in-distribution) classes contemplated for the training of each local classifier. As expected, we can see that the

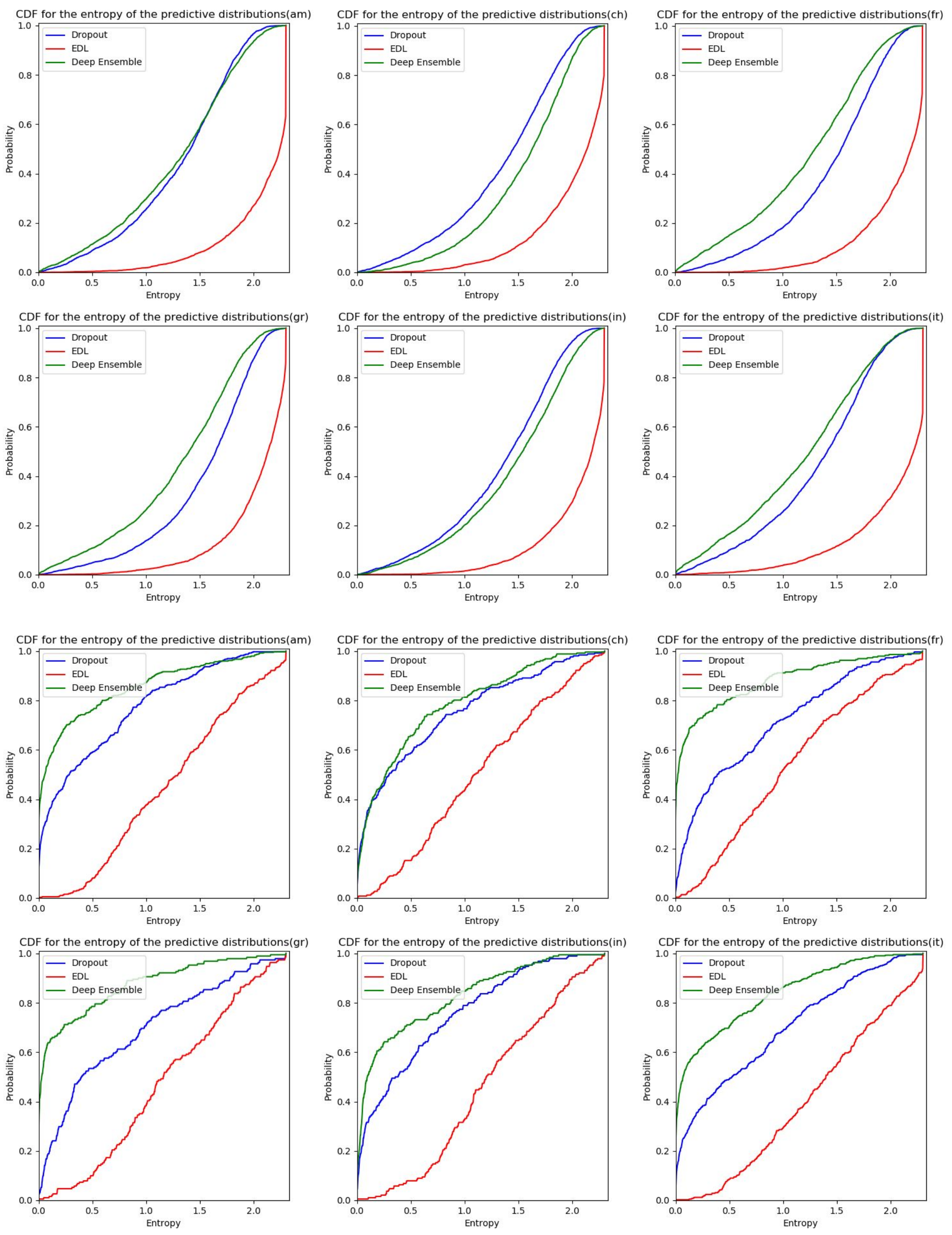


FIGURE 4.4: CDF for the entropy of the predictive distributions on the out-of-distribution (1st & 2nd row) and in-of-distribution (3rd & 4th row) cuisines on MAFood-121.

EU tends to be small when the test images are similar to the images used in the classifier training (see bottom of Fig. 4.4). However, despite belonging to the same distribution, it is possible to obtain high value for the EU. We illustrate the latter in Fig. 4.3, where we show a sample for some classes in which we get the largest and smallest estimate of EU.

TABLE 4.4: Results on Food-101 in terms of Accuracy.

		Food Groups		**Local**		**Flat**		**Overall**
Approach	**EU method**	#Imgs	*Acc*	#Imgs	*Acc*	#Imgs	*Acc*	*Acc*
GT+Local	-	25250	100.00%	25250	87.96%	-	-	87.96%
LCPN	-	25250	91.90%	23205	89.49%	-	-	82.24%
Flat	-	-	-	-	-	25250	83.26%	83.26%
our	MC-dropout	11611	94.03%	10918	88.62%	13639	83.22%	83.28%
our	Deep Ensemble	20534	97.00%	19918	**93.74%**	4716	54.26%	**84.08%**
our	EDL	20898	96.94%	20259	93.10%	4352	53.70%	83.96%

In Table 4.3 and Table 4.4, we summarize the results achieved by the four approaches evaluated on MAFood-121 and Food-101 datasets: a) GT+Local, to reflect the performance of the flat classifier (dish) when we have a perfect LPC; b) LCPN, considered as baseline when we chose LCC directly from the LPC prediction; c) Flat, which performs recognition with a classifier trained with all classes; and d) Our model, which integrates LCPN and Flat approaches taking into account the EU to decide the classifier responsible for the final prediction. For the latter, we show three results, one for each method used to estimate EU. The results obtained show that it is possible to achieve a large increase in accuracy when we consider a perfect LPC recognition and perform the final prediction with the LCC (see GT+Local). Instead, due to error propagation, the lowest classification accuracy is achieved when we choose the LCC directly from the LPC prediction even though the LPC provided high performance (91.71% on MAFood-121 and 89.49% on Food-101). Regarding our scheme, we show that it allows to reduce the miss-classification produced by errors in LPC predictions regardless the method used to estimate the EU. From all of them, the best result is achieved when the EU is quantified by the Deep Ensemble method. This method not only has the best overall accuracy, but also provides the best result by the local classifiers. Furthermore, we are able to recognize well, using local classifiers, more than 60% of total images in both datasets. Interestingly, regardless of the method used to estimate the EU, our method outperforms the classification results obtained through LCPN and Flat approaches.

Finally, in Table 4.5, we can see how the different criteria considered in our scheme affect the overall accuracy. When we use the Deep Ensemble method to estimate the UE, both $C1$ and $C2$ provide better performance than the LCPN and Flat approaches in both datasets. However, the criterion applied in the best performance achieved differs between the datasets. From the results, we note that the better the LPC, the better the overall accuracy can be with $C4$, otherwise with $C3$.

TABLE 4.5: Accuracy of our proposal for both datasets, according to the criteria applied, using the Deep Ensemble method to estimate uncertainty.

Criteria	MAFood-121	Food-101
$C1$	81.55%	83.58%
$C2$	81.40%	83.60%
$C3$	**81.62%**	83.10%
$C4$	81.33%	**84.08%**

4.5 Conclusions

In this chapter, we demonstrate the benefits of our proposal to perform food recognition by the integration of a hierarchical classification with a flat classifier and uncertainty modeling. The proposed method deals with the propagated error from parent to child nodes in a hierarchical classification approach. To tackle this, we use local classifiers as long as it is very likely that the prediction will be good. Otherwise, the classification is performed with the flat classifier. For this purpose, we define novel criteria that combine local classifier predictions (with dropout turned on or off) with the analysis of the EU. We observed that our approach provides a good performance, not only using MC-dropout to estimate the EU, but also when we use Deep Ensemble and EDL methods. In all cases, the proposal allowed us to further reduce the propagation error. As a conclusion, we show the high accuracy and flexibility of the proposed approach that does not depend on a particular EU estimation method, which allows us to face the prediction of a large number of food classes in a beneficial way. Finally, as future work, we will explore the application of EU within an active learning framework to ease the process of image annotation necessary to achieve food recognition of thousands of different food classes.

Chapter 5

Uncertainty-aware data augmentation for food recognition

5.1 Introduction

In the current world society, sometimes unhealthy habits are disregarded in favour of trying to optimize our time by spending less time on tasks that we do not consider essential. Cooking healthy food, or even looking for what is healthy and what is not when we buy groceries is not always an action on which we are willing to spend enough time. Furthermore, not only the lack of time, but also the lack of knowledge or awareness about the nutritional value of food that we eat during our daily life affects our eating patterns [10]. Following unhealthy food habits is a situation too common in developed countries, and in the long term, they lead to chronic medical conditions like diabetes, cardiovascular diseases or obesity. Additionally, even if we are fully aware, it is not easy to manually calculate the nutritional information associated to the food that we eat [142]. The development of automated systems that aid on logging each food that we eat daily could help, in most of the cases, to prevent us from following unhealthy eating behaviours [24]. At the same time, these methods could be transformed into tools oriented to both healthcare and nutrition professionals in order to better understand the eating problems of a certain individual or, in a larger scale analysis, even of the society as a whole [43].

The creation of automated food recognition algorithms would, at the same time, help in several other daily life problems. A clear example of this would be applying them to create self checkout systems for self-service restaurants [6].

Since the last decade, deep learning has become a must when developing new algorithms that aim for offering solutions in real-world scenarios. This popularity becomes even natural when comparing their extraordinary results against more traditional methods [157]. Even though, from a machine learning perspective, dealing with the food recognition problem offers several challenging tasks due to the nature of food images (Fig.

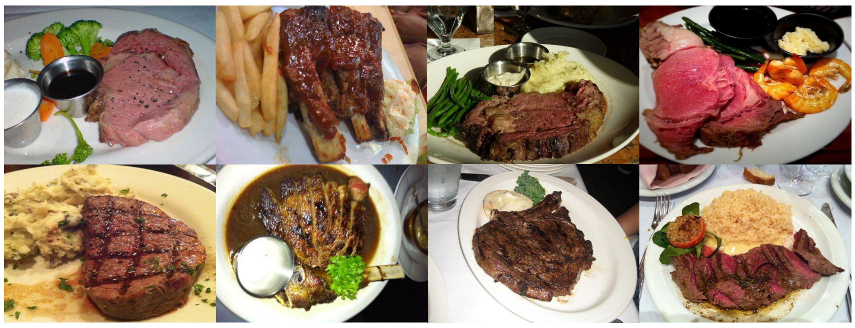


FIGURE 5.1: Example of high intra-class (left-right) and low inter-class (bottom-top) variability in food images for prime_rib (top) and steak (bottom) classes.

5.1). In addition, we must never forget other problems like data over-fitting, recurrent when working with deep learning models. In order to avoid it, we must overcome the challenge of acquiring datasets diverse enough that are able to represent all the possible aspects of real-world data.

When data is scarce (e.g. less than 1000 images per class), data augmentation is a mandatory technique that must be considered in model training in order to increase sample size and variability. In this way, we can prevent the classifier from being over-fitting and take better advantage of the learning capabilities provided by deep learning algorithms. Traditional data augmentation methods are applied in training time by performing several transformations (e.g. flips, cropping, rotation, etc.) on each image before sending it to the classification model. However, are all data augmentation techniques appropriate for any types of data? Do we really need to increase all the data interchangeably? Can there be data that require more attention than others? We argue that, within a dataset, on the one hand there could be some techniques that can be beneficial or detrimental for training, depending on the type of image that will be analyzed. For example, in the context of food recognition adjusting the color balance could be counterproductive, when we have to learn the distinctive features of near classes as *white rice with brown rice* or *pumpkin soup with carrot or tomato soup*, among other. On the other hand, there could be more and less complex classes; or some images could be more beneficial for the classification than others; therefore more emphasis should be placed on these data. For instance, using by additional data augmentation methods to hard samples we could help the model learn the distinctive features that characterize them. So the next questions are: how can we discover the appropriate data augmentation techniques according to the type of data? How can we discover hard samples to classify correctly within a dataset? Uncertainty modeling is a research field that can help us determine those techniques or data samples that the model has not yet been able to learn.

In this work, we proposes three uncertainty-based method. In the first, a class-conditional data augmentation method is proposed. where we observe the variation of epistemic uncertainty for the samples of a particular class when we change data augmentation strategies to determine which method is likely to be beneficial for each class. The second is a method that explore a combination of both fields: GANs and uncertainty modelling, with the aim of generating new data focusing on the samples that the model has not been able to learn well (with high uncertainty). The last one correspond to train a Convolutional Neural Network following the principles applied in an active learning framework, where the new data to be added corresponds to synthetic data generated on the training data set that meet our uncertainty-based criteria.

Our major contributions are: a) we provide an epistemic uncertainty-based method, which allows incorporating specific image processing techniques, chosen according to the class in which the input images belong. b) we demonstrate the importance of focusing on an uncertain sample to perform data augmentation, instead of applying it on the all training data; and c) we provide an uncertainty-aware method based on an active learning framework, which allows to get an improvement in the performance and a better balance in the classification on datasets with unbalanced data. The rest of the work is organized as follows: Section 5.2 presents the recent literature related to our work. Section 5.3 details the proposed methods. Section 5.4 describes the experiments. Finally, Section 5.5 provides the conclusions and final remarks.

5.2 Related Work

In this section, the most relevant recent literature related to the research fields of the proposed methods is discussed.

5.2.1 Food Recognition

Automated food recognition is a challenging computer vision task. Food images are very complex [166] - they may be mixed or composed of many food items in a single plate. The food classes have high intra-class variance and low inter-class variance (see Fig. 5.1), making them fine-grained in nature.

Early food recognition literature works dealt with the problem in a constrained environment. Some of the tasks involved using different hand-crafted features such as color, texture and shape [112, 73, 22]. Public food images datasets were available as early as 2009 [29]. However, these initial datasets had fewer images and/or small number of classes [32, 112, 27].

With the advent of CNNs, the task of food recognition handled more complex data. Different CNN architectures have been applied either directly or modified for the task of food recognition. AlexNet [173], GoogLeNet [170, 104], Network-In-Networks [162], Inception V3 [58], ResNet50 [120], Ensemble NN [124], Wide Residual Networks [111], and CleanNet [97] are some noteworthy examples. With more computation capabilities, networks were able to handle large image sizes [116] and complex networks such as fusion of CNNs [1], Teacher-Student learning [189] and attention networks [119] have been developed and have also proved effective. Current food image datasets cover a wider variety of classes and also have a larger number of images in each class [23, 33, 41].

With more diverse food cuisines, food recognition has moved from single-labeled to multi-labeled data, which also increases the complexity of the problem. To handle the complexity of the task, different approaches such as using food detection networks preceding the food recognition networks [12], using food localization and recognition [21], food segmentation networks [11, 28], and others semantically combining both segmentation and detection networks [6] were proposed. Multi-task learning [3, 193] networks are the latest interests in the domain of food recognition.

One of the major challenges for creating models with better recognition performance is the diversity of images. Collecting large datasets with more classes and more images are difficult and it is also not possible to have images that differentiate the high inter-class similarity of food classes. To tackle this challenge, data augmentation strategies can be helpful in increasing data and its variability.

5.2.2 Data Augmentation Strategies

Diversity of images used for training determines the performance of the network. The more diverse the images are, the better generalization performance the model obtains. Data augmentation is used to create more variety of synthetic data from real images. Augmentation strategies fall under geometric transformations, color space transformations, kernel filters, mixing images, random erasing, feature space augmentation, adversarial training and GAN-based augmentation [165].

Flipping, cropping, rotation and translation are common geometric transformations. Adding noise to images improves the robustness of the model [122]. The Synthetic Minority Oversampling Technique [25] selects and joins nearest neighbors to create new data. Random erasing [192], mixing images by random cropping and concatenation [158] and Kernel filters [78] have shown interesting and promising performance improvements. In all the above literature, data augmentation schemes are proven effective, but they require a manual selection of the method. Automatic selection of the augmentation policy that increases the model performance was proposed as AutoAugment [35] and later a faster search scheme was proposed as Fast AutoAugment [101]. Data Augmentation using these schemes is able to generate synthetic data, but they always are confined within the space of the original images.

GANs are generative networks used to create synthetic images from an existing distribution which can be different from the source images. Traditional GANs generate images either unsupervisedly or following a condition. However, with more deeper architectures, more realistic images have been possible [183]. Recent GAN networks are designed to handle imbalance in datasets. BAGAN [110] uses a framework to handle unbalanced datasets by augmenting images from minor classes. Fine-grained generation and classification of minority classes are handled in MFC-GAN [9]. GANs are also able to learn from single images [147] to generate the synthetic ones.

GAN-based augmentation has been applied to generate food images from source datasets [65, 68]. Synthetic food images were also created from lists of ingredients [57] or from the recipe of the food [194]. With more diverse food images, it was also possible to create food classes that are unseen during training [64]. However, all the food-based GANs require considerably large training sets.

We argue that by understanding better the model uncertainty, it is possible to select the dataset part that would be effective to augment and thus to improve the classifier performance.

5.2.3 Uncertainty Modeling

Uncertainty can simply be defined as ignorance about the behavior of the trained model with respect to a dataset. This can be categorized as either aleatory or epistemic [82]. The aleatory uncertainty case corresponds to the uncertainty that is present in almost all data, and reflects the noise that may be present in the observations (e.g. random errors produced by the measuring instrument). The epistemic uncertainty case corresponds to the model uncertainty and can be reduced by gathering more data, leading to a better fit of the distribution.

In Deep Learning, stochastic variational inference has been adopted to capture uncertainty rather than directly using a Bayesian neural network, because the latter becomes intractable in deeper networks [19, 50, 146]. Among the successful variational inference methods applied [19, 50, 121, 107], the most popular is the Monte Carlo (MC)-dropout [50], which provides a simple way to estimate uncertainty.

There are several recent image classification works that consider uncertainty modeling in their method. Some of them incorporate the uncertainty in the loss function to weigh different losses in a multi-task learning framework [82, 3], or to guide the learning taking into account the difficulty of each class or image [84]. Others analyze uncertainty in order to decide the method of data augmentation that can best contribute to learning the discriminating features of a particular class [5], to judge when to use a hierarchical or flat classifier to give the final prediction [4] or to incorporate new synthetic images generated by SINGAN from highly uncertain training images [7]. However, more related to our work are those who use uncertainty in an active learning framework [51, 16, 126]. In all of them the uncertainty is analyzed to identify samples that are likely to contain new information, instead of selecting them randomly and, therefore, should improve the model performance. Unlike [51, 16], in [126] the new samples correspond to synthetic images generated by a GAN model trained on the real ones. On the other hand, instead of selecting uncertain samples, in [113], the authors propose an active learning method that considers generating such samples using a trained GAN.

Our proposal differs primarily from the uncertainty-based active learning approach in the following: a) After each training cycle, we analyze the uncertainty to select an appropriate set of images from the training set, taking into account those that meet our uncertainty-based criterion C_1, and from them generate new synthetic ones; b) the new image automatically has the same label as the real one, and therefore we do not need an external oracle to annotate the labels.

5.3 Uncertainty-based Data Augmentation Methods

In this section, we describe the three proposed uncertainty-based data augmentation methods: Class-Conditional Data Augmentation (CCDA), Uncertainty-Aware GAN-augmented Food Recognition (UAGAN), and Uncertainty-Aware Data Augmentation (UDA).

5.3.1 Class-Conditional Data Augmentation

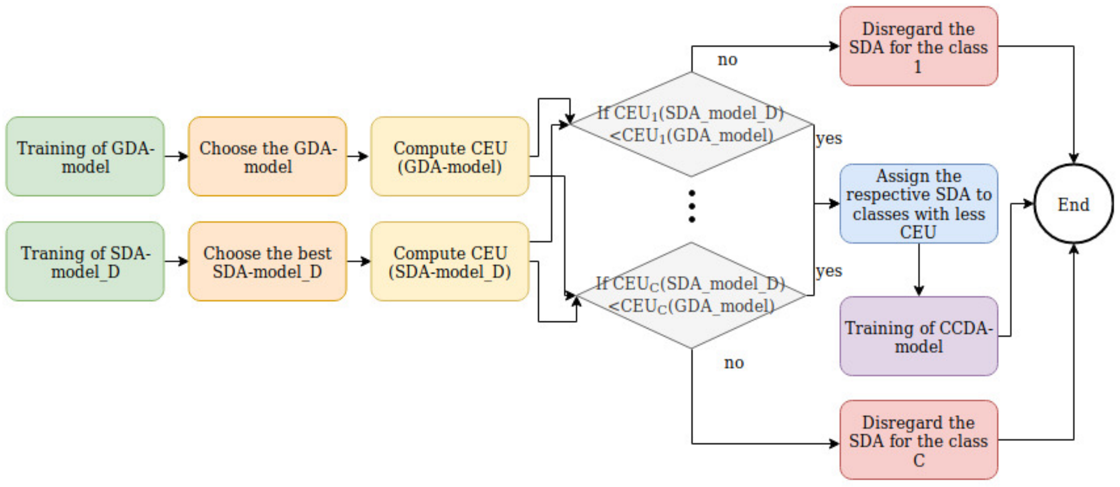

Training of GDA-model
Choose the GDA-model
Compute CEU (GDA-model)
Traning of SDA-model_D
Choose the best SDA-model_D
Compute CEU (SDA-model_D)
If CEU₁(SDA_model_D) <CEU₁(GDA_model)
If CEU_C(SDA_model_D) <CEU_C(GDA_model)
no
yes
yes
no
Disregard the SDA for the class 1
Assign the respective SDA to classes with less CEU
Training of CCDA-model
Disregard the SDA for the class C
End

FIGURE 5.2: Flowchart of the proposed class-conditional data augmentation procedure. SDA_model_D denotes the model which incorporate the Dth specific data augmentation technique (eg. color, contrast) into the generic data augmentation process.

In this subsection, we explain each step involved in the proposed approach, which takes advantage of the epistemic uncertainty intrinsic in the data to determine the type of data augmentation process applying to the images during the training of the model. We argue that some data augmentation techniques may depend on a particular class. To validate it, we focus on traditional transformations and propose group them into two subgroups: generic data augmentation techniques (GDA), those that apply geometric transformations to images (e.g. crops, reflection, zoom, etc.), which are independent of the image class; and specific data augmentation techniques (SDA), those based on the pixel transforms of the images (e.g. color, brightness, contrast, etc.), which are class-dependent.

The MC-dropout method was adopted in our scheme to determine the epistemic uncertainty. In practical terms, it means to predict K times the same image using the model with the dropout layer turned on. Then, EU will correspond to the predictive entropy calculated from the K predictions given.

Formally, EU can be expressed as follows:

Let $\overline{p(y_c = \hat{y}_c|x)}$ be the average probability that the prediction y_c is equal to ground-truth $\hat{y}_c$ given the image x, calculated from K MC-dropouts simulations. Then,

$$EU(x_t) = -\sum_{c=1}^{C} \overline{p(y_c = \hat{y}_c|x_t)} \ln(\overline{p(y_c = \hat{y}_c|x_t)}), \tag{5.1}$$

where

$$\overline{p(y_c = \hat{y}_c|x)} = \frac{1}{K}\sum_{k=1}^{K} p(y_c^k = \hat{y_c^k}|x). \tag{5.2}$$

As we discussed earlier, our intention is to determine, through the EU, which data augmentation technique is most appropriate for the images of a particular class. By definition, the EU tends to reduce when we increase the samples in the dataset. So, if we are able to reduce the EU for some classes, when we incorporate specific data augmentation techniques, it is likely that these techniques contribute to the learning of these classes. Therefore, in our case, it is interesting for us to know the average EU considering all the samples within each class, we will name it as Class Epistemic Uncertainty (CEU). In the proposed method, CEU is used to compare between a model trained with GDA with other models that incorporate the SDA to the generic one. Then, we consider that a specific technique is appropriate for a class when the model trained with this technique incorporated in the data augmentation process obtains the lowest CEU value.

The process involved in our scheme to determine the SDAs is as follows (see Fig. 5.2):

1. A model that considers generic data augmentation technique (GDA-model) is trained with the aim of being compared with the models that also incorporate a specific data augmentation technique.

2. A model is trained for each specific data augmentation technique (SDA-model) that we would like to incorporate into the process. Each model adds only a specific technique to the generic one, which will be used during the training.
3. The best model is chosen with respect to the accuracy obtained in the validation set for the models trained in steps 1 and 2.
4. CEU is computed for all models in step 3.
5. With the results in step 4, the GDA-model is compared with the SDA-model. To those classes in which the SDA-model obtains a lower CEU than the GDA-model, we assign them the respective specific technique used by that SDA-model.
6. Finally, considering the specific techniques selected for each class from step 5, we train a new model using data augmentation with generic techniques for all classes and also the selected one for each class.

5.3.2 Uncertainty-Aware GAN-augmented Food Recognition

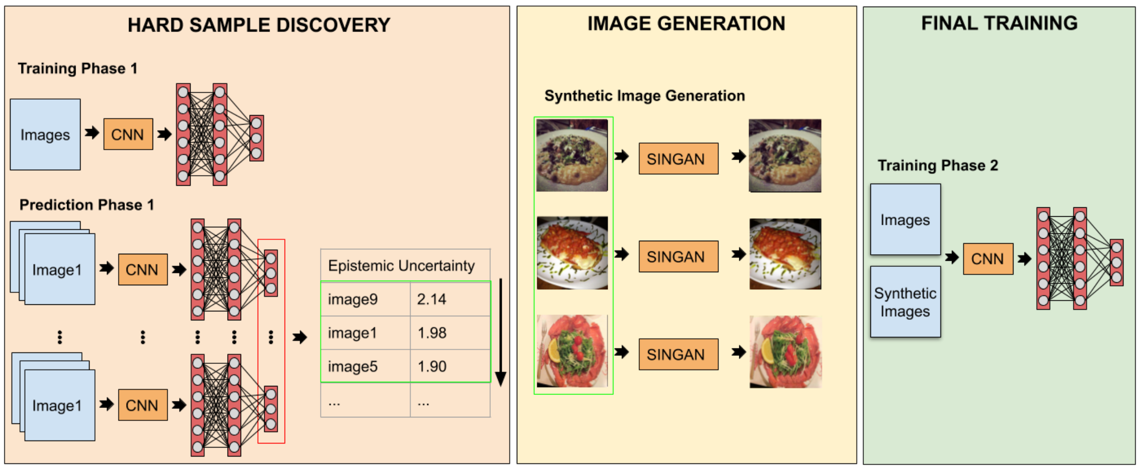


FIGURE 5.3: Main scheme of the our UAGAN food recognition method.

In this subsection, we describe all phases involved in the Uncertainty-Aware GAN-Augmented method to perform food recognition using uncertainty modeling and GANs. As you can see in Fig. 5.3, the method contemplates 3 main phases with the following purposes: a) hard samples discovery, b) synthetic image generation and c) final training.

The first step of our proposed approach involves the analysis of the food images of the training set, with the aims of identifying those that are hard to classify. To do this, our criterion is based on the analysis of Epistemic Uncertainty through the calculation of the entropy. The samples with high uncertainty are those in which the model has not been able to learn well their discriminant features and, therefore, are considered hard samples. On the other hand, we adopt the method called MC-dropout [50] for EU estimation, mainly due to its simple implementation. Basically, we need to add a dropout layer before each fully connected layer, and after the training, we perform K predictions with the dropout turned on. The K probabilities (softmax outputs) are averaged and then the entropy is calculated to reflect the EU. Finally, the images are ordered with respect to their EU, and we select the top n images with higher EU to perform the next step.

Once the images have been chosen, the next step corresponds to increasing the data with nearby images in terms of visual appearance. We believe that one of the determining factors that does not allow the model to learn the features of hard images corresponds to the fact that they differ from most images that represent a particular class. This hard images may be present in the training set due to the complexity of the acquisition and also after dividing the data for the training. The latter is due to the fact that during the generation of subsets only the sample size is considered and not the variability of the sample. Therefore, we propose to make new images by applying small changes to the original ones. The best method for this purpose is the recent GAN-based method called SINGAN [147], which can learn from a single image and generate different samples carrying the same visual content of the input image. In this step, we adopt the SINGAN to generate one synthetic image for each chosen image according to the uncertainty criterion.

Finally, in the last step, the whole CNN model is trained with both types of images: the synthetic images obtained with SINGAN and the original images.

5.3.3 Uncertainty-Aware Data Augmentation

In this subsection, we describe the proposed Uncertainty-aware Data Augmentation method for model training.

The motivation of our work is that when homogeneous data augmentation across the dataset is no longer useful for improving performance, we should focus on data samples that the model has been unable to learn well in order to improve it further. The idea behind this is that not all images are equally complex. Thus, for difficult images we need to provide more data samples with alternative visual appearances in order to ease the model learning for this type of content. To guide the model learning, the proposed method follows an active learning framework taking into account the epistemic uncertainty. Unlike other approaches, we do not analyze uncertainty in new unlabeled images in order to incorporate them into the training set. Instead, we focus on increasing the data by generating new images from the original ones.

Let us consider $L_i = \{(x_j, y_j)\}$ the set of labeled images from the training set and M_i the model trained in the i-th training cycle, where M_0 corresponds to the model trained with the real images L_0. After completing each training cycle, we can identify food images that are difficult to classify based on Epistemic Uncertainty analysis by calculating the entropy as $EU_i(x_j) = \sum_{y_j} -P_i(y_j|x_j)log(P_i(y_j|x_j))$. In CNN models, the easiest way to estimate the EU of the image x_j, is through the MC-dropout method [50], which performs, during the prediction, K forward passes with dropout turned on. From this, we can obtain $P_i(y_j|x_j)$ averaging the K softmax outputs. Note that when the images into the dataset are not properly curated, the most uncertain images are most likely outliers. For this reason, instead of directly using the most uncertain images, we define an upper threshold T_u, to avoid selecting and generating new images from out-distribution data. Furthermore, we also define a lower threshold T_l to avoid the generation of data from images that may not contribute to recognition, making the training process faster and more efficient. Then, after each cycle, we select the real images that meet with the criterion $C_1 : EU(x_j) > T_l \land EU(x_j) < T_u$, and from them generate the new synthetic images x_j^* (or acquire new images) with a visual appearance close to the real ones. After that, we incorporate the new images in the training set ($L_{i+1} = L_i \cup \{(x_j^*, y_j)\}$) and retrain the full model. The training cycle is repeated until satisfactory results are met.

Algorithm 1: UDA procedure

```
input: Labeled Data $L_0$, Synthetic Data $S_i \leftarrow \emptyset$, Lower Threshold $T_l$, Upper
  Threshold $T_u$, Generator $G$;
while we are not satisfied with the performance do
    Train the classifier $M_i$ on $\{L_0 \cup S_i\}$;
    for $x_j \in L_0$ do
        Calculate the $EU(x_j)$;
        if $T_l < EU(x_j) < T_u$ then
            Create the synthetic image $x_j^*$ with $G$;
            Update $S_i \leftarrow S_i \cup \{x_j^*\}$;
        end
    end
end
```

The UDA training procedure is summarized in Algorithm 1 and involves the following steps:

1. Train a CNN model M_i on training set L_i.
2. Select the images that meet the criterion C_1.
3. Acquire images or generate synthetic ones from the real images (x_j) that meet the criterion.
4. Incorporate the images into the training set (L_{i+1}).
5. Repeat the steps until we meet a satisfactory result.

5.4 Validation

In this section, we describe the datasets chosen to validate the proposed CCDA, UAGAN and UDA methods; the metrics used to perform the analysis of the trained models; the experimental setup and present the results obtained with the different approaches evaluated.

5.4.1 Food datasets

The three public datasets chosen to validate our methods are described below. Specifically, the food datasets: Food-101 and UECFOOD-256, were chosen for CCDA; and MAFood-121 for UAGAN and UDA.

Food-101

This is an international food dataset which contains the top 101 most popular food plates[23]. For each dish, the dataset provides of $1,000$ images collected from foodspotting.com. The distribution of the dataset given by the authors is of 75% for training and the remaining 25% for testing. In our case, we keep the same data for the test set, but we divide the train set leaving 90% for training and 10% for validation.

UECFOOD-256

This is a food image dataset, containing 256 dishes and $31,395$ images [81]. The dataset mainly includes Japanese dishes, but also consists of foods from several countries such as China, France, Indonesia, Italy, Thailand, United States and Vietnam. Each image has annotations for the type of dish and also the bounding box information where the dish is located. In addition, the authors provide a 5-fold cross validation annotations that include $28,280$ images, with a minimum, maximum and average of images per class of $87, 637$ and 110.47. In our case, we used the first fold for the test and, from the remaining four folds, we randomly selected about 90% for training and 10% for validation. Note that, the object (food) bounding box information is not used in our experiments.

MAFood-121

This is a multi-attribute food dataset comprising of 21,175 images belonging to 121 food dishes distributed evenly across 11 different types of cuisine. The dataset contains 72.5% images for training, 12.5% for validation, and the remaining 15% for testing. Particularly for UAGAN, we use all the images of the dishes that belong to the Italian cuisine. In total, 11 dishes were chosen, which are composed of 2468 images with a maximum, minimum and average of 250, 104 and 224 images, respectively. The data within this class group is distributed as 72% of the images for training, 11% for validation and 17% for test. In the UDA case, all dataset was used.

Note that one of the main challenges for the deep learning algorithm is being able to learn the distribution of data from small datasets. Here, data augmentation is a must in order to avoid over-fitting and improve the performance. To highlight the benefits of our proposals (UAGAN and UDA), we chose the food dataset MAFood-121 [3], mainly because it has a small or medium number of images per class (from 100 to 250) and provides annotations that allow us to divide this dataset and independently evaluate our approach into any of the 11 semantically coherent class groups.

5.4.2 Evaluation metrics

In order to evaluate the performance and highlight the benefits that the proposed methods provide, we define particular metrics for each.

Metrics for CCDA

The standard metric for object recognition, overall Accuracy, and the mean Epistemic Uncertainty are used to evaluate our approach. As for the latter, the maximum value depends on the numbers of classes, and for this reason we normalized it leaving the possible values between zero to one, which we call Normalized Epistemic Uncertainty (NEU). Both metrics are formally defined as follows:

$$Acc = \frac{1}{T}\sum_{c=1}^{C} TP_c, \tag{5.3}$$

where C is the number of classes, TP_c (True Positives) is the amount of images belonging to the class c classified correctly, and T is the total images evaluated.

$$NEU = \frac{1}{T}\sum_{t=1}^{T} \frac{EU(x_t)}{\ln(C) - \ln(1)}, \tag{5.4}$$

where $\ln(C) - \ln(1)$ corresponds to the normalization factor obtained when the average prediction for each class is equal to a random prediction.

Metric for UAGAN

Overall Accuracy (Equation 5.3) is used to evaluate our proposal. We running 5 times the same experiments and show the result in terms of mean accuracy and the respective standard deviation.

Metrics for UDA

Taking into account that we face a multi-class classification problem in an unbalanced dataset, we decided to evaluate the model performance based on the recall metric using both macro- and micro-averaging, with the aim of understanding the behavior of the model at the class and sample level. Formally, the metrics used are defined as:

$$R_{micro}(y, \hat{y}) = \frac{|y \cap \hat{y}|}{|\hat{y}|}$$

and

$$R_{macro}(y, \hat{y}) = \frac{1}{|C|} \sum_{c \in C} R_{micro}(y_c, \hat{y_c})$$

where y, $\hat{y}$, C denote the set of predicted labels - ground-truth and classes - for all our data; and y_c and $\hat{y_c}$ correspond to the subset of y and $\hat{y}$ specifically with class c.

Note that the results shown correspond to the median performance when running each experiment 5 times.

5.4.3 Experimental Setup

All experiments have been carried out by adopting ResNet50 [59] as the base CNN architecture. To calculate the uncertainty based on the MC-dropout method, we adapted the model by removing the output layer, and instead added an output layer with softmax activation and neurons equal to the number of classes of the respective dataset preceded by a dropout layer, with dropout rate of 50%. For simplicity, the resulting architecture preserve the same name as the original (ResNet50).

As for training, we use the categorical cross-entropy loss and the Adam optimizer to train all models with initial learning rate of 0.0002, decay of 0.2 every 8 epochs and patience of 10 epochs. In the CCDA case, the training was during 50 epochs with a batch-size of 16. In the UAGAN and UDA case, the training was during 40 epochs with a batch-size of 32. Keras with TensorFlow as backend was used as our deep learning training framework.

Additional specific setups considered in the experiments to validate each method are discussed below.

Specific setups for CCDA

Eight models were trained for each dataset based on the same architecture, but changing the data augmentation applied. The models considers for benchmark purpose are the following:

1. *ResNet50*: CNN Model trained without applying data augmentation.

2. *ResNet50+DA*: CNN model with the traditional randoms crops and horizontal flips strategies to data augmentation.
3. *ResNet50+DA+B*: The base is ResNet50+DA with the incorporation of a random adjustment of the image brightness in the data augmentation process.
4. *ResNet50+DA+CN*: The base is ResNet50+DA with the incorporation of a random adjustment of the image contrast in the data augmentation process.
5. *ResNet50+DA+CL*: The base is ResNet50+DA with the incorporation of a random adjustment of the image color in the data augmentation process.
6. *ResNet50+DA+S*: The base is ResNet50+DA with the incorporation of a random adjustment of the image sharpness in the data augmentation process.
7. *ResNet50+DA+A*: The base is ResNet50+DA with the incorporation of a random adjustment of the all image enhancement strategies (brightness, contrast, color, and sharpness) to the images in the data augmentation process.
8. *CCDA*: The base is ResNet50+DA with the incorporation of a random adjustment of image enhancement strategies in the data augmentation process, which are selected according to the image class. Note that, in some cases, all or none of the strategies could be applied.

Regarding the data augmentation process, for models 2 to 8, the original image is resized to (256,256) and then randoms crops with a size of (224,224) and horizontal flip with a 50% probability are applied. In the case of models 3 to 8, in addition, we apply image enhancement methods available from the image processing library PIL. With the aim that the application of specific techniques of data augmentation do not drastically affect the information contained in an image, we empirically select a range of values between 0.75 and 1.25 for all methods, where the value applied in the images will be obtained randomly in each iteration. Notice that a value equal to 1 means obtaining the same image. On the other hand, a value equal to 0 is obtained for the brightness, the color, the contrast or the sharpness, that is to obtain a black, black and white, solid gray or blurred image, respectively. Therefore, a factor less than 1 means, for example, less color and vice versa.

Examples of the resulting images with the minimal and maximal value can be seen in the Fig. 5.4.

Specific setups for UAGAN

Three different training strategies of the same model are used for a benchmark purpose:

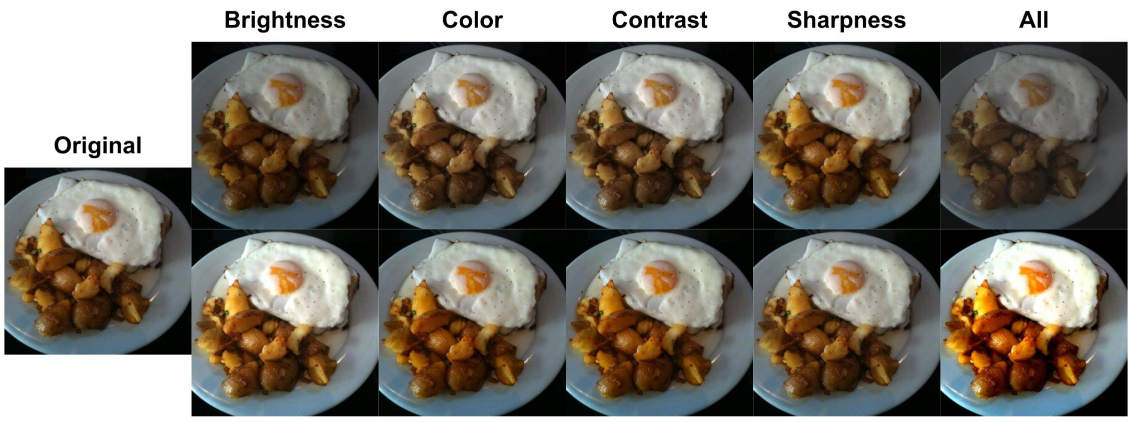


FIGURE 5.4: An example of the resulting image when we apply the minimal (top) and maximal (bottom) values used in the experimentation for each strategy corresponding to the image enhancement methods.

- ResNet50: baseline model training with the original images without data augmentation.
- ResNet50+SDA: baseline model with standard data augmentation applying during the training, like random crops and horizontal flips.
- UAGAN: ResNet50+SDA using the real and synthetic images.

With respect to the image generation, we use the default parameters proposed by the authors of SINGAN.

Specific setups for UDA

For comparative purposes, four different training strategies in ResNet50 were conducted:

- S_1: training with real images without applying any type of data augmentation,
- S_2: training with standard online data augmentation, such as random crops and horizontal flips, applied on real images only,
- S_3: training with standard online data augmentation, applied on a dataset consisting of both synthetic images (one for each real image) and real images,
- S_4: training with standard online data augmentation, applied on a dataset generated by our UDA method.

Regarding the parameters used in the proposed approach, we set the UDA thresholds to $T_l = 10\% \times U_{max}$ and $T_l = 90\% \times U_{max}$, where U_{max} denotes the maximum entropy, which is given by the equation $U_{max} = Log(|C|)$. Furthermore, we set to three the maximum training cycles.

For all the strategies, we trained the same model 5 times, with random initialization of the weights and shuffling the images. In the proposed method case, for each training cycle we select the model that provides the median result, considering the metric R_{micro}, to discover the most uncertain images and generate new ones.

With respect to the generation of synthetic images, we simultaneously apply various transformations on the real images, such as rotation, width and height shift, shear, zoom, etc. Fig. 5.5 shows an example of the generated images.

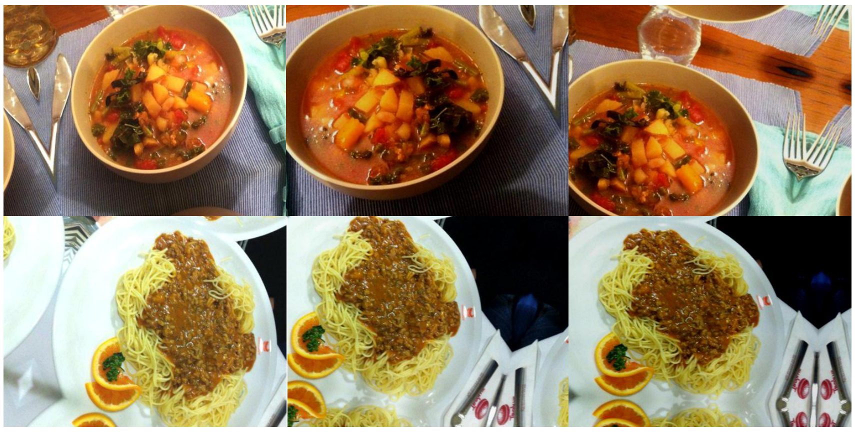


FIGURE 5.5: Sample of the synthetic images from the generator applied.

5.4.4 CCDA Results

In this section, we present the results obtained by the models in the datasets Food-101 and UECFOOD-256.

As we explained in Sec. 5.3, to train our proposed approach, we must first select the specific data augmentation techniques that will be applied to each image taking into account the class to which the image belongs. For this purpose, we trained five models, one used as reference (ResNet50+DA) and other four models which incorporate one SDA technique each one (ResNet50+DA+B, ResNet50+DA+CN, ResNet50+CL and ResNet50+S). Once the models have been trained, we calculate the epistemic uncertainty of the samples contained in the validation sets. Then, we average the epistemic uncertainty of the samples within the same class (CEU). After that, we compared the CEU of the ResNet50+DA model with respect to the rest of the models for each class. If at least one of models have a CEU smaller than the reference model, we consider that the enhanced method incorporated in these models is likely to benefit the learning of this class. The results obtained by applying this procedure can be seen in Fig. 5.6. In this figure, we show in red color the SDA techniques selected for each class in the UECFOOD-256 datasets. For example: in case that images belong to the $c_{id} = 0$, we obtain that we must to adjust randomly the

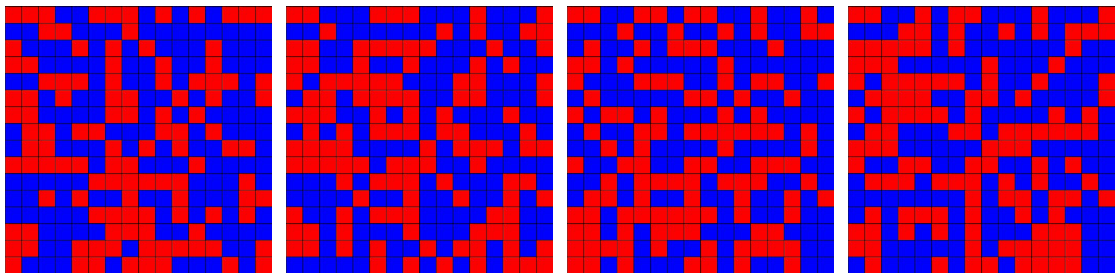


FIGURE 5.6: From left to right: classes selected of UECFOOD-256 to apply adjustment of: a) brightness, b) color, c) contrast and d) sharpness. The class identifier (c_{id}) can be obtained for the formula: $c_{id} = 16 * col + row$, where the first position (upper-left corner) corresponds to $(row, col) = (0, 0)$.

brightness, the color, the contrast and the sharpness; for the images belongs to $c_{id} = 15$, brightness and contrast; and so on.

Regarding the results during the test phase, these can be seen, in terms of Acc (with dropout turned down) and NEU (with dropout turned up), in Table 5.1 and Table 5.2 for the Food-101 and UECFOOD-256 datasets respectively. As one might expect, the lowest performance in terms of Acc was achieved in both datasets, when we trained the model without data augmentation (ResNet50). Despite this, it is interesting to highlight that this model has the best result in terms of NEU. Note that NEU is calculated from the predictive entropy, so, in essence, a high value also implies a high entropy and, consequently, the model is more prone to change the prediction when there are perturbations in the data that produces a loss of some discriminative feature. For this reason, a model with low NEU means that the model remains stable against the perturbation, keeping the good and bad predictions. On the other hand, when analyze the results obtained with ResNet50+DA+A, we can see that the incorporation of all SDA techniques into the data augmentation process negatively affects the learning of our model, reducing the correct prediction rate and model stability. Finally, when we compare our proposed method with the rest, we can see that a correct trade-off between both measure was achieved, the Acc is improved, and the NEU is the lowest regarding to the models trained with data augmentation.

5.4.5 UAGAN Results

In this section, we present the results obtained by the proposed method. The first step of our method corresponds to selecting those images difficult to classify (with high uncertainty). After training the model with the original images, we determine the EU and build a histogram for the training images (see Fig. 5.7). The right side of the histogram

TABLE 5.1: Results on Food-101 in terms of Acc and NEU for the models trained with different data augmentation techniques. For NEU, less is better.

Model	*Acc*	*NEU*
ResNet50	77.66%	19.85%
ResNet50+DA	82.65%	27.35%
ResNet50+DA+A	82.54%	29.45%
Proposed method	82.82%	26.25%

TABLE 5.2: Results on UECFOOD-256 in terms of Acc and NEU for the models trained with different data augmentation techniques. For NEU, less is better.

Model	*Acc*	*NEU*
ResNet50	61.00%	30.22%
ResNet50+DA	65.02%	33.55%
ResNet50+DA+A	64.65%	36.53%
Proposed method	65.54%	33,51%

corresponds to all the images considered to generate the new ones. The criterion applied corresponds to selecting all images with EU equal to or greater than the uncertainty calculated by the average between the maximum and minimum uncertainty predicted for all images. A total of 120 images was selected.

In Fig. 5.8, we can see the distribution of the training images along each dish, the average

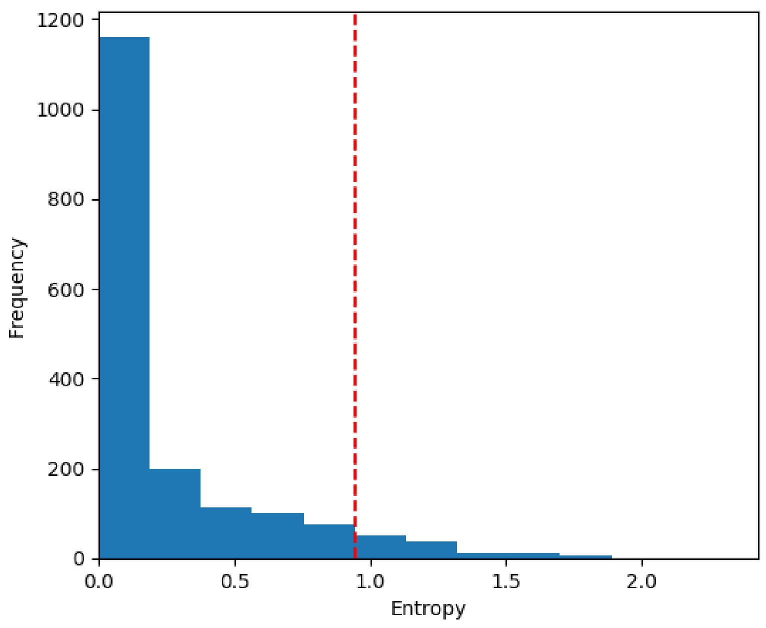


FIGURE 5.7: Histogram for the entropy of the predicted images.

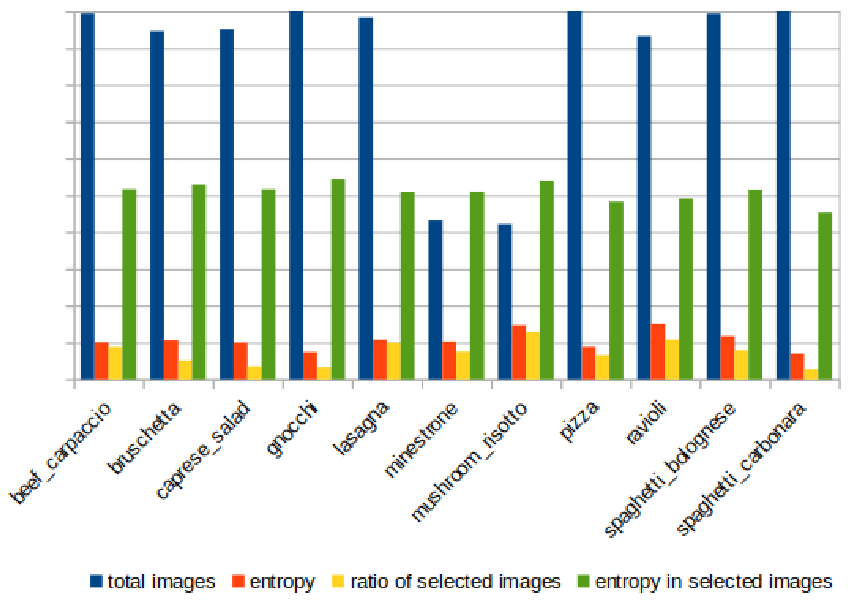


FIGURE 5.8: Training images vs epistemic uncertainty.

TABLE 5.3: Results obtained on the test set in term of accuracy with the standard deviation.

Method	Acc	Std
ResNet50	79.15%	0,60%
ResNet50 + SDA	81.00%	0,78%
UAGAN (our proposal)	82.32%	0,96%

entropy in all the images, the proportion of the selected images and the average entropy for the selected images. Unlike the evidence shown in [84] for CIFAR-10, for this type of data, the frequency of the images is not a factor that determines a high or low uncertainty for a specific class. In our case, we believe that uncertainty occurs due to the great variability of visual appearance that may be present in the images belong to the same class of dish, where the factor to consider is the diversity of the collected sample and not only the size of the sample. To fill the gap of poorly represented sample for a class, we duplicate the presence of images with high uncertainty through a generation of synthetic images with the SINGAN method. In Fig. 5.9, some examples of the generated images are shown.

With a total of 1889 training images, 1679 originals and 120 synthetic ones, we train the final model. The results obtained by three different training strategies of the same model are shown in the Table 5.3. All models were fine-tuned from ImageNet [91] and retrained the whole network using the target training set. For each strategy, 5 models were trained with random initialization values and random order of images. Then, we calculated the average accuracy and the standard deviation achieved for the best model obtained

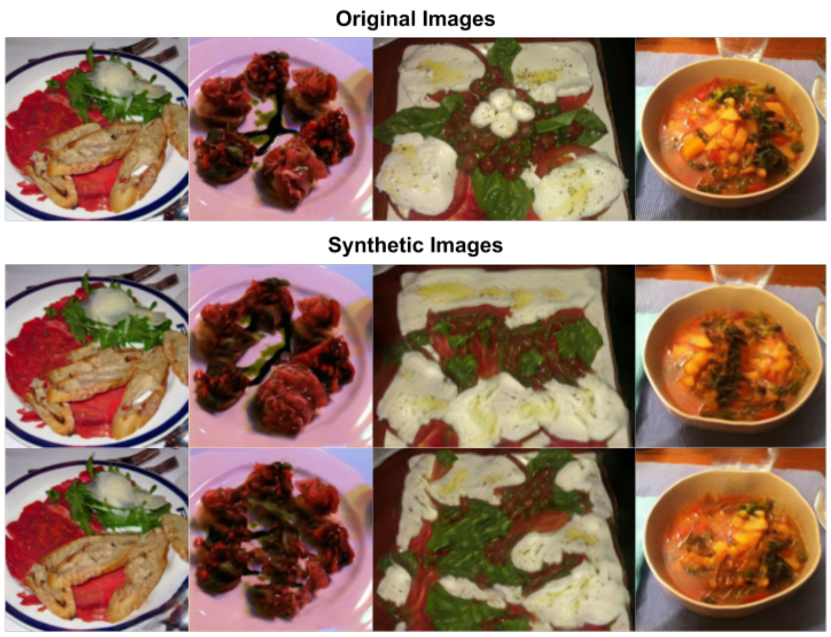


FIGURE 5.9: Synthetic image generated on the selected images from the training set.

TABLE 5.4: Synthetic images added after each training cycle. We mark with * the last images added for each selected model.

Dataset	$Cycle_0$	$Cycle_1$	$Cycle_2$
American	504	410	434*
Chinese	326*	274	258
French	362	388	283*
Greek	266	260	224*
Indian	279	268	253*
Italian	554	438	439*
Japanese	388	381	303*
Mexican	642*	458	518
Thai	331	312*	292
Turkish	455	397*	354
Vietnamese	291	382*	241

on each iteration according to the performance on the validation set. For the results achieved, we can see that UAGAN improved the performance in terms of accuracy with respect to the rest of strategies. Specifically, the improvement is 3,17% on ResNet50 and 1,32% on ResNet50+SDA.

5.4.6 UDA Results

In this section, we present and analyze the results of the different strategies evaluated on the ResNet50 model, including the proposed UDA method.

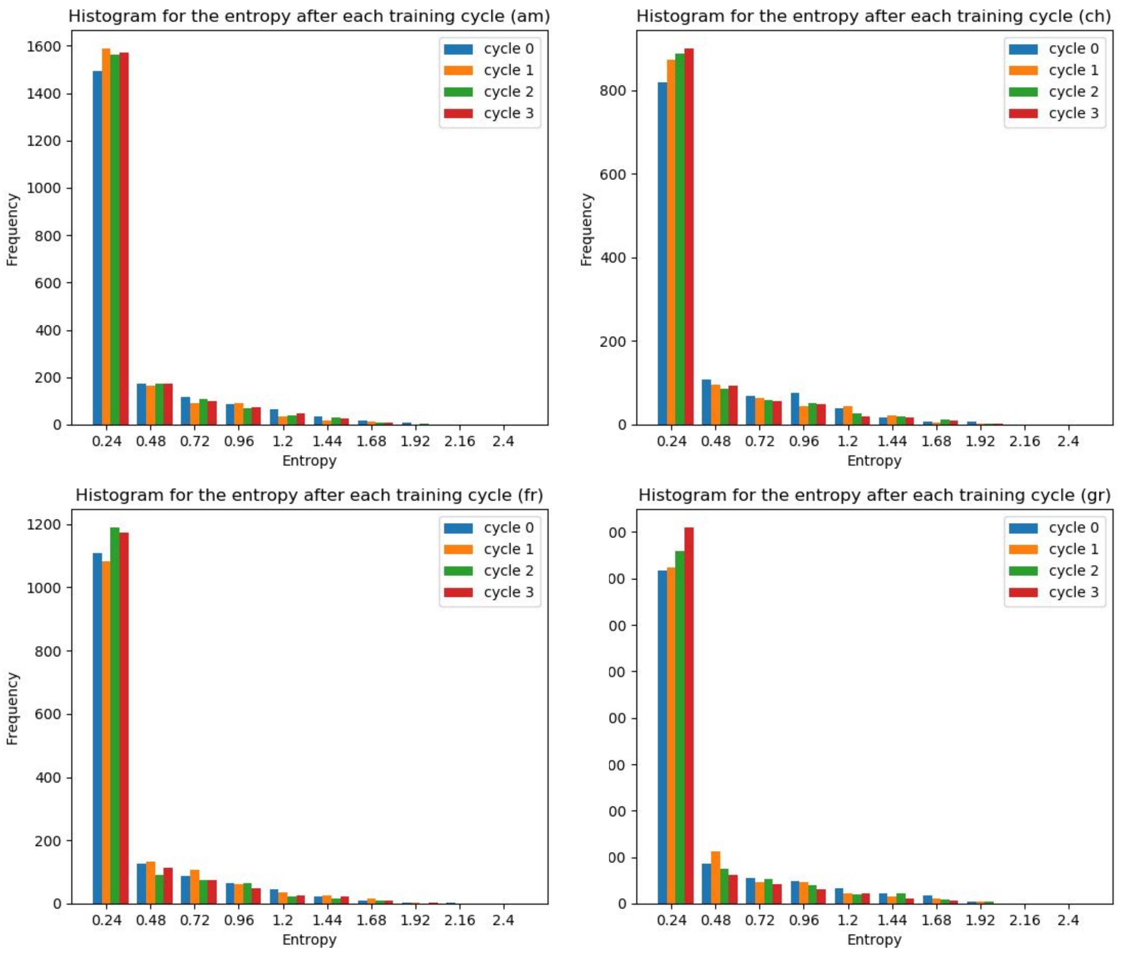


FIGURE 5.10: Histogram of the entropy after each training cycle when training with UDA (S_4).

Like in the active learning frameworks, the proposed UAD method follows training cycles and augments training data before starting each new cycle. Consider that for the initial cycle, we use the same ResNet50 model trained with $S2$ strategy. Then, to illustrate how the EU changes after each training cycle, we plotted a histogram of 10 bins for four of the eleven sub-datasets used for evaluation (see Fig. 5.10). As expected, when new data is incorporated into the training set, the EU tends to decrease. However, it is interesting to note that with our UDA approach, higher reductions are achieved by only adding less than 30% of synthetic samples (see Table 5.4) to the training data (orange bar in Fig. 5.10). Furthermore, we can see that after each cycle, more data is less uncertain. However, the improvement in the first two cycles is markedly greater than the last two. This behaviour can be explained by the need, in some very hard samples, of a generator that provides more diverse transformations. As we can see in Fig. 5.11, there are some images for which, despite having added synthetic samples related to them during three

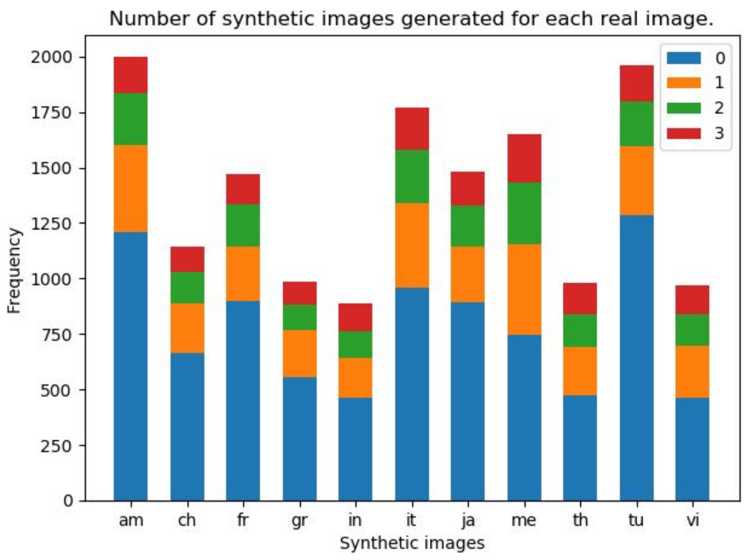


FIGURE 5.11: Number of synthetic images generated after the third training cycle.

TABLE 5.5: Results obtained on the test sets in terms of R_{micro}.

Dataset	**ResNet50 (S_1)**	**ResNet50 (S_2)**	**ResNet50 (S_3)**	**ResNet50 (S_4)**
American	82,07%	83,84%	84,09%	**84,34%**
Chinese	89,49%	89,49%	91,44%	**91,83%**
French	89,49%	91,53%	**92,88%**	92,20%
Greek	90,05%	90,05%	91,10%	**92,15%**
Indian	87,37%	92,63%	**93,16%**	92,11%
Italian	79,15%	81,75%	82,70%	**83,65%**
Japanese	89,36%	81,84%	**92,20%**	91,49%
Mexican	79,23%	79,87%	**82,11%**	81,15%
Thai	76,88%	**82,26%**	**82,26%**	81,18%
Turkish	91,37%	91,59%	91,81%	**92,04%**
Vietnamese	85,49%	87,59%	88,60%	**90,16%**

cycles, the model can not yet learn their discriminative features (shown in red). On the other hand, we can see in the same figure that, after three cycles, our approach selects less than 50% of the initial hard samples (blue vs red bars). This means that the model successfully learns most of the samples that were considered hard in the initial cycle.

In Table 5.5, Table 5.6 and Table 5.7 we present the results obtained, for the four training strategies in eleven subsets consisting of dishes belonging to each cuisine type from MAFood-121, in terms of R_{micro}, R_{macro} and $|R_{micro} - R_{macro}|$. From these tables, we can see that the UDA method (S_4) provides better results in more datasets than the remaining strategies, both when we evaluate the performance at sample level (R_{micro}) and at class

TABLE 5.6: Results obtained on the test sets in terms of R_{macro}.

Dataset	**ResNet50 (S_1)**	**ResNet50 (S_2)**	**ResNet50 (S_3)**	**ResNet50 (S_4)**
American	81,99%	83,69%	84,10%	**84,26%**
Chinese	87,93%	90,05%	90,60%	**91,17%**
French	89,01%	90,33%	**94,12%**	92,54%
Greek	89,12%	89,34%	89,90%	**92,11%**
Indian	87,67%	92,96%	**93,29%**	92,41%
Italian	80,72%	82,44%	**84,31%**	84,07%
Japanese	88,08%	90,85%	**91,20%**	90,93%
Mexican	79,12%	80,37%	81,64%	**81,96%**
Thai	70,98%	**79,91%**	79,85%	79,22%
Turkish	91,44%	91,65%	91,92%	**92,15%**
Vietnamese	84,67%	86,99%	88,14%	**89,85%**

TABLE 5.7: Results obtained on the test sets in terms of $|R_{micro} - R_{macro}|$.

Dataset	**ResNet50 (S_1)**	**ResNet50 (S_2)**	**ResNet50 (S_3)**	**ResNet50 (S_4)**
American	0.08%	0.15%	**0.01%**	0.08%
Chinese	1.56%	**0.55%**	0.84%	0.66%
French	0.48%	1.19%	1.24%	**0.34%**
Greek	0.93%	0.71%	1.20%	**0.04%**
Indian	0.3%	0.33%	**0.13%**	0.31%
Italian	1.57%	0.68%	1.61%	**0.42%**
Japanese	1.28%	0.99%	1.00%	**0.56%**
Mexican	**0.11%**	0.50%	0.47%	0.81%
Thai	5.9%	2.35%	2.41%	**1,97%**
Turkish	0.07%	**0.05%**	0.10%	0.11%
Vietnamese	0.82%	0.57%	0.47%	**0.31%**

level (R_{macro}). Moreover, it is the strategy that provides less bias in the classification with respect to the classes with more or less images (see Table 5.7). Therefore, focusing on hard samples not only provides an efficient way to augment the data by reducing the required images (e.g. S_4 use much less than S_3), but also contributes to achieving the right balance during classification.

Note that although there is some generic resemblance to active learning, an explicit comparison with such approaches is out of scope. Our goal is to demonstrate that focusing on hard samples to generate the synthetic new ones can help the model learn about the features it represents. Then, the objective is how to design the generation of new data rather than deciding the order or selection of samples to feed the classifier model.

5.5 Conclusions

In this chapter, we presented three methods to perform data augmentation based on the analysis of the epistemic uncertainty: CCDA, UAGAN and UDA.

In CCDA case, we differentiate the traditional data augmentation in two type, where one of them depends on the class of the image. For the latter, the techniques to be used are selected by the analysis of the epistemic uncertainty for each class of the same model trained with different data augmentation strategies. Note that the selection process can be considered computationally expensive. However, the cost involved in this process is only present in the training time. Therefore, it does not harm the model performance during the prediction regarding the use of resources computational or response time. From the results obtained, we observed that the proposed approach outperforms the results with respect to the rest of the model evaluated without affecting its stability. Summarizing, we have shown that it is beneficial for model learning to incorporate specific data augmentation techniques taken into account the class to which the image belongs.

As for UAGAN, correspond to a novel method for sample-level uncertainty-aware data augmentation composed of three phases: 1) identification of hard samples, by means of analysis of the epistemic uncertainty; 2) generating new data from identified samples; and 3) performing the final training with the original and synthetic images. We demonstrated the effectiveness of the approach proposed on the Italian dishes from MAFood121 public dataset. The result obtained shows that our proposal outperforms the classification by incorporating only 120 synthetic images based on the uncertainty analysis (5% of the total).

Finally, UDA is a novel method for uncertainty-aware data augmentation that follows an active learning framework and takes into account the most uncertain images in order to generate new ones. We validated our approach on eleven subsets of data from the public food dataset MAFood-121, where each one consists of dishes that belong to the same type of cuisine. The results obtained show that, in many cases, it is not necessary to generate data-augmented images for all the samples, but it is possible to improve the performance only by focusing on those that present a greater uncertainty. Furthermore, our method contributes to getting more balanced classification in the unbalanced dataset.

As future work, we will explore both sample-level and class-level uncertainty together to increase deep learning on datasets through an active learning framework, explore the application of epistemic uncertainty in outlier detection and dive deeper into the data generation process, based on uncertain samples, incorporating them into GAN-based methods.

Chapter 6

Exploring Food Detection using CNNs

6.1 Introduction

In the last decades, the amount of people with overweight and obesity is progressively increasing [125], whom generally maintain an excessive unhealthy diet consumption. Additionally to the physical and psychological consequences involved to their condition, these people are more prone to acquire chronic diseases such as heart diseases, respiratory diseases, and cancer [127]. Consequently, it is highly necessary to build tools that offer high accuracy in nutritional information estimation from ingested foods, and thus, improve the control of food consumption and treat people with nutritional problems.

Recently, the computer vision community has focused its efforts on several areas devoted to developing automated systems for visual food analysis, which usually involve using a food detection method [76, 21, 114, 153]. These methods, also called food/non-food classification, have as purpose to determine the presence or absence of food in an image. Generally, they are applied as a pre-processing prior to food analysis, and can also be useful for selecting food images from huge datasets acquired from the WEB or from wearable devices.

Food detection has been investigated in the literature in different works [76, 153, 87, 48, 130], where it has been proven that the best results obtained are based on Convolutional Neural Networks. The first method based using this technique was proposed by [76], which achieved a 93.8% using AlexNet model [91] on a dataset composed of 1,234 food images and 1,980 non-food images acquired from social media sources. They proved that using a CNN provided a 4% higher accuracy compared to using hand crafted features [87]. In [75], the authors improved the accuracy on this dataset to 99.1% using the NIN model [103]. In addition, they evaluated their model on other datasets, IFD and FCD, obtaining 95% and 96% of accuracy, respectively. Evaluation on a huge dataset with

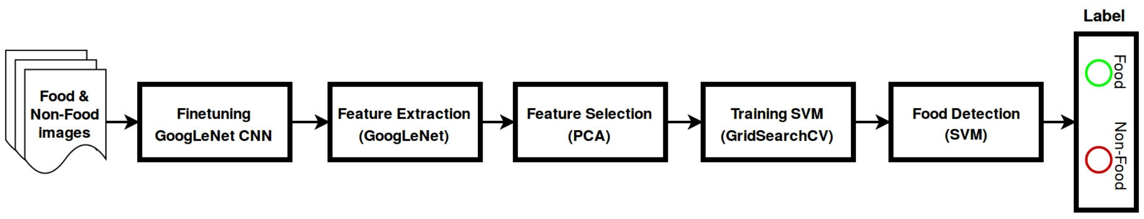


FIGURE 6.1: Method overview for our food detection approach.

over 200,000 images constructed from Food101 [23] and ImageNet Challenge was done in [114], where the authors achieved 99.02% using an efficient CNN model based on inception module called GoogLeNet [159]. The same model was used in [21], the authors obtained 95.64% of accuracy on a dataset composed of Food101; food-related images extracted from the ImageNet Challenge dataset; and Pascal [44] (used as non-food images). Evaluation of different CNN models and settings was proposed by [130] on a dataset that we call RagusaDS. The authors obtained the best results using AlexNet for feature extraction and Binary Support Vector Machine (SVM) [34] for classification. In terms of accuracy, they achieved 94.86%. In [153], the authors apply fine-tuning on the last six layers of a GoogLeNet obtaining high accuracy, but tested their model on a balanced dataset of only 5,000 images (Food-5k). Since the proposed models were evaluated on different datasets, the results obtained are not directly comparable. Therefore, in order to compare our results with the state-of-the-art, we selected the available datasets with more than 15,000 images.

Furthermore, we explored the food detection problem using the GoogLeNet, because this CNN model presented the best results in the classification of objects in the ILSVRC challenge [141]. In particular for food detection it has also presented good results on multiplies datasets with images acquired in different conditions [21, 114, 153]. Specifically, we propose a food detection model combining GoogLeNet for feature extraction, PCA [72] for feature selection and SVM for classification, which prove the best accuracy in the state-of-the-art with respect to the previous works on the same datasets.

This chapter is organized as follows: in section 6.2, we present our methodology. In section 6.3, we present and discuss the datasets used and the results obtained. Finally, in section 6.4, we present conclusions and future work.

6.2 Methodology for food detection

We propose a methodology for food detection, which involves the use of the GoogLeNet model for feature extraction, PCA for feature selection and SVM for classification. In Fig.6.1, we show the pipeline of our food detection approach which will be explained below.

6.2.1 GoogLeNet for feature extraction

The first step in our methodology consists in training the GoogLeNet CNN model. For this purpose, we pre-train the GoogLeNet model on ImageNet [141], as a base model, and then we change the number of classes in the output layer for our binary classification problem (food/non-food). Then, GoogLeNet is fine-tuned on the last two layers until the accuracy on training set stops to increase, then we choose the model that gives the best accuracy on the validation set.

Once GoogLeNet is fine-tuned, we use the resulting model as a feature extractor. The feature vector for each image is extracted using the penultimate layer, with which a 1024-dimensional vector is obtained for each image. Then, we calculate a transformation that distributes normally the data through a Gaussian distribution function with zero mean and unit variance, by means of the feature vectors obtained from the training set. Finally, we normalize the data, in a range of $[-1, 1]$, by applying this transformation to each extracted feature vector.

6.2.2 PCA for feature selection

The following step in our methodology consists in reducing the dimensions of the feature vectors obtained in the previous steps by means of Principal Component Analysis (PCA) [72], which transforms the data to a new coordinate system leaving the greatest variance of the images in the first axes (principal components). We apply PCA on all feature vectors normalized from the training set and then the principal components are analyzed to select the first dimensions that retain the most discriminant information. To do this, we selected the features based on the Kaiser Criterion [77], which consists of retaining those components with eigenvalues greater than 1. The feature vectors reduced are used during the training of SVM and also during classification.

6.2.3 SVM for classification

An SVM is a classification algorithm that optimizes a boundary $f(x) = Wx + b$ for maximizing the margin between two types of data, where the maximum margin is found solving a quadratic optimization problem. The dual SVM formulation is defined as follows:

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$s.t. \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, n, \quad \sum_{i=1}^{n} \alpha_i y i = 0.$$

where x_i are the reduced feature vectors calculated from the n training set images, y_i the images class identified by the label -1 or 1, α_i are the Lagrange multipliers, and $K(x_i, x_j) = \tanh(\gamma x_i^T x_j)$ is the chosen kernel function, as in [130], for SVM classifier. The parameters C and γ are obtained by means of the GridSearchCV strategy, and the parameters α_i are adjusted during the optimization.

With the solution of the optimization problem, we compute the parameters of the objetive function f(x), where $W = \sum_{i=1}^{n} \alpha_i y_i x_i$, and b is the bias. Finally, the class is predicted using the $sgn(f(x))$ function, which returns +1 when $f(x)$ is positive and -1 when $f(x)$ is negative.

6.3 Experiments

6.3.1 Datasets

In this section, we present the selected datasets for the evaluation of the proposed model and comparison of the results. Both datasets, FCD and RagusaDS, were selected because they contain a significant amount of images, at least 15.000, and also they have free access to the images.

FCD was constructed from two public datasets widely used: Food-101 [23] and Caltech-256 [55] for food and non-food images, respectively (see Fig.6.2). Food-101 is a dataset for food recognition, which contains 101 international food categories with 1.000 images each one. Caltech256 contains 256 categories of objects with a total of 30.607 images, in which each object has a minimum of 80 images. For the construction of FCD, not all images of these datasets were considered. To balance the amount of food and non-food, we selected 250 images for each category in the Food101. The selection was based on the color histogram of the images, keeping those with the highest color variance within the same category and thus, keeping the most highly variable set, obtaining a total of 25.250 food images. In Caltech 256, all images were selected except the food-related ones, resulting in 28.211 non-food images. To evaluate our approach, we used 64% of the images for training, 16% validation and 20% test.

RagusaDS consists of three datasets acquired in different conditions: UNICT-FD889 [47] and Flickr-Food [48] for positive; and Flickr-NonFood [48] for negative samples. UNICT-FD889 is a dataset composed of 3.583 images of meals of 889 different dishes acquired from multiple perspectives with the same device in real-world scenarios, where images were acquired from a top view avoiding the presence of other objects. The Flickr images datasets were manually labeled as being food or non-food images. These datasets, which are called Flickr-Food and Flickr-NonFood, contain 4.805 images of food and 8.005 of non-food, respectively. Compared to UNICT-FD889, they contain less restricted images, and specifically for Flickr-Food the images can contain additional objects as well as food

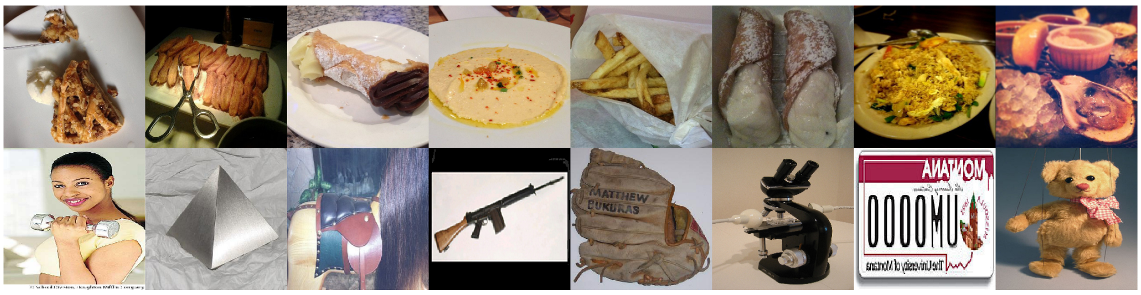


FIGURE 6.2: Example of images contained in the FCD Dataset. Top row shows food images from Food101 and bottom row shows non-food images from Caltech256.

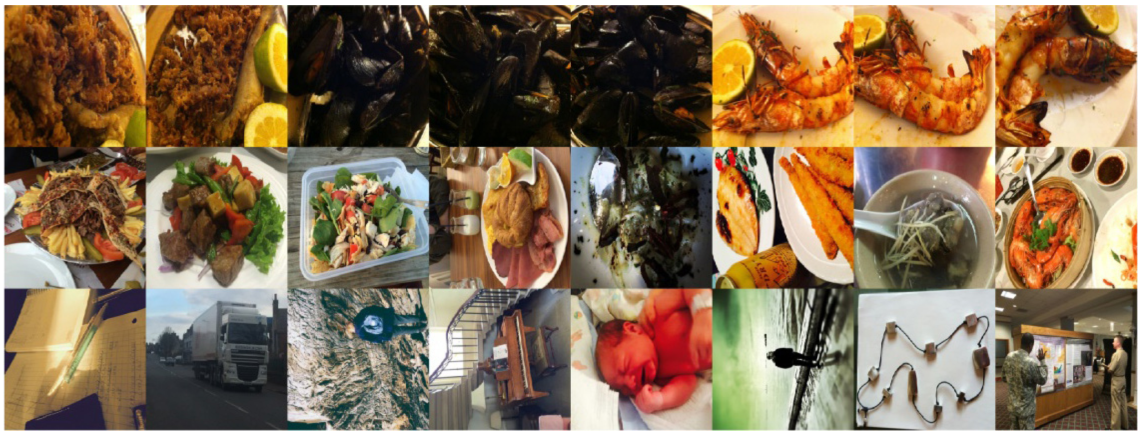


FIGURE 6.3: Example of images contained in the RagusaDS Dataset. Top row shows food images from UNICT-FD889, middle row shows food images from Flickr-Food, and bottom row shows non-food images from Flickr-NonFood.

and were taken from different points of view. In total, the dataset contains 8.388 images of food and 8,005 of non-food (see Fig.6.3). From the UNICT-FD889 dataset we split 80% of the data for training and 20% for validation. The first 3.583 images of Flickr-NonFood were also used for validation, and the remaining 4,422 as well as all images from Flickr-Food were used for testing.

6.3.2 Experimental Setup

We used Caffe [70] for training our CNN model. We fine-tuned the last two layers of our model applying ten times the default learning rate. We set a learning rate of 1×10^3, with a decay of 0.96 every 5.000 iterations and a batch size of 32. We pre-processed the images by resizing them to 256×256 pixels, subtracted the training average of ImageNet and maintained the original color pixels scale. During training, data augmentation was applied by using horizontal mirroring and random crops of 224×224 pixels. During prediction, a center crop is applied.

TABLE 6.1: Results obtained by models based on CNN on RagusaDS and FCD datasets on the food detection task. All results are reported in %.

	RagusaDS			FCD		
	Acc	**TPr**	**TNr**	**Acc**	**TPr**	**TNr**
AlexNet + SVM [130]	94.86	**94.28**	95.50	-	-	-
NIN [75]	-	-	-	96.40	96.00	97.00
GoogLeNet	94.66	91.53	98.06	98.87	98.48	**99.22**
GoogLeNet + SVM	94.95	91.53	**98.67**	98.96	**98.85**	99.06
GoogLeNet + PCA-SVM	**94.97**	91.57	**98.67**	**99.01**	**98.85**	99.15

TABLE 6.2: Results obtained when GoogLeNet + PCA-SVM is trained on both datasets together (RagusaDS+FCD) and evaluated separately and jointly.

Test dataset	**Acc**	**TPr**	**TNr**
RagusaDS	95.78%	93.65%	98.10%
FCD	98.81%	98.60%	99.01%
RagusaDS+FCD	97.41%	96.19%	98.61%

The GoogLeNet was fine-tuned during 10 epochs, in the case of FCD, and during 40 epochs for RagusaDS. Training of the models was stopped when accuracy converged in the training set. The feature vector extracted from each image is reduced selecting the principal components based on the Kaiser Criterion, resulting in 186 dimensions on RagusaDS and 206 dimensions on FCD. As for the optimization of C and γ parameters, we applied a 3-fold cross validation. We defined a range of 14 values uniformly distributed on a base-10 logarithmic scale. In the case of the C parameter, we used a range from 1×10^{-4} to 1×10^{2} and for γ parameter from 1×10^{-8} to 1×10^{-2}. Finally, the best parameters are used to train the SVM from scratch with all the training set.

6.3.3 Metrics

We used different metrics to evaluate the performance of our approach, namely: overall Accuracy (Acc), True Positive rate (TPr) and True Negative rate (TNr), which are defined as follows: $Acc = \frac{TP+TN}{T}$, where TP (True Positive) and TN (True Negative) are the amount of correctly classified images as Food and Non-Food, respectively; $TPr = \frac{TP}{TP+FN}$, where FN (False Negative) is the amount of misclassified images as Non-Food; $TNr = \frac{TN}{FP+TN}$, where FP (False Positive) is the amount of images misclassified as food.

6.3.4 Results

In this section, we present the results obtained during the experiments. In Table 6.1, the first two rows correspond to the state-of-the-art algorithms that gave the best prediction on RagusaDS and FCD datasets, respectively. The last three methods are variations of

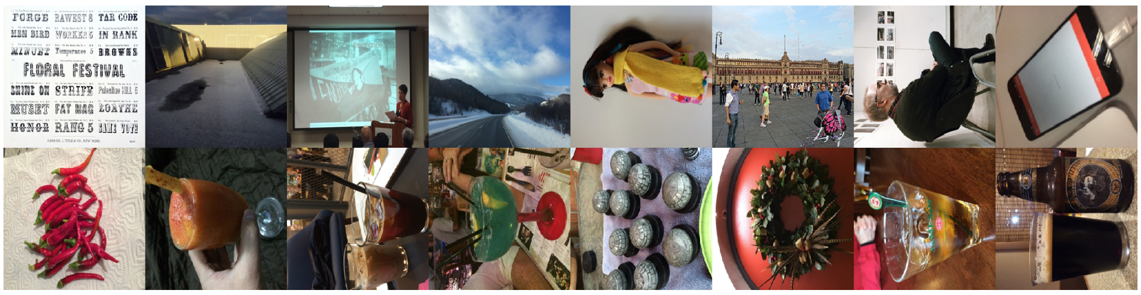


FIGURE 6.4: FP (top) and FN (bottom) on RagusaDS dataset.

our proposal, which is based on the GoogLeNet. The results show the Acc, the TPr and TNr obtained when evaluating each method on the FCD and RagusaDS datasets. In the case of the FCD, it can be seen that the model obtains a high precision in the global classification and maintains a slightly higher performance on TNr, which may be due to the small imbalance between food and non-food images of this dataset. On the other hand, for RagusaDS the difference between TPr and TNr is about 7% better for TNr. We believe that this occurs considering that food images used during training are very different from those used for evaluation and therefore the model is not able to recover enough discriminant information that allows to generalize over a sample acquired under different conditions. GoogLeNet + PCA-SVM is selected for the next experiment given that it achieved the best results on both datasets.

Following, we trained the best model and evaluated its performance using RagusaDS and FCD datasets together, maintaining the same sets of training, validation and test, which we named RagusaDS+FCD. Table 6.2 shows the results obtained by training our approach using the training sets from RagusaDS+FCD and evaluating on the test sets from RagusaDS+FCD, RagusaDS and FCD. The results show that, when the model is trained on RagusaDS+FCD, it improves the classification significantly on RagusaDS although it presents a slight decrease on FCD. We believe that the improvement on RagusaDS is mainly due to an increase in the detection of food-related images. We deduce that by combining the training datasets, our method is able to extract features from various types of food acquired in different conditions, which allows to have a more robust classifier achieving a better generalization on the test set of RagusaDS dataset.

Some FPs FNs obtained in both datasets are shown in Fig. 6.4 and Fig. 6.5. Analyzing the FNs, we can observe that in the case of RagusaDS most errors occurred in images in which food was a liquid (drink, coffee, etc). The reason for this is because the training set contains a wide variety of dishes but none of these correspond to beverages and therefore the classifier does not recognize them as food. In addition, other factors that influence classification are poorly labeled images such as food and also the cases where in the same image there are a lot of dishes. In the case of FCD, there are also some errors caused by wrong labels in both categories.

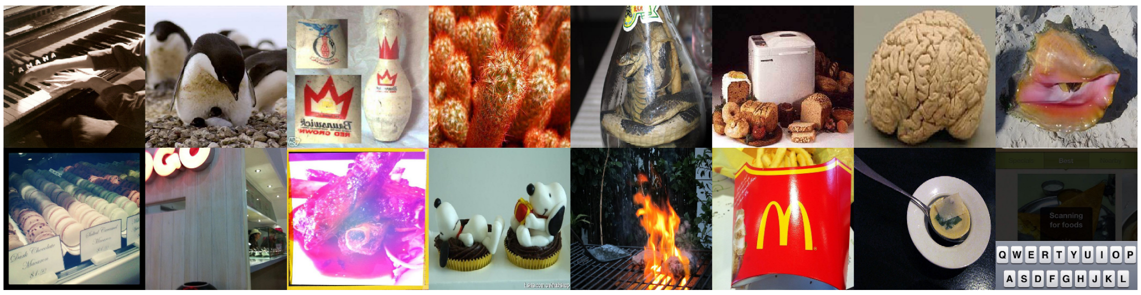


FIGURE 6.5: FP (top) and FN (bottom) on FCD dataset.

6.4 Conclusions

In this chapter, we addressed the food detection problem and proposed a model that uses GoogLeNet for feature extraction, PCA for feature selection and SVM for classification. Furthermore, we applied a benchmark on the two more widely used publicly available datasets. From the results obtained, we observed that the best accuracy is achieved in both datasets with our proposed approach. Specifically, the improvement in the overall accuracy is more than 2% on FCD and about 1% for RagusaDS, when both datasets are combined for training and evaluated on the respective datasets. In addition, the overall accuracy when combining both datasets is 97.41%. As a conclusion, we explored the problem of food detection comparing the last works in the literature and our proposed approach provides an improvement on the state-of-the-art with respect to both public datasets. Moreover, models based on GoogLeNet, independently of the settings, gave the highest accuracy on the food detection problem. As future work, it would be interesting to evaluate the performance of CNN-based models on larger datasets containing a much wider range of dishes and beverages such as food images and diversity of environments for non-food images. In addition, considering the large number of possible images of food and non-food, it is very likely that the model will behave unexpectedly in some images with visual content never seen before. To provide a more robust food detection method, we can model uncertainty to make a better decision on uncertain predictions.

Chapter 7

Food Recognition using Fusion of Classifiers based on CNNs

7.1 Introduction

As we previously commented, food recognition has caused a lot of interest for researchers considering its applicability in solutions that improve people's nutrition and hence, their lifestyle [167]. In relation to the healthy diet, traditional strategies for analyzing food consumption are based on self-reporting and manual quantification [149]. Hence, the information used to be inaccurate and incomplete [140]. Having an automatic monitoring system and being able to control the food consumption is of vital importance, especially for the treatment of individuals who have eating disorders, want to improve their diet or reduce their weight.

Food recognition is a key element within a food consumption monitoring system. Originally, it has been approached by using traditional approaches [23, 104], which extracted ad-hoc image features by means of algorithms based mainly on color, texture and shape. More recently, other approaches focused on using Deep Learning techniques [104, 173, 111, 58]. In these works, feature extraction algorithms are not hand-crafted and additionally, the models automatically learn the best way to discriminate the different classes to be classified. As for the results obtained, there is a great difference (more than 30%) between the best method based on hand-crafted features compared to newer methods based on Deep Learning, where the best results have been obtained with Convolutional Neural Networks architectures that used inception modules [58] or residual networks [111].

Food recognition can be considered as a special case of object recognition, being a very active topic in computer vision lately. The specific part is that dish classes have a much higher inter-class similarity and intra-class variation than usual Imagenet objects (cars,

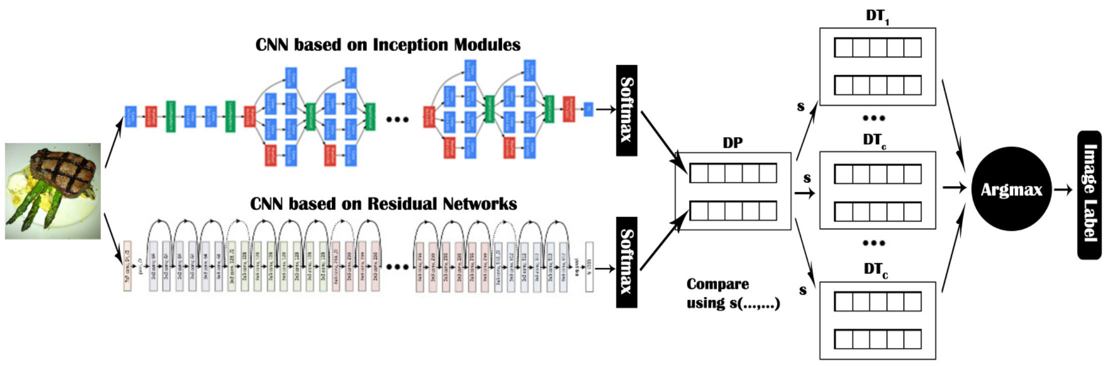


FIGURE 7.1: General scheme of our CNNs Fusion approach.

animals, rigid objects, etc.) (see Fig.7.3). If we analyze the last accuracy increase in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [141], it has been improved thanks to the depth increase of CNN models [91, 182, 159, 59] and also to the fusion of CNNs models[182, 59]. The main problem of CNNs is the need of large datasets to avoid overfitting the network as well as the need of high computational power for training them.

Considering the use of different classifiers, in general, trained on the same data, one can observe that patterns misclassified by the different models would not necessarily overlap [88]. This suggests that they could potentially offer complementary information that can be used to improve the final performance [88]. An option to combine the outputs of different classifiers was proposed in [93], where the authors used what they call a decision templates scheme instead of simple aggregation operators such as the product or average. As they showed, this scheme maintains a good performance using different training set sizes and is also less sensitive to particular datasets compared to the other schemes.

In this chapter, we integrate the fusion concept into the CNN framework, with the purpose of demonstrating that the combination of the classifiers' output, by using a decision template scheme, allows to improve the performance on the food recognition problem. Our contributions are the following: 1) we propose the first food recognition algorithm that fuses the output of different CNN models, 2) we show that our CNNs fusion approach has better performance compared to the use of CNN models separately, and 3) we demonstrate that our CNNs Fusion approach keeps a high performance independently of the target (dishes, family of dishes) and dataset validating it on 2 public datasets.

The organization of the chapter is as follows. In section 7.2, we present the CNNs Fusion methodology. In section 7.3, we present the datasets, the experimental setup and discuss the results. Finally, in section 7.4, we describe the conclusions.

7.2 Methodology

In this section, we describe the CNN Fusion methodology (see Fig. 7.1), which is composed of two main steps: training K CNN models based on different architectures and fusing the CNN outputs using the decision templates scheme.

7.2.1 Training of CNN models

The first step in our methodology involves separately training two CNN models. We chose two different kind of models winners of the ILSVRC in the object recognition task. Both models won or are based on the winner of the challenges made in 2014 and 2015 proposing novel architectures: the first based its design on "inception models" and the second on "residual networks". First, each model was pre-trained on the ILSVRC data. Later, all layers were fine-tuned by a certain number of epochs, selecting for each one the model that provides the best results in the validation set and that will be used in the fusion step.

7.2.2 Decision templates for classifiers fusion

Once we trained the models on the food dataset, we combined the softmax classifier outputs of each model using the Decision Template (DT) scheme [93].

Let us annotate the output of the last layer of the k-th CNN model as $(\omega_{1,k}, \dots, \omega_{C,k})$, where $c = 1, ..., C$ is the number of classes and $k = 1, ...K$ is the index of the CNN model (in our case, K=2). Usually, the softmax function is applied, to obtain the probability value of model k to classify image x to a class c: $p_{k,c}(x) = \frac{e^{\omega_{k,c}}}{\sum_{c=1}^{C} e^{\omega_{k,c}}}$. Let us consider the k-th decision vector D_k:

$$D_k(x) = [p_{k,1}(x), p_{k,2}(x), ..., p_{k,C}(x)]$$

Definition [93]: A **Decision Profile,** DP for a given image x is defined as:

$$DP(x) = \begin{bmatrix} p_{1,1}(x) & p_{1,2}(x) & ... & p_{1,C}(x) \\ & \dots & & \\ p_{K,1}(x) & p_{K,2}(x) & ... & p_{K,C}(x) \end{bmatrix} \tag{7.1}$$

Definition [93]: Given N training images, a **Decision Template** is defined as a set of matrices $DT = (DT^1, \dots, DT^C)$, where the c-th element is obtained as the average of the decision profiles (7.1) on the training images of class c:

$$DT^c = \frac{\sum_{j=1}^{N} DP(x_j) \times Ind(x_j, c)}{\sum_{j=1}^{N} Ind(x_j, c)},$$

where $Ind(x_j, c)$ is an indicator function with value 1 if the training image x_j has a crisp label c, and 0, otherwise [92]. Finally, the resulting prediction for each image is determined considering the similarity $s(DP(x), DT^c(x))$ between the decision profile $DP(x)$ of the test image and the decision template of class c, $c = 1, \ldots, C$. Regarding the arguments of the similarity function $s(.,.)$ as fuzzy sets on some universal set with $K \times C$ elements, various fuzzy measures of similarity can be used. We chose different measures [93], namely 2 measures of similarity, 2 inclusion indices, a consistency measure and the Euclidean Distance. These measures are formally defined as:

$$S_1(DT^c, DP(x)) = \frac{\sum_{k=1}^{K}\sum_{i=1}^{C}\min(DT^c_{k,i}, DP_{k,i}(x))}{\sum_{k=1}^{K}\sum_{i=1}^{C}\max(DT^c_{k,i}, DP_{k,i}(x))},$$

$$S_2(DT^c, DP(x)) = 1 - \sup_u\{\left|DT^c_{k,i} - DP_{k,i}(x)\right| : i = 1, \ldots, C, k = 1, \ldots, K\},$$

$$I_1(DT^c, DP(x)) = \frac{\sum_{k=1}^{K}\sum_{i=1}^{C}\min(DT^c_{k,i}, DP_{k,i}(x))}{\sum_{k=1}^{K}\sum_{i=1}^{C}DT^c_{k,i}},$$

$$I_2(DT^c, DP(x)) = \inf_u\{\max(\overline{DT^c_{k,i}}, DP_{k,i}(x)) : i = 1, \ldots, C, k = 1, \ldots, K\},$$

$$C(DT^c, DP(x)) = \sup_u\{\min(DT^c_{k,i}, DP_{k,i}(x)) : i = 1, \ldots, C, k = 1, \ldots, K\},$$

$$N(DT^c, DP(x)) = 1 - \frac{\sum_{k=1}^{K}\sum_{i=1}^{C}(DT^c_{k,i} - DP_{k,i}(x))^2}{K \times C},$$

where $DT^c_{k,i}$ is the probability assigned to the class i by the classifier k in the DT^c, $\overline{DT^c_{k,i}}$ is the complement of $DT^c_{k,i}$ calculated as $1 - DT^c_{k,i}$, and $DP_{k,i}(x)$ is the probability assigned by the classifier k to the class i in the DP calculated for the image, x. The final label, L is obtained as the class that maximizes the similarity, s, the inclusion index, the consistency measure or the Euclidean distance between $DP(x)$ and DT^c: $L(x) = argmax_{c=1,\ldots,C}\{s(DT^c, DP(x))\}$.

7.3 Experiments

7.3.1 Datasets

The data used to evaluate our approach are two public datasets of very different images: Food-11 [153] and Food-101 [23], which are chosen in order to verify that the classifiers fusion provides good results regardless of the different properties of the target datasets, such as intra-class variability (the first one is composed of many dishes of the same general category, while the second one is composed of specific fine-grained dishes),

FIGURE 7.2: Images from the Food-11 dataset. Each image corresponds to a different class.

FIGURE 7.3: Example images of Food-101 dataset. Each image represents a dish class.

inter-class similarity, number of images, number of classes, images acquisition condition, among others.

Food-11 is a dataset for food recognition [153], which contains 16,643 images grouped into 11 general categories of food: bread, dairy products, dessert, egg, fried food, meat, noodle/pasta, rice, seafood, soup and vegetable/fruit (see Fig. 7.2). The images were collected from existing food datasets (Food-101, UECFOOD100, UECFOOD256) and social networks (Flickr, Instagram). This dataset has an unbalanced number of images for each class with an average of 1,513 images per class and a standard deviation of 702. For our experiments, we used the same data split, images and proportions, provided by the authors [153]. These are divided as 60% for training, 20% for validation and 20% for test, that is 9,866, 3,430 and 3,347 images for each set, respectively.

Food-101 is a standard to evaluate the performance of visual food recognition [23]. This dataset contains 101.000 real-world food images downloaded from foodspotting.com, which were taken under unconstrained conditions. The authors chose the top 101 most popular classes of food (see Fig. 7.3) and collected 1,000 images for each class: 75% for training and 25% for testing. With respect to the classes, these consist of very diverse and fine-grained dishes of various countries, but also with highly intra-class variation and inter-class similarity in most occasions. In our experiments, we used the same data splits provided by the authors. Unlike Food-11, and keeping the procedure followed by other authors [104, 111, 58], we validate and test our model on the same data split.

7.3.2 Experimental Setup

As usually, every CNN model was pre-trained on the ILSVRC dataset. Following, we adapted them by changing the output of the models to the number of classes for each target dataset and fine-tuned the models using the new images. For the training of the

CNN models, we used the Deep Learning framework Keras[1]. The models chosen for Food-101 dataset due to their performance-efficiency ratio were InceptionV3 [160] and ResNet50 [59]. Both models were trained during 48 epochs with a batch size of 32, and a learning rate of 5×10^{-3} and 1×10^{-3}, respectively. In addition, we applied a decay of 0.1 during the training of InceptionV3 and of 0.8 for ResNet50 every 8 epochs. The parameters were chosen empirically by analyzing the training loss.

As to the Food-11 dataset, we kept the ResNet50 model, but changed InceptionV3 by GoogLeNet [159], since InceptionV3 did not generalize well over Food-11. We believe that the reason is the small number of images for each class not sufficient to avoid overfitting; the model quickly obtained a good result on the training set, but a poor performance on the validation set. GoogLeNet and Resnet50 were trained during 32 epochs with a batch size of 32 and 16, respectively. The other parameters used for the ResNet50 were the same used for Food-101. In the case of GoogLeNet, we used a learning rate of 1×10^{-3} and applied a decay of 0.1 during every 8 epochs, that turned out empirically the optimal parameters for our problem.

7.3.3 Data Preprocessing and Metrics

The preprocessing made during the training, validation and testing phases was the following. During the training of our CNN models, we applied different preprocessing techniques on the images with the aim of increasing the samples and to prevent the overfitting of the networks. First, we resized the images keeping the original aspect ratio as well as satisfying the following criteria: the smallest side of the resulting images should be greater than or equal to the input size of the model; and the biggest side should be less than or equal to the maximal size defined in each model to make random crops on the image. In the case of InceptionV3, we set to 320 pixels as maximal size, for GoogLeNet and ResNet50 the maximal size was defined as 256 pixels. After resizing the images, inspired by [58], we enhanced them by means of a series of random distortions such as: adjusting color balance, contrast, brightness and sharpness. Finally, we made random crops of the images, with a dimension of 299x299 for InceptionV3 and of 224x224 for the other models. Then, we applied random horizontal flips with a probability of 50%, and subtracted the average image value of the ImageNet dataset. During validation, we applied a similar preprocessing, with the difference that we made a center crop instead of random crops and that we did not apply random horizontal flips. During test, we followed the same procedure than in validation (1-Crop evaluation). Furthermore, we also evaluated the CNN using 10-Crops, which are: upper left, upper right, lower left, lower right and center crop, both in their original setup and also applying an horizontal flip [91]. As for 10-Crops evaluation, the classifier gets a tentative label for each crop, and then majority

[1]www.keras.io

TABLE 7.1: Overall test set error rate of Food-11 obtained for each model. The distance measure is shown between parenthesis in the CNNs Fusion models.

Authors	Model	1-Crop	10-Crops	N/A
[153]	GoogLeNet	-	-	16.5%
us	GoogLeNet	9.89%	9.29%	-
us	ResNet50	6.57%	6.39%	-
us	CNNs Fusion (S_1)	6.36%	5.86%	-
us	CNNs Fusion (S_2)	6.12%	**5.65**%	-
us	CNNs Fusion (I_1)	6.36%	5.89%	-
us	CNNs Fusion (I_2)	6.30%	**5.65**%	-
us	CNNs Fusion (C)	6.45%	6.07%	-
us	CNNs Fusion (N)	6.36%	5.92%	-

voting is used over all predictions. In the cases where two labels are predicted the same number of times, the final label is assigned comparing their highest average prediction probability.

We used four metrics to evaluate the performance of our approach, overall Accuracy, Precision, Recall, and F_1 score.

7.3.4 Experimental Results on Food-11

The results obtained during the experimentation on Food-11 dataset are shown in Table 7.1 giving the error rate (1 - accuracy) for the best CNN models, compared to the CNNs Fusion. We report the overall accuracy by processing the test data using two procedures: 1) a center crop (1-Crop), and 2) using 10 different crops of the image (10-Crops). The experimental results show an error rate of less than 10 % for all classifiers, achieving a slightly better performance when using 10-Crops. The best accuracy is achieved with our CNNs Fusion approach, which is about 0.75% better than the best result of the classifiers evaluated separately. On the other hand, the baseline classification on Food-11 was given by their authors, who obtained an overall accuracy of 83.5% using GoogLeNet models fine-tuned in the last six layers without any pre-processing and post-processing steps. Note that the best results obtained with our approach have been using the point-wise measures (S2, I2). The particularity of these measures is that they penalize big differences between corresponding values of DTs and DP being from the specific class to be assigned as the rest of the class values. From now on, in this section we only report the results based on the 10-Crops procedure.

As shown in Table 7.2, the CNNs Fusion is able to properly classify not only the images that were correctly classified by both baselines, but in some occasions also when one or both fail. This suggests that in some cases both classifiers may be close to predicting the correct class and combining their outputs can make a better decision.

TABLE 7.2: Percentage of images well-classified and misclassified on Food-11 using our CNNs Fusion approach, distributed by the results obtained with GoogLeNet (CNN_1) and ResNet50 (CNN_2) models independently evaluated.

	CNNs evaluated independently			
CNNs Fusion	**Both wrong**	**CNN_1 wrong**	**CNN_2 wrong**	**Both fine**
Well-classified	3.08%	81.77%	54.76%	99.97%
Misclassified	96.92%	18.23%	45.24%	0.03%

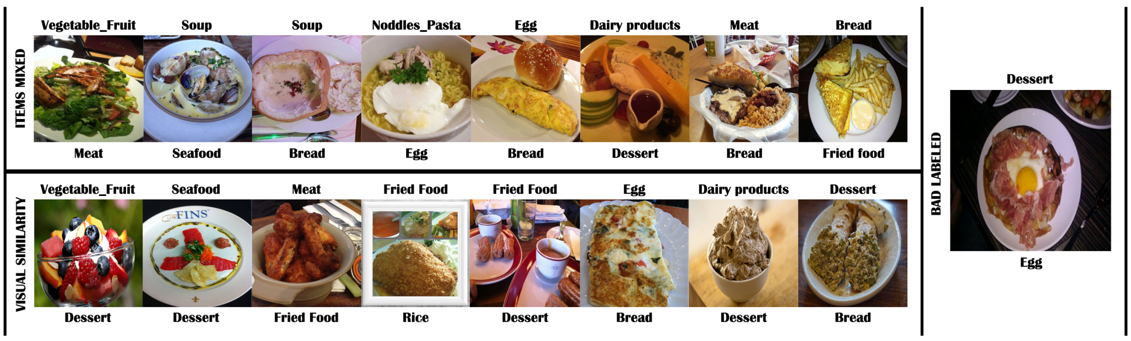


FIGURE 7.4: Misclassified Food-11 examples: predicted labels (on the top), and the ground truth (on the bottom).

Samples misclassified by our model are shown in Fig. 7.4, where most of them are produced by mixed items, high inter-class similarity and wrongly labeled images. We show the ground truth (top) and the predicted class (bottom) for each sample image.

In Table 7.3, we show the precision, recall and F_1 score obtained for each class separately. By comparing the F_1 score, the best performance is achieved for the class Noodles_Pasta and the worst for Dairy products. Specifically, the class Noddles_Pasta only has one image misclassified, which furthermore is a hard sample, because it contains two classes together (see items mixed in Fig. 7.4). Considering the precision, the worst results are obtained for the class Bread, which is understandable considering that bread can sometimes be present in other classes (e.g. soup or egg). In the case of recall, the worst results are obtained for Dairy products, where an error greater than 8% is produced for misclassifying several images as class Dessert. The cause of this is mainly, because the class Dessert has a lot of items in their images that could also belong to the class Dairy products (e.g. frozen yogurt or ice cream) or that are visually similar.

7.3.5 Experimental Results on Food-101

The overall accuracy on Food-101 dataset is shown in Table 7.4 for two classifiers based on CNN models, and also for our CNNs Fusion. The overall accuracy is obtained by means of the evaluation of the prediction using 1-Crop and 10-Crops. The experimental results show better performance (about 1% more) using 10-Crops instead of 1-Crop. From

TABLE 7.3: Some results obtained on the Food-11 using our CNNs Fusion approach.

Class	#Images	Precision	Recall	F1
Bread	368	**88.95%**	91.85%	90.37%
Dairy products	148	89.86%	**83.78%**	**86.71%**
Meat	432	94.12%	92.59%	93.35%
Noodles_Pasta	147	**100.00%**	**99.32%**	**99.66%**
Rice	96	94.95%	97.92%	96.41%
Vegetable_Fruit	231	98.22%	95.67%	96.93%

TABLE 7.4: Overall test set accuracy of Food-101 obtained for each model.

Author	Model	1-Crop	10-Crops	N/A
[58]	InceptionV3	-	-	88.28%
[111]	ResNet200	-	88.38%	-
[111]	WRN	-	88.72%	-
[111]	WISeR	-	90.27%	-
us	ResNet50	82.31%	83.54%	-
us	InceptionV3	83.82%	84.98%	-
us	CNNs Fusion (S_1)	85.52%	86.51%	-
us	CNNs Fusion (S_2)	86.07%	86.70%	-
us	CNNs Fusion (I_1)	85.52%	86.51%	-
us	CNNs Fusion (I_2)	85.98%	**86.71**%	-
us	CNNs Fusion (C)	85.24%	86.09%	-
us	CNNs Fusion (N)	85.53%	86.50%	-

now on, in this section we only report the results based on the 10-Crops procedure. In the same way as observed in Food-11, the best accuracy obtained with our approach was by means of point-wise measures S2, I2, where the latter provides a slightly better performance. Again, the best accuracy is also achieved by the CNNs Fusion, which is about 1.5% higher than the best result of the classifiers evaluated separately. Note that the best performance on Food-101 (overall accuracy of 90.27%) was obtained using WISeR [111]. In addition, the authors show the performance by another deep learning-based approaches, in which three CNN models achieved over a 88% (InceptionV3, ResNet200 and WRN [181]). However, WISeR, WRN and ResNet200 models were not considered in our experiments since they need a multi-GPU server to replicate their results. In addition, those models have 2.5 times more parameters than the models chosen, which involve a high cost computational especially during the learning stage. Following the chapter steps, our best results replicating the methods were those using InceptionV3 and ResNet50 models used as a base to evaluate the performance of our CNNs Fusion approach.

As shown in Table 7.5, in this dataset the CNNs Fusion is also able to properly classify

TABLE 7.5: Percentage of images well-classified and misclassified on Food-101 using our CNNs Fusion approach, distributed by the results obtained with InceptionV3 (CNN_1) and ResNet50 (CNN_2) models independently evaluated.

	CNNs evaluated independently			
CNNs Fusion	**Both wrong**	**CNN_1 wrong**	**CNN_2 wrong**	**Both fine**
Well-classified	1.95%	73.07%	64.95%	99.97%
Misclassified	98.05%	26.93%	35.05%	0.03%

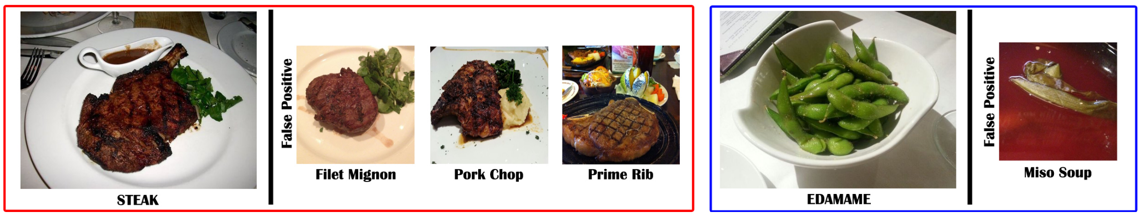


FIGURE 7.5: Misclassified examples for the Food-101 classes that obtained the worst (steak) and best (edamame) classification results by F1 score (groundtruth label - bottom).

not only the images that were correctly classified for both classifiers, but also when one or both fail. Therefore, we demonstrate that our proposed approach maintains its behavior independently of the target dataset.

Table 7.6 shows the top three worst and best classification results on Food-101 classes. We highlight the classes with the worst and best results. As for the worst class (Steak), the precision and recall achieved are 60.32% and 59.60%, respectively. Interestingly, about 26% error in the precision and 30% error in the recall is produced with only three classes: Filet mignon, Pork chop and Prime rib. As shown in Fig. 7.5, these are fine-grained classes with high inter-class similarities that imply high difficulty for the classifier, because it should identify small details that allow to determine the corresponding class of the images. On the other hand, the best class (Edamame) was classified achieving 99.60% of precision and 100% of recall. Unlike Steak, Edamame is a simple class to classify, because it has a low intra-class variation and low inter-class similarities. In other words, the images in this class have a similar visual appearance and they are quite different from the images of the other classes. Regarding the only one misclassified image, its visual appearance is close to the class Edamame as for the shape and color.

7.4 Conclusions

In this chapter, we addressed the problem of food recognition and proposed a CNNs Fusion approach based on the concepts of decision templates and decision profiles and their similarity that improves the classification performance with respect to using CNN

TABLE 7.6: Top 3 better and worst classification results on Food-101.

Class	**Precision**	**Recall**	**F1**
Spaghetti Bolognese	94.47%	95.60%	95.03%
Macarons	97.15%	95.60%	96.37%
Edamame	**99.60%**	**100.00%**	**99.80%**
Steak	**60.32%**	**59.60%**	**59.96**%
Pork Chop	75.71%	63.60%	69.13%
Foie Gras	72.96%	68.00%	70.39%

models separately. Evaluating different similarity measures, we show that the optimal one is based on the infinimum of the maximum between the complementary of the decision templates and the decision profile of the test images. On Food-11, our approach outperforms the baseline accuracy by more than 10% of accuracy. As for Food-101, we used two CNN architectures providing the best state-of-the-art results where our CNNs Fusion strategy outperformed them again. As a future work, we plan to evaluate the performance of the CNN Fusion strategy as a function of the number of CNN models. Furthermore, it would be interesting to analyze epistemic uncertainty to weigh the importance of each model within an ensemble for the classification of the input image.

Chapter 8

Grab, Pay and Eat: Semantic Food Detection for Smart Restaurants

8.1 Introduction

Having a poor routine of physical exercises and poor nutritional habits are two of the main possible causes of people's health-related issues like obesity or diabetes, among others. For these reasons, nowadays people are more concerned about these aspects of their daily life. Therefore, the need for applications that allow to keep track of both physical activities and nutrition habits are rapidly increasing, a field in which the automatic analysis of food images plays an important role. Focusing on self-service restaurants, food recognition algorithms could enable both monitoring of food consumption and the automatic billing of the meal grabbed by the customer. The latter is quite relevant because remove the need for a manual selection of the chosen dishes, allowing to speed-up the service offered by these restaurants.

From the computer vision side, several approaches have been proposed to tackle the problem, most of them using Convolutional Neural Networks [75, 130, 2, 153]. Several of the published work consider the development of methods for food recognition, i.e. being able to recognize the dish depicted in a picture in which a single plate is shown. An important consideration to take into account when modeling visual food-related information is its fine-grained nature, meaning that specially in the problem of food analysis the intra and inter-class similarity are hardly making difficult the problem of obtaining robust food recognition methods.

Several works in the literature have proposed methods for food intake self-monitoring [8, 164], in which the user should take pictures of each meal and the system would consequently track any nutritional information associated. Other approaches related to the problem of food intake include food portion estimation by using two images acquired by mobile devices [36]; food ingredients recognition from recipes using CNNs as multi-label

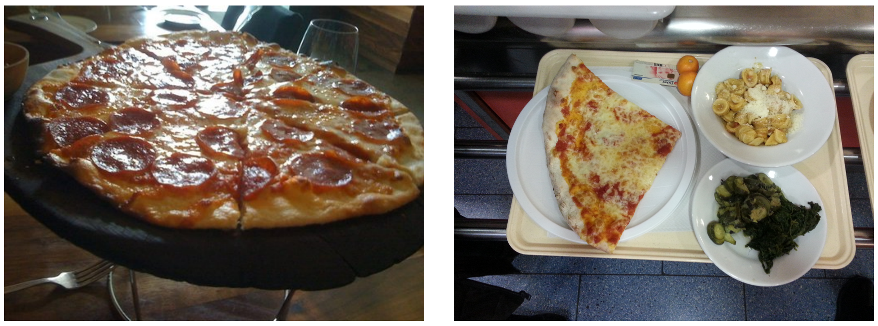


FIGURE 8.1: Example of images used in traditional approaches to food analysis (left) and food tray analysis (right).

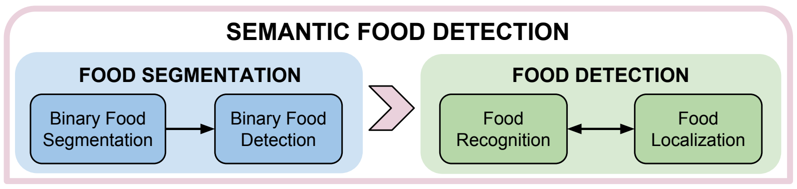


FIGURE 8.2: Main tasks of our Semantic Food Detection framework.

predictors [20, 27]; multimodal multitask deep belief networks for learning both visual information and image-ingredient representation [118]; bayesian models for analyzing similarities between cuisines [117]; or cross-modal learning for multi-attribute recognition and recipe retrieval [26].

Instead of applying personalized tracking, there are several contexts where social monitoring or recognition is required. A clear example is food tray detection in public spaces [31, 32], where the sample consists of a tray picture that includes all the food that a user is about to consume (see Fig. 8.1) and the model is intended to process all pictures from any possible users taking food at the same restaurant. The development of a system able to apply food tray detection in a controlled, but social and public environment could enable several applications. The most straightforward context of applicability would be automatic billing in self-service restaurants, where the system could solve the need for a person selecting what the customer grabbed before paying. A different application could consider the design of smart trays [131], which could provide food recommendations depending on what the customer is selecting. The provided recommendations could be based on calorie counting, healthy food, specific nutritional composition, etc. In addition, if we also consider a system able to log the food consumed by every individual along time, it could provide health-related recommendations in a long-term way.

There are several aspects that make the food tray analysis a challenging problem [32]: 1) multiple foods placed on the same placemat, 2) different foods served in the same dish, 3) visual distortions and illumination changes due to shadows, and 4) objects placed on a tray that do not correspond to any type of food. On the other hand, unlike traditional approaches to food analysis, difficulties due to intra-class variability have less influence on the problem of food tray detection.

In this work, we propose a novel method that unifies the problems of food detection, localization, recognition and segmentation into a new framework that we call Semantic Food Detection. As Fig. 8.2 shows, we integrate the information extracted by two main approaches: a) food segmentation and b) object detection trained for food detection, by taking advantage of the benefits provided by both algorithms in a CNN framework. The first one allows us to determine where the food is in terms of pixel and bounding boxes. The second one allows us to locate and recognize the foods present in the images. The Semantic Food Detection framework combines the information that both algorithms provide in order to prevent false food detections and thus provide a better performance.

Our main contributions are: 1) a novel framework that integrates the problems of food detection, localization, recognition and segmentation; and 2) a novel approach to address the problem of food tray analysis, that integrates a fully convolutional network for semantic segmentation and a convolutional neural network for object detection through a probabilistic approach and a custom non-maximum suppression. Our method achieves about 90% in terms of F2-score, and it is able to outperform the state-of-the-art methods by more than 10% and 20% with respect to recall and mean average accuracy.

The remainder of this chapter is organized as follows: Section 8.2 includes an overview of the related work, Section 8.3 presents the proposed Semantic Food Detection approach, Section 8.4 shows the experimental results and discussion, and Section 8.5 closes with the conclusions and future research.

8.2 Related Work

Nowadays, there is a great interest in conducting research for visual food analysis, mainly in its applicability for diet monitoring based on the intrinsic nutritional information contained in food images. In this field, researchers have focused on different several aspects related to automatic food analysis.

The most basic aspect tackled in the literature is the *binary food detection* problem that determines the presence or absence of food in an image. This problem is also called food/non-food classification or food detection [76]. The first approximation was proposed by Kitamura et al. [87], who combine a BoF model and a SVM achieving a high accuracy on a tiny dataset of 600 images. An improvement of about 4% is achieved in terms of overall accuracy using a CNN-based method [76]. From this, numerous researchers have proposed CNN-based models either for feature extraction [130, 2] or for the whole recognition process [75, 153]. The best results obtained on public datasets with more than 15,000 images [75, 130] have been reported in [2] through the combination of CNN GoogLeNet for feature extraction, PCA for dimension reduction and SVM for classification. As for its applicability, this problem has commonly been investigated for indexing WEB images [87] or as a pre-processing method for an automatic food recognition system [75, 153]. It has been also used to detect bounding boxes in an food images [21], and to automate the process of image cleaning required when gathering images of a food dataset [115].

In food analysis, once images containing food are identified, *food recognition* is usually the next step to apply. Again, CNN-based models have been able to progressively improve the results of food recognition models reaching an accuracy of about 90% in datasets with around 100 different food classes [111]. In general, the best proposals are based on the winning models of the ILSVRC challenge [141], and a fine-tuning process is usually applied either making some architectural model changes (e.g. addition or removal of layers) [173, 104] or not [58]. Several datasets have been proposed to tackle this problem: a) datasets including fine-grained classes (e.g. apple pie, pork chop, pizza), like UECFOOD-256 [81] or Food-101 [23]; and b) datasets based on high-level categories (e.g. dessert, meat, soup), like Food-11 [153]. The best result when using fine-grained classes was achieved by the WISeR model [111], which combines the food traits and the vertical structure of some food, extracted by the standard squared convolutional kernel and the proposed slice convolutional kernel, respectively. Regarding the high-level categories, the best results were obtained by [1] through a novel approach that fuses several CNN models, achieving a 10% improvement in terms of accuracy with respect to the baseline method.

Most of the approaches focused on food recognition only exploit the visual content, but they ignore the context. However, geolocation and other information have also been explored in the literature for restaurant-oriented food recognition: on-line restaurant information is used in [18], similarly to [15] in which nutritional information is also retrieved; whilst the menu, the location and user images of dished are used in [172]. On the other hand, Herranz et al. [61] go a step further since their target is not only to improve both classification performance and efficiency, but also to better model contextual data and its relation with the other elements.

To date, most food recognition algorithms and datasets focus on classifying images that include only one dish [111, 104, 58]. However, in some cases, there may be more than one dish in the image and, in some cases, the dish can contain several kinds of food. *Food localization* and *food segmentation* are two tasks intended to cope with these problems. The former consists in extracting the regions of the images where the food is located. Up to our knowledge, the only available approach that does not require segmenting the food before extracting the bounding boxes is the one proposed by [21]. The task of food segmentation consists in classifying each pixel of the images representing a food. The latest research for food segmentation proposes an automatic weakly supervised methods [150, 151], which are based on Deep Convolutional Neural Networks and Distinct Class-specific Saliency Maps, respectively.

Regarding image segmentation for general purposes, fully convolutional networks (FCNs) [148] are the state-of-the-art in semantic segmentation. They are composed of convolutional layers only, i.e they do not have any fully-connected layer. They consist of a down-sampling path and an up-sampling path, which allow to take input images of arbitrary size and produce outputs of equivalent size, by means of an efficient inference and learning process. Several FCN models can be found in the literature applied to semantic segmentation. SegNet [14] is a deep FCN that consists of a VGG16-based encoder, a decoder and a final pixel-wise classification layer. DeepLab [28] uses *atrous convolutions* in the up-sampling path, allowing to incorporate larger context with no increase in parameters. RefineNet [102] is a multi-path refinement network that allows to obtain high-resolution predictions by using residual connections. PSPNet [190] is a pixel-level prediction framework that includes a pyramid pooling module to exploit the capability of global context information. Tiramisu [69] is an extension of Densely Connected Convolutional Neural Networks (DenseNets) for semantic segmentation, based on the idea of connecting each layer to every other layer in a feed-forward fashion. Its main benefits include a more accurate and easier training, with much less parameters.

According to the experimentation presented in the respective manuscripts, PSPNet [190] and Tiramisu [69] are the most competitive models. PSPNet is based on Residual Networks (ResNets), whilst Tiramisu is based on DenseNets. DenseNets can be seen as an extension of ResNets, with some characteristics that make them very appropriate for semantic segmentation problems: parameter efficiency, implicit deep supervision, and feature reuse. For all these reasons, Tiramisu will be the model of reference in our research.

In this manuscript, we deal with the identification of different foods placed on a food tray, by integrating the four food analysis problems mentioned above. To the best of our knowledge, only one approach with this purpose has been evidenced in the literature [32]. The authors introduced an additional food dataset composed of images taken in a canteen environment named UNIMIB2016. In addition, they proposed a pipeline for food recognition that performs classification based on the candidate regions obtained by combining two separate images segmentation processes, through saturation and color texture (JSEG). The best result was achieved by combining global and local (patch-based) classification approaches. Regarding the classification, for each region, they are carried out both in a sub-image (global strategy) and in several image patches (local strategy), the feature extraction using a CNN model based in AlexNet, and then the classification by an SVM. Furthermore, in the local strategy, an additional post-processing phase is needed to

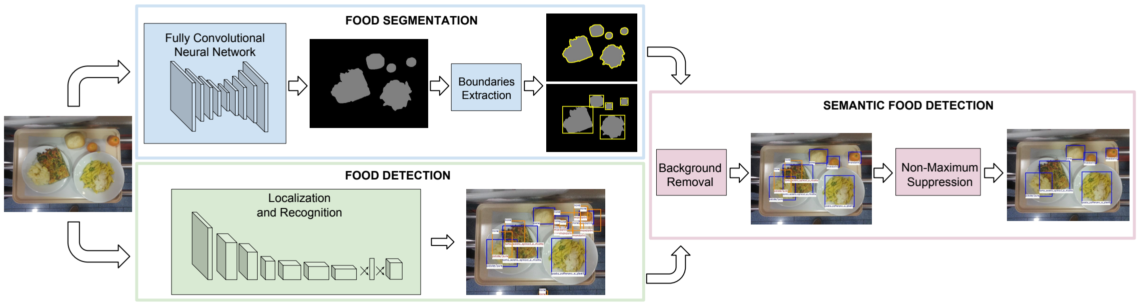


FIGURE 8.3: Detailed workflow of the proposed Semantic Food Detection method: food segmentation and food detection methods are applied in parallel, before combining them for a final detection on food tray images.

merge the labels of all image patches of the respective region. Then, the classification obtained by both approaches is combined by exploiting the sum of posterior probabilities to judge the final classification decision. Our approach differs mainly in three aspects: 1) we perform semantic segmentation by learning the best discriminant features between different foods from the dataset instead of using a segmentation approach based on generic image processing methods; 2) we locate and classify simultaneously all the foods placed in the tray by considering the context instead of performing the classification for each region individually, which implies a significant improvement in both result and processing time; and 3) we integrate the outputs of both methods to avoid false detections and thus make better decisions, instead of performing the classification directly based on the segmentation results. Additionally, our method is able to perform the food segmentation and detection processes in parallel, allowing to speed up the processing time.

8.3 Semantic Food Detection

This work proposes a method for food tray semantic detection that integrates food vs non-food semantic segmentation with food localization and recognition. Fig. 8.3 depicts the pipeline of our approach, subsequently explained in detail.

8.3.1 Food Segmentation

Food segmentation deals here with the problem of separating the food and food-related items, from the tray and other background elements, thus obtaining a binary image. For this purpose, we apply semantic segmentation techniques that work in a supervised learning framework, unlike the most segmentation methods that focus on image properties (e.g. color or texture). Notice that semantic segmentation could be used to directly segment the input image into the different food categories. However, the most recent methods in this field provide great results with datasets that contain a relatively low number of classes, such as CamVid with 11 semantic classes or Gatech with 8 [69]. The number of categories used in food analysis is much higher, thus increasing the difficulty of the task and providing not so satisfactory results [148].

Among the FCN models found in the literature applied to semantic segmentation, the Tiramisu model was considered [69], as mentioned in Section 8.2. Its down-sampling and up-sampling paths are connected by skip connections, and its architecture is composed of dense blocks, each one of them containing a set of concatenated layers for a better training.

After training our FCN model with food tray images, the binary images predicted by it are used in the next step, which aims at tracing the exterior boundaries of the food regions, avoiding the holes inside them. In this manner, small holes that may appear inside regions are discarded and thus the regions are homogenized. For this task, we use the Moore-Neighbor tracing algorithm modified by Jacob's stopping criteria [53].

Once the boundaries are traced, the bounding boxes that contain the regions are determined, thus obtaining a binary food detection. As small regions may also appear in the predicted images, and they usually correspond to false positives, this step also includes their elimination by considering a threshold criterion. Fig. 8.4 illustrates an example of the outputs obtained in the food segmentation procedure, including the binary image provided by the FCN model, the boundaries extracted and the bounding boxes generated.

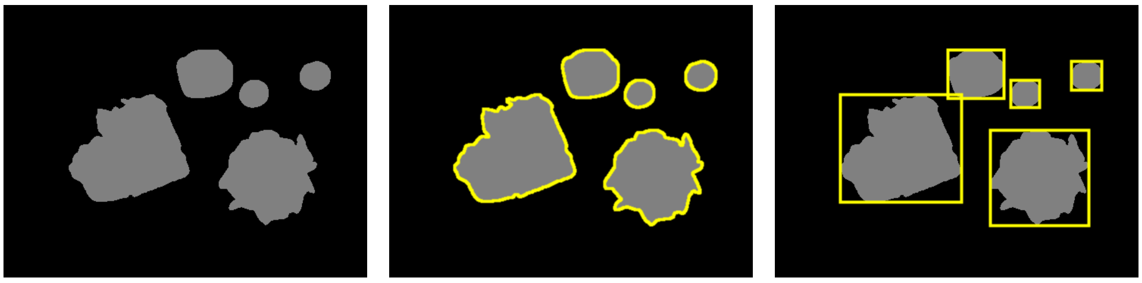


FIGURE 8.4: Food segmentation output (left to right): binary prediction by FCN, regions boundaries and food bounding boxes.

8.3.2 Food Detection

In this work, following the definition of the object detection problem [141], we consider as Food Detection the localization and recognition of food. For this purpose, we propose re-training an object detection algorithm to apply food detection instead. In particular, we chose one of the best object detection approaches in the state-of-the-art, YOLOv2 [134, 135]. As for the model, the authors propose a new FCN called Darknet-19, composed by 19 convolutional layers and 5 max pooling layers to tackle the recognition task. They modified this network for object detection by removing the last convolutional layer and adding four convolutional layers for producing 13x13 feature maps. At each cell on the output feature maps, the network predicts B bounding boxes with five coordinates for each, among them is the confidence score t_o, and $c = 1, \ldots, C$ conditional class probabilities, $Pr(Class_c|Object)$. Predictions are obtained from the last convolutional layer having a size equal to 1×1 and F filters, where the number of filters is calculated as: $F = (B \times (5 + C))$. From this, it is possible to determine the class-specific confidence score, CS_c for each bounding box as follows:

$$CS_c = Pr(Class_c|Object) * \sigma(t_o) \tag{8.1}$$

where $\sigma(.)$ stands for a logistic activation to constrain the predictions to fall in the range between 0 and 1. Note that, in the experiment, we use the original setting of B (equal to 5).

8.3.3 Semantic Food Detection

In object detection, one of the most common errors are false positives, which can be classified based on the type of error: localization error, confusion with similar objects, confusion with dissimilar objects, and confusion with background [63]. Our Semantic Food Detection proposal focuses on reducing two of the most common errors of object detectors [134]: localization errors, specifically those corresponding to duplicate detections; and errors produced by the confusion with the background. For this purpose, we propose the following procedure that integrates the detection and segmentation algorithms:

Background Removal

The first step involves the application of both boundaries extracted (contour and bounding box) from the Food Segmentation procedure in order to remove the background detections. Let $Y = \{b_1^Y, \ldots, b_N^Y\}$ be the set of bounding boxes obtained with the detection method, $S_1 = \{b_1^S, \ldots, b_L^S\}$ and $S_2 = \{c_1^S, \ldots, c_L^S\}$ the set of bounding boxes and contours extracted by the Food Segmentation method, respectively. Considering each element belonging to the sets named above as a set of points (x, y) that defines a polygon, we calculate the probability of a bounding box, b_i^Y to belong to the background Bkg as follows:

$$Pr(Bkg|b_i^Y) = \min(CS_c(\overline{b_i^Y}), \max(Pr(\overline{S_1|b_i^Y}), Pr(\overline{S_2|b_i^Y})))$$

where $CS_c(\overline{b_i^Y})$ is the complement of the confidence score, $1 - CS_c(b_i^Y)$ for the i-th detection, $Pr(\overline{S_1|b_i^Y})$ is the probability that b_i^Y is a false detection on the extracted boxes, S_1:

$$Pr(\overline{S_1|b_i^Y}) = 1 - \max_{j=1,\ldots,L} \frac{|b_i^Y \cap b_j^S|}{|b_i^Y|}$$

where $|.|$ stands for the cardinality of a set of pixels corresponding to an image region, and $Pr(\overline{S_2|b_i^Y})$ the probability that b_i^Y does not intersect with any contour in S_2:

$$Pr(\overline{S_2|b_i^Y}) = \min_{j=1,\dots,L} Ind(b_i^Y \cap c_j^S = \emptyset)$$

where $Ind(*)$ is an indicator function with value 1 if the condition is true, and 0 otherwise.

Bounding boxes with a probability higher than 50% to be background ($Pr(Bkg|b_i^Y) > T, T = 0.5$) are considered to be false detections, and are therefore removed.

Non-Maximum Suppression

The second step involves the application of a greedy procedure to eliminate duplicate detections by non-maximum suppression [49]. Once the Background Removal is applied, the remaining detections $Y' \subseteq Y$ are sorted in descending order by the confidence score $CS_c(b_j^Y)$ and grouped into C sets $Y^1, \dots, Y^C \subset Y'$, where C is the number of classes. Then, for each $Y^c, c = 1, ...C$, we greedily select the highest scoring bounding boxes while removing detections that are lower in the ranking and their maximum intersection ratio (MIR) with respect to the i-th previously selected bounding boxes is more than 50%, where MIR score for the j-th bounding box is calculated as:

$$MIR_j = \max_{\forall i, i<j} \frac{|b_i^Y \cap b_j^Y|}{\min(|b_i^Y|, |b_j^Y|)}$$

Notice that the chosen food detection method already incorporates a non-maximum suppression procedure. In our framework, we propose an additional personalized non-maximum suppression that differs mainly in two aspects: 1) we consider the predicted classes for the bounding boxes, and 2) we propose a MIR score instead of the traditional IoU. The last one was applied because in some cases the overlapped predictions for the same class could have a completely different dimension and proportion, and then, the IoU score will be very small even if one bounding box is completely inside the other.

8.4 Experimental Results

In this section, we first describe the dataset used to evaluate the proposed approach, which is composed of images taken in self-service restaurants. Then, we describe the evaluation measures used and present the results obtained with the different methods and model configurations.

8.4.1 Dataset

UNIMIB2016 [32] is a food dataset that has been collected in a self-service canteen. Each image includes a tray with some food placed both on plates and placemats. The acquisition process was performed on a semi-controlled environment using a Samsung Galaxy S3 smartphone. As a result, images acquired have a resolution of 3264×2448 in RGB, and present visual distortions and variable illuminations, making them challenging for any task of automatic food analysis.

The dataset is composed of 1,027 images that include a total of 73 food categories. Among them, only 1,010 images and 65 categories were used for experimentation, as suggested in [32] due to the low number of samples of the categories not considered. For experimental purposes, the dataset has been split in training and test sets: the former contains 650 images ($\approx 64\%$), whilst the latter contains 360 ($\approx 36\%$).

The annotations included in the dataset contain, for each food item: the polygon defining its boundaries, the bounding box and the food label. Fig. 8.5 illustrates an image of the UNIMIB2016 dataset with its corresponding annotations.

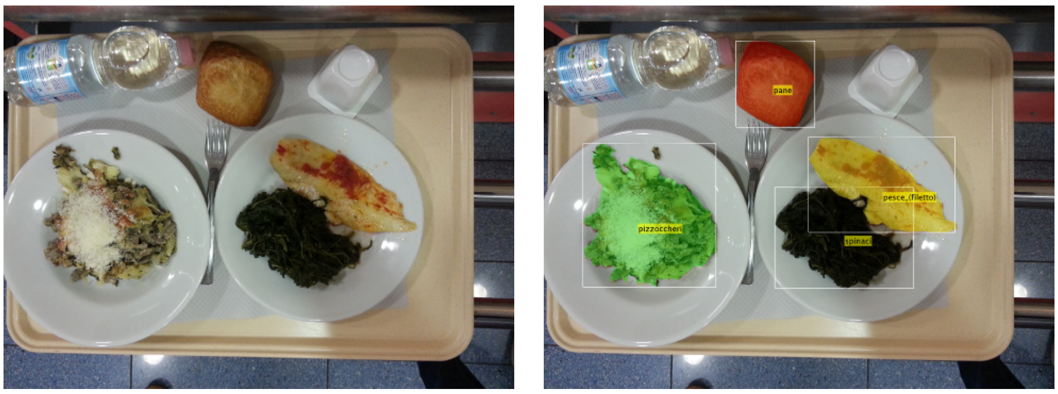


FIGURE 8.5: A representative sample of the UNIMIB dataset [32]: original image (left) and food annotations (right).

8.4.2 Food Segmentation

Metrics. In order to evaluate the different food segmentation approaches, several performance measures have been used. First, two pixel-wise metrics commonly used in semantic segmentation problems have been considered [148]:

- *Global pixel accuracy (GA).* The pixel-wise accuracy computed over all the pixels of the dataset.
- *Intersection over Union (IoU).* Also known as Jaccard index, it is defined as:

$$IoU(c) = \frac{\sum_i t_i == c \wedge p_i == c}{\sum_i t_i == c \vee p_i == c} \tag{8.2}$$

where c is a class, i represents all the pixels of the dataset, t_i are the target labels, and p_i are the predicted labels. Note that this metric is calculated for each single class c, and then the mean across the classes is computed.

To perform a fair comparison with [32], three region-based metrics have been also considered [13]:

- *Covering (CO)*. The covering of the ground truth (GT) by the segmented (S) images measures the level of overlapping between each pair of regions (R and R'):

$$C(S \rightarrow GT) = \frac{1}{N} \sum_{R \in GT} |R| \cdot \max_{R' \in S} \frac{|R \cap R'|}{|R \cup R'|} \tag{8.3}$$

  where N is the number of pixels of the image.

- *Rank index (RI)*. It compares the compatibility of assignments between pairs of elements in the ground truth (GT) and the segmented (S) images:

$$\begin{aligned} &RI(S, GT) = \\ &\frac{1}{\binom{N}{2}} \sum_{i<j} \left[\mathbb{I}(t_i == t_j \wedge p_i == p_j) + \mathbb{I}(t_i \neq t_j \wedge p_i \neq p_j) \right] \end{aligned} \tag{8.4}$$

  where $\binom{N}{2}$ is the number of possible unique pairs among the N pixels of each image, and $\mathbb{I}$ is the identity function.

- *Variation of information (VI)*. It measures the distance between the ground truth (GT) and the segmented (S) images in terms of their average conditional entropy:

$$VI(S, GT) = H(S) + H(GT) - 2 \cdot MI(S, GT) \tag{8.5}$$

  where H and MI are, respectively, the entropy and the mutual information. In this case, the lower the better.

Notice that these three metrics are calculated for each single image, and then the mean across images is computed.

Experimental setup. Regarding the methods used for semantic segmentation, we trained three networks based on Tiramisu [69]: 1) *Tiramisu56*: 56 layers, with 4 layers per dense block and a growth rate of 12; 2) *Tiramisu67*: 67 layers, with 5 layers per dense block and a growth rate of 16; and 3) *Tiramisu103*: 103 layers, with a variable number of layers per dense block (from 12 to 4 in the downsampling path, and from 4 to 12 in the upsampling) and a growth rate of 16. Additionally, the *Classic Upsampling*, which uses standard convolutions in the upsampling path instead of dense blocks [138], has been also considered for comparative purposes.

All the FCN models were trained with the UNIMIB2016 dataset [32] (images resized to 360×480), and two-target labels: food vs non-food. The models were initialized with HeUniform and trained with RMSprop [69]. The training process consists of two steps: first, the models were trained with cropped images (224×224) for data augmentation and batch size 3, with an initial learning rate of $1e-3$ and an exponential decay of 0.995 per epoch; and second, their parameters were fine-tuned with full size images (360×480) and batch size 1, using a learning rate of $1e-4$. The outputs were monitored using the global accuracy and the IoU, with a patience of 100 during pre-training and 50 during fine-tuning.

Table 8.1 includes the results achieved with the four networks for semantic segmentation, as well as with the two segmentation methods from [32]: the JSEG algorithm [38], and the segmentation pipeline proposed in [32]. With respect to the pixel-wise measures, all the networks produced competitive results (over 0.96). The Tiramisu models outperformed the Classic Upsampling, thanks to the dense blocks, despite a lower number of parameters used. In general, the Tiramisu model benefits from having more parameters and depth. However, in this binary problem the Tiramisu103 produced overfitting whilst the Tiramisu67 achieved the best results, with a good trade-off between depth and performance. Regarding the region-based measures, all the FCNs provided better results than the two approaches from [32], which demonstrated the adequacy of the proposed methods for our problem.

TABLE 8.1: Results obtained with our Food Segmentation approach in test set.

	No.	Pixel-wise		Region-based		
	param	GA	IoU	CO	RI	VI
JSEG [38]	-	-	-	0.385	0.389	3.106
Ciocca et al. [32]	-	-	-	0.916	0.931	0.429
Classic Upsam.	12.7M	0.991	0.962	0.984	0.982	0.125
Tiramisu56	1.4M	0.992	0.967	0.986	0.984	0.112
Tiramisu67	3.5M	**0.993**	**0.971**	**0.987**	**0.986**	**0.105**
Tiramisu103	9.4M	0.992	0.968	0.986	0.984	0.111

8.4.3 Semantic Food Detection Performance

Metrics. In order to evaluate food recognition and localization, we chose three standard measures commonly used in multi-class object recognition problems:

- *Recall (Rec).* The proportion of true positives detected.
- *Precision (Pre).* The proportion of the true positives against all the positive results.

- *F_β-measure*. A weighted average of precision and recall. We use $\beta = 2$ (F_2) to place more emphasis on wrong classified or undetected foods.

For comparative purposes, the measures used by Ciocca et al. [32] were also considered:

- *Standard Accuracy (SA)*. It is equivalent to the recall.
- *Macro Average Accuracy (MAA)*. The proportion of correctly classified foods, but taking into account the class imbalance of the dataset:

$$MAA = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{NF_c}, \tag{8.6}$$

  where C is the number of classes, TP_c is the number of correctly classified foods of class c, and NF_c is the total number of foods of class c.

- *Tray Accuracy (TA)*. The percentage of trays for which all the foods contained are correctly recognized:

$$TA = \frac{1}{T}\sum_{t=1}^{T} Ind(\frac{TP_t}{NF_t} = 1), \tag{8.7}$$

  where T is the number of food tray images, TP_t is the number of correctly classified foods on the tray t, and NF_t is the total number of foods on the tray t.

Experimental setup. YOLOv2 was first pre-trained on the ILSVRC dataset. Following, we adapted it by changing the output of the model to 65 classes and applied a fine-tuning using UNIMIB2016 images (resized to 416×416). For training the model, we used the framework Darknet [133]. The models were trained during 4000 iterations with a batch size of 32, and a learning rate of $1e-3$. In addition, we applied a decay of 0.9 to the iterations 3000 and 3500. To avoid overfitting, we use standard data augmentation procedures with random crops and distortions in the HSV color space [135].

Once YOLOv2 training is completed, the next step is to determine the confidence threshold to be used during localization and recognition of the food. A low confidence threshold implies a greater number of detections, which maximizes the likelihood that all the foods present in the image will be detected. At the same time, it also increases the chances of obtaining false detections. Taking into account that the confidence defined by the detection method considers two factors (the fit of the bounding box to the object and the predicted class), we chose the minimum threshold according to the number of classes. Given that the target dataset has 65 classes, the minimum threshold chosen is $\frac{1}{65}$. With this value, it can be interpreted that the bounding boxes extracted will have a recognition probability greater than a random value when the detected bounding box fits the object perfectly. Following the interpretation given, we chose $\frac{1}{2}$ as maximum threshold, which implies a high probability, at least 50%, that the localized object is correctly classified.

TABLE 8.2: Results obtained by YOLOv2 and the proposed approach in training set using different confidence thresholds.

		1/65	1/32	1/16	1/8	1/4	1/2
YOLOv2 [135]	*Pre*	**0.511**	0.687	0.832	0.926	0.968	0.994
	Rec	0.999	0.998	0.997	0.995	0.988	0.966
	MAA	0.999	0.997	0.995	0.992	0.981	0.952
	TA	0.997	0.992	0.988	0.982	0.960	0.895
Proposed	*Pre*	**0.918**	0.952	0.973	0.984	0.991	0.996
	Rec	0.998	0.997	0.996	0.994	0.987	0.965
	MAA	0.999	0.999	0.996	0.994	0.981	0.951
	TA	0.995	0.992	0.988	0.982	0.957	0.894

TABLE 8.3: Tray Food Analysis results, from top to bottom: food detection method 1) without segmentation, 2) with segmentation, and 3) with ground-truth segmentation to perform the recognition. The best results per block are in boldface.

	F_2	*Pre*	*Rec*	*MAA*	*TA*
YOLOv2 [135]	0.786	0.489	**0.927**	**0.850**	**0.769**
YOLOv2 + 8.3.3	**0.856**	**0.659**	**0.925**	**0.849**	0.772
Ciocca et al. [32]	-	-	0.798	0.636	**0.789**
YOLOv2 + 8.3.3	0.844	0.628	**0.923**	**0.846**	0.761
Proposed	**0.905**	**0.841**	**0.922**	**0.845**	0.764
Mezgec et al. [115]	-	-	0.864	-	-
Ciocca et al. [32]	-	-	0.891	0.684	**0.871**
YOLOv2 + 8.3.3	0.854	0.651	**0.926**	**0.850**	0.769
Proposed	**0.911**	**0.856**	**0.926**	**0.850**	0.775

Table 8.2 shows the results obtained in the training set using different confidence thresholds. The tested thresholds range from the minimum and maximum values mentioned above. As can be observed, when the threshold increases, the precision also increases considerably, whilst the rest of the indicators are hardly affected. When comparing the

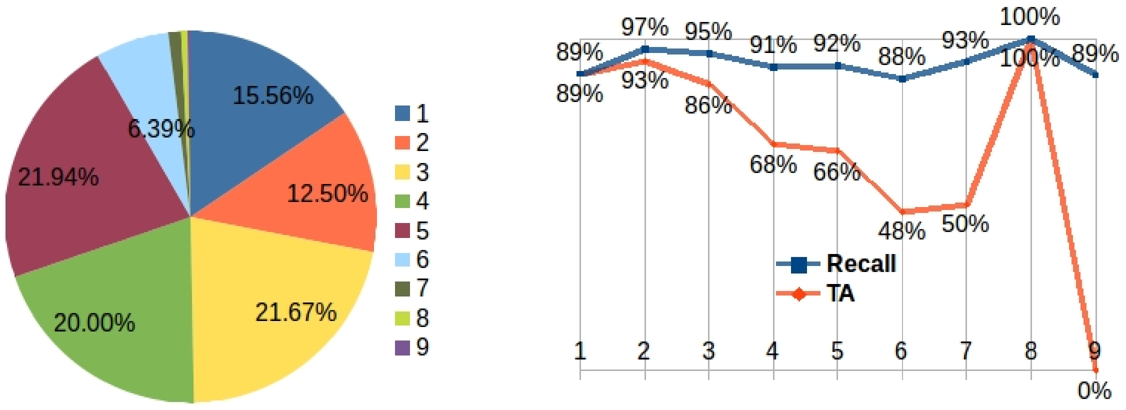


FIGURE 8.6: a) Distribution of the trays according to the number of foods that are placed in them. b) Results in terms of *Recall* (blue) and *TA* (orange), for each item of the distribution.

results obtained between YOLOv2 and the proposed method, for the minimum threshold, it can be observed that a significant improvement in precision is obtained ($\approx$40%) with only a slight decrease in the other indicators (0.1%-0.2%). Another interesting aspect to highlight is the comparison of results when using the maximum threshold, since they are practically identical for both methods. This means that, for a threshold of $\frac{1}{2}$, there are almost no false detections that can be reduced with our procedure. For the remaining experiments, the minimum threshold was chosen for two main reasons: 1) it obtains the best results for the *Recall*, MAA and *TA* indicators; and 2) it allows us to discard the false positives that appear when combining the results with the food segmentation procedure.

The Semantic Food Detection results on the test set are shown in Table 8.3. In order to see the performance of the different parts of our pipeline, we group the results of this table in three rows: the first one corresponds to the results obtained with YOLOv2 retrained for food detection, and our proposed framework without considering the information extracted from the segmentation method to perform the classification (YOLOv2 + 8.3.3); the second one corresponds to the results of the baseline method [32], our framework without considering the personalized non-maximum suppression procedure (YOLOv2 + 8.3.3), and our proposed framework; and the third row is similar to the second one, but replacing the segmentation method by the ground truth segmentation. As for the results achieved, it should be highlighted that our proposal outperforms the food recognition, with respect to the state-of-the-art method (Ciocca et al. [32]) in a 12.4% for *Recall* and 20.9% for *MAA*. Regarding *TA*, a decrease of 2.5% is observed. However, we consider that this measure does not reflect how well the recognition works mainly due to the imbalance in the quantity of food in the trays, which varies between 1 and 9 (see Fig. 8.6 (a)), as well as because *TA* measures the amount of food trays in which all positive samples have been correctly predicted, but does not penalize when there are false positives.

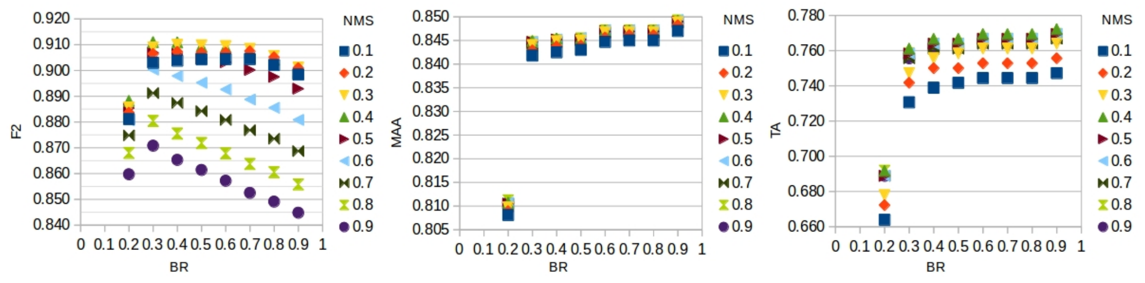


FIGURE 8.7: Results of our proposal when varying the background removal (BR) and non-maximum-suppression (NMS) thresholds.

In order to apply a complete comparison, we also replicated the evaluation proposed by [32], in which the authors considered a perfect segmentation using the ground truth (GT) and applied their detection method (bottom section of Table 8.3). In our case, there is no significant improvement with respect to the use of the proposed semantic segmentation, because our proposal considers the integration of the extracted information with the segmentation to refine the predictions already obtained by the object detection method. In contrast, Ciocca et al. [32] performed the recognition directly on the segmented objects. Comparing to the results obtained in [32], we can see that their method improves significantly in terms of *Recall* using the GT for segmentation, achieving to match our results. However in terms of *MAA*, despite improving its performance, our results are still about 16% better. A low *MAA* with a high *Recall* implies that the classifier has a strong bias towards the classes that have a greater amount of instances. Therefore, even if we consider a perfect segmentation to contrast the results, our proposal keeps a better performance for recognition and a lower bias towards the dominant classes.

The results obtained with the proposed approach based on the number of objects to be classified per food tray is shown in Fig. 8.6 (b). As expected, the *TA* measure tends to decrease as the number of objects increases, however there is no clear trend for the *Recall*. One of the lowest results in both measures is obtained in trays containing 6 foods, whereby we can determine that the errors correspond to 17 misclassified objects along 12 trays, that is, an average error of 1.42 objects per incorrectly classified tray. Despite having a low *TA* (0.478), the results are good considering the *Recall* obtained, since it is preferable to minimize the number of errors per tray if we think of a semi-automatic food billing system, in which the operator would make minor corrections if necessary.

When reviewing the overall mean of errors by misclassified trays, we can see that our classifier has an average of 1.09 errors along 85 trays classified incorrectly, compared to [32] that has an average of 3.33 errors along 76 trays classified incorrectly. That said, even though the baseline method achieves to completely classify 9 trays more than our proposal, due to its overall performance, the misclassified trays have about three times as many objects wrongly classified per tray.

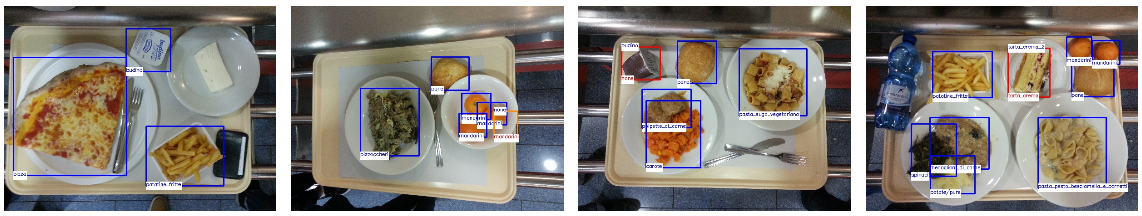


FIGURE 8.8: Some samples of the results obtained using the proposed approach, from left to right: food tray with all the objects correctly detected (blue), false detection sample (orange), and two samples with one misclassified object (red).

The results achieved with our approach consider a value of 0.5 for the thresholds used in both procedures, Background Removal (BR) and Non-Maximum Suppression (NMS). However, our approach achieves a good performance not only with a unique combination of values, but also with a wide range of them. Specifically, for the problem at hand, we can obtain results close to the ones described with any value in the ranges [0.3-0.6] for BR and [0.3-0.5] for NMS thresholds (see Fig. 8.7). The flexibility of choosing threshold values in a wide range suggests that our approach is robust with respect to its parameters. Furthermore, considering the F2 score, any value of the parameters for our proposed method produces better results than YOLOv2 + III-C1.

Finally, some examples of the results obtained by means of our proposed Semantic Food Detection method are shown in Fig. 8.8. In general terms, the classifier achieves a good performance in a variety of food items, where the main difficulties encountered are due to the following issues: 1) unlabeled food items, because they are not part of the 65 classes (eg. fresh cheese) or because they are not belonging to the same tray and that have been recognized by our algorithm; 2) the same food items placed very close (eg. mandarine); 3) foods ignored because they are not clearly distinguishable whether correspond to a meal or not (eg. pudding); and 4) confusions with classes corresponding to different kinds of cakes (eg. torta_cream), meats, pastas, among others.

8.5 Conclusions

We present a novel system that performs Semantic Food Detection applied to the problem of food tray analysis in self-service restaurants. More precisely, we integrate both techniques, food/non-food semantic segmentation with food detection, through the application of two procedures: a probabilistic procedure that allow us remove the background detections, and a custom non-maximum suppression procedure to avoid the occurrence of duplicate detections.

Regarding the architecture, we deal with the problem at hand using two pathways in parallel for food detection and semantic segmentation. The purpose of applying this separate computation is to take advantage of the benefits of each method separately to later combine them. In this manner, they do not condition each other, but reinforce themselves. In particular, if we propose an end-to-end architecture which directly feeds the segmentation output into the detection, the segmentation errors could not be recovered and, therefore, they could negatively influence the detection performance.

As for the results, our proposal significantly outperforms the state-of-the-art in terms of recall and mean average accuracy. Furthermore, our model is less sensitive to class imbalance and the mean of errors per foods placed on a tray is about 1, when the classifier is not able to recognize the whole tray well. The latter is quite relevant if our approach is applied in a semi-automatic billing system, in which the cashier would have to make only small changes to generate the final bill, and in this way to streamline the process involved in a self-service restaurant of *grab a meal, pay, and eat*. Furthermore, our proposed approach takes less than 0.5 seconds to predict all foods present in a image, considering the use of a personal computer with a low performance GPU (GeForce 940MX).

Our future research is focused on semantic detection of food ingredients and completely automating the self-service billing by integrating the restaurant menu by geolocalization. Moreover, it is interesting to explore if modeling the uncertainty for the bounding box location and food segmentation will minimize possible false positives and provide more reliable predictions.

Chapter 9

Conclusions and Future Lines

In this chapter we provide the concluding remarks derived from the work carried out in this thesis and future lines of research.

There is no doubt that deep learning algorithms have revolutionized the industry with computer vision solutions, feasible under real-life conditions, that were previously unimaginable. Food analysis is one of the fields benefited by the appearance of this type of machine learning algorithms. However, the complexity of the food itself requires more effort to provide robust and accurate algorithms.

In this thesis, we addressed the problem of visual food analysis, breaking it down into some of the most relevant tasks required to provide an automatic food analysis system. After an extensive review of the literature, we designed solutions for food detection, food recognition, and semantic food detection, and introduced uncertainty analysis into our proposed food recognition frameworks. For all the proposals, we decided as a common training step to transfer the learning from the deep learning models previously trained in the large scale ImageNet dataset to our domain. Even though there are not too many class of foods within ImageNet, the knowledge gained by the pretrained model regarding low or medium level features (e.g. edges, corners, colors) was helpful in guiding the learning of our models on different food datasets avoiding early overfitting and obtaining better results than training them from scratch. In all the food analysis tasks described above, we contrasted and improved the results. For this, in food detection, we proposed a more elaborate method of classifying rather than simply using the softmax classifier. For food recognition, uncertainty-aware methods and a fuzzy similarity measures to ensemble the CNN models were proposed. And for semantic food detection, we integrated semantic segmentation and object detection approaches into a custom framework.

Specifically, in the first part of our work, we contributed by deepening uncertainty analysis in many ways, resulting in novel uncertainty-based food recognition approaches. First, we proposed a new model for MTL that expanded the aleatoric (homocesdatisc) uncertainty-based method proposed in [83], to enable its application in SL and ML image classification tasks, which also incorporated a MTL regularization term into the loss function to improve coherence of the outputs from different food-related task. Additionally, we designed a new uncertainty-based hybrid food recognition method that considers a criterion based on quantifying uncertainty to make the final prediction through a LCPN or flat approach, which takes advantage of the benefits of hierarchical approaches and minimizes the propagation of parent-to-child errors frequently present on them. Moreover, we proposed three uncertainty-aware methods to perform data augmentation according to class-level uncertainty or sample-level uncertainty. For the method based on class-level uncertainty, we shown through the proposed frameworks that the incorporation of specific data augmentation techniques taken into account the class to which the image belongs leads to better learning by the model. Furthermore, regarding methods based on sample-level uncertainty, we demonstrated the importance of increasing the data of the training set with synthetic images generated from those difficult to classify in order to ease the learning of its discriminatory features. We could also observe with this approach, that it was possible to minimize the bias that occurs on the dominant classes in unbalanced databases.

As for the second part we contributed in three different tasks within visual food analysis. In the food detection case, we overcomes the SoA in two public dataset, FCD and RagusaDS, through a model that considers GoogLeNet for feature extraction, PCA for feature selection and SVM for classification. As for food recognition task, we proposed a CNNs fusion approach based on the concepts of decision templates and decision profiles and their similarity that improves the classification performance on Food-11 (more than 10%) and Food-101. Regarding the semantic food detection task, we significantly outperforms the SoA in terms of recall and average accuracy in UNIMIB2016 dataset by means of a novel system that integrate both techniques, food/non-food semantic segmentation with food detection, through the application of two procedures: a probabilistic procedure that allow us remove the background detections, and a custom non-maximum suppression procedure to avoid the occurrence of duplicate detections.

In addition to the experiments, during this thesis we contributed to building a large international food data set with more than 500,000 images, 650 classes and annotations of multiple food-related attributes (single label and multi-label). A subset of this dataset, named MAFood-121, became available to the community and was published in [3] (`www.ub.edu/cvub/mafood-121/`).

Although we have made important contributions in the field of visual food analysis, there is a wide range of possible problems to be resolved. In the following section we discuss some ideas and research lines that may become future studies within this field.

9.1 Future works

There are several ways to improve the visual analysis of foods that can extend the work of this thesis or on sub-tasks (new or recent) that were not addressed here, such as:

- **Combo-plate food items recognition:** Although most works consider recognizing a single class from food images, in some cases a particular served dish may comprise more than one food. This is the combo-plate case, where ML food recognition is required. Recipe extraction and ingredient recognition are two ML task widely researched. However with theses tasks, we known whether the image contains a single plate with mixed items (e.g. sushi) or a combo plate (e.g. salmon, rice and avocado serviced in a dish). Combo-plate food items recognition is a challenging new field of research that can help to better description of food images.

- **Integrating GNN and CNN for food recognition:** The incorporation of the relationship of the different food concepts within the learning process has not been studied too much. In this thesis we considered it into our method as a straightforward regularization term (see Chapter 3). We suggest that the integration of knowledge of food ontology with a powerful method such as Graph neural network (GNN), with the features extracted from visual content through CNN, may be beneficial for the final classification.

- **Model uncertainty into GAN methods:** Generative adversarial networks have shown to be efficient to generate synthetic data conserving the visual content from the source images, but increasing the variability of the data. Theses characteristics have conducing their application in the data augmentation process. We shown, in Chapter 5, the importance of incorporating uncertain images to improve the performance of food recognition algorithms. Therefore, a possible improvement would be to directly model the uncertainty in the GAN models to generate good quality and uncertain images instead of selecting rich images after generating them.

- **Volume or portion estimation:** Monitor food consumption in people who are overweight or obese is one of the most attractive applications based on visual food analysis. Here, it is very important to take a correct balance on calorie intake to prevent the development of chronic disease related to poor diet. Recognizing a food is not enough to provide an exact value on the calories involved, but it is also very important to know the volume or the portion. Therefore, volume or portion estimation is a very recent and useful task that needs more effort to be applicable in real life.

Finally, a cross-sectional future work is to minimize dependency and the time spent by the user to capture food images. We propose to evaluate and extend the visual food analysis methods on images acquired from wearable cameras, which is clearly a more complex and challenging source of data.

Bibliography

[1] E. Aguilar, M. Bolaños, and P. Radeva, "Food recognition using fusion of classifiers based on cnns", in *International Conference on Image Analysis and Processing*, Springer, 2017, pp. 213–224.

[2] ——, "Exploring food detection using cnns", in *EUROCAST 2017*, 2018, pp. 339–347.

[3] ——, "Regularized uncertainty-based multi-task learning model for food analysis", *Journal of Visual Communication and Image Representation*, vol. 60, pp. 360–370, 2019.

[4] E. Aguilar and P. Radeva, "Class-conditional data augmentation applied to image classification", in *International Conference on Computer Analysis of Images and Patterns*, Springer, 2019, pp. 182–192.

[5] ——, "Food recognition by integrating local and flat classifiers", in *Iberian Conference on Pattern Recognition and Image Analysis*, Springer, 2019, pp. 65–74.

[6] E. Aguilar, B. Remeseiro, M. Bolaños, and P. Radeva, "Grab, pay, and eat: Semantic food detection for smart restaurants", *IEEE Transactions on Multimedia*, vol. 20, no. 12, pp. 3266–3275, 2018.

[7] E. Aguilar., B. Nagarajan., R. Khatun., M. Bolaños., and P. Radeva., "Uncertainty modeling and deep learning applied to food image analysis", in *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 3 BIOINFORMATICS: BIOSTEC*, 2020, pp. 9–16.

[8] K. Aizawa, Y. Maruyama, H. Li, and C. Morikawa, "Food balance estimation by using personal dietary tendencies in a multimedia food log", *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 2176–2185, 2013.

[9] A. Ali-Gombe and E. Elyan, "Mfc-gan: Class-imbalanced dataset classification using multiple fake class generative adversarial network", *Neurocomputing*, vol. 361, pp. 212–221, 2019.

[10] I. U. N. Alliance, "National adult nutrition survey", *Public Health*, 2019.

[11] M. M. Anthimopoulos, L. Gianola, L. Scarnato, P. Diem, and S. G. Mougiakakou, "A food recognition system for diabetic patients based on an optimized bag-of-features model", *IEEE journal of biomedical and health informatics*, vol. 18, no. 4, pp. 1261–1271, 2014.

[12] M. Anzawa, S. Amano, Y. Yamakata, K. Motonaga, A. Kamei, and K. Aizawa, "Recognition of multiple food items in a single photo for use in a buffet-style restaurant", *IEICE TRANSACTIONS on Information and Systems*, vol. 102, no. 2, pp. 410–414, 2019.

[13] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.

[14] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, 2017.

[15] O. Beijbom, N. Joshi, D. Morris, S. Saponas, and S. Khullar, "Menu-match: Restaurant-specific food logging from images", in *IEEE Winter Conf. on Appl. of Comput. Vision*, 2015, pp. 844–851.

[16] W. H. Beluch, T. Genewein, A. Nürnberger, and J. M. Köhler, "The power of ensembles for active learning in image classification", in *Proc. of the IEEE Conf. on CVPR*, 2018, pp. 9368–9377.

[17] L. Bertoni, S. Kreiss, and A. Alahi, "Monoloco: Monocular 3d pedestrian localization and uncertainty estimation", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6861–6871.

[18] V. Bettadapura, E. Thomaz, A. Parnami, G. D. Abowd, and I. Essa, "Leveraging context to support automated food recognition in restaurants", in *IEEE Winter Conf. on Appl. of Comput. Vision*, 2015, pp. 580–587.

[19] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network", in *ICML*, 2015, pp. 1613–1622.

[20] M. Bolaños, A. Ferrà, and P. Radeva, "Food ingredients recognition through multi-label learning", in *International Conference on Image Analysis and Processing*, Springer, 2017, pp. 394–402.

[21] M. Bolaños and P. Radeva, "Simultaneous food localization and recognition", in *2016 23rd International Conference on Pattern Recognition (ICPR)*, IEEE, 2016, pp. 3140–3145.

[22] M. Bosch, F. Zhu, N. Khanna, C. J. Boushey, and E. J. Delp, "Combining global and local features for food identification in dietary assessment", in *2011 18th IEEE International Conference on Image Processing*, IEEE, 2011, pp. 1789–1792.

[23] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests", in *European conference on computer vision*, Springer, 2014, pp. 446–461.

[24] V. Bruno and C. J. Silva Resende, "A survey on automated food monitoring and dietary management systems", *Journal of health & medical informatics*, vol. 8, no. 3, 2017.

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique", *J. of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[26] J. Chen, C. Ngo, and T. Chua, "Cross-modal recipe retrieval with rich food attributes", in *ACM Multimedia Conf.*, 2017, pp. 1771–1779.

[27] J. Chen and C.-W. Ngo, "Deep-based ingredient recognition for cooking recipe retrieval", in *Proceedings of the 2016 ACM on Multimedia Conference*, ACM, 2016, pp. 32–41.

[28] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs", *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[29] M. Chen, K. Dhingra, W. Wu, L. Yang, R. Sukthankar, and J. Yang, "Pfid: Pittsburgh fast-food image dataset", in *2009 16th IEEE International Conference on Image Processing (ICIP)*, IEEE, 2009, pp. 289–292.

[30] J. Choi, D. Chun, H. Kim, and H.-J. Lee, "Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 502–511.

[31] G. Ciocca, P. Napoletano, and R. Schettini, "Food recognition and leftover estimation for daily diet monitoring", in *Int. Conf. on Image Analysis and Processing*, 2015, pp. 334–341.

[32] ——, "Food recognition: A new dataset, experiments and results", *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 3, pp. 588–598, 2017. DOI: `10.1109/JBHI.2016.2636441`.

[33] ——, "Learning cnn-based features for retrieval of food images", in *International Conference on Image Analysis and Processing*, Springer, 2017, pp. 426–434.

[34] C. Cortes and V. Vapnik, "Support vector machine", *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[35] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data", in *Proc. of the IEEE Conf. on CVPR*, 2019, pp. 113–123.

[36] J. Dehais, M. Anthimopoulos, S. Shevchik, and S. Mougiakakou, "Two-view 3D Reconstruction for Food Volume Estimation", *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 1090–1099, 2017.

[37] A. P. Dempster, "A generalization of bayesian inference", *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 30, no. 2, pp. 205–232, 1968.

[38] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 8, pp. 800–810, 2001.

[39] R. Dinic, M. Domhardt, S. Ginzinger, and T. Stütz, "Eatar tango: Portion estimation on mobile devices with a depth sensor", in *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*, ACM, 2017, p. 46.

[40] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving", in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 2095–2104.

[41] I. Donadello and M. Dragoni, "Ontology-driven food category classification in images", in *International Conference on Image Analysis and Processing (2)*, ser. Lecture Notes in Computer Science, vol. 11752, Springer, 2019, pp. 607–617.

[42] T. Ege and K. Yanai, "Simultaneous estimation of food categories and calories with multi-task cnn", in *Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on*, IEEE, 2017, pp. 198–201.

[43] C. F. El Khoury, M. Karavetian, R. J. Halfens, R. Crutzen, L. Khoja, and J. M. Schols, "The effects of dietary mobile apps on nutritional outcomes in adults with chronic diseases: A systematic review", *Journal of the Academy of Nutrition and Dietetics*, 2019.

[44] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge", *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[45] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1625–1634.

[46] G. M. Farinella, D. Allegra, M. Moltisanti, F. Stanco, and S. Battiato, "Retrieval and classification of food images", *Computers in biology and medicine*, vol. 77, pp. 23–39, 2016.

[47] G. M. Farinella, D. Allegra, and F. Stanco, "A benchmark dataset to study the representation of food images", in *European Conference on Computer Vision*, Springer, 2014, pp. 584–599.

[48] G. M. Farinella, D. Allegra, F. Stanco, and S. Battiato, "On the exploitation of one class classification to distinguish food vs non-food images", in *International Conference on Image Analysis and Processing*, Springer, 2015, pp. 375–383.

[49] P. F. Felzenszwalb, R. B. Girshick, D. Mcallester, and D. Ramanan, "Object Detection with Discriminatively Trained Part Based Models", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1–20, 2009.

[50] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”, in *international conference on machine learning*, 2016, pp. 1050–1059.

[51] Y. Gal, R. Islam, and Z. Ghahramani, “Deep bayesian active learning with image data”, in *Proc. of the 34th ICML-Volume 70*, JMLR. org, 2017, pp. 1183–1192.

[52] Y. Gao, Q. She, J. Ma, M. Zhao, W. Liu, and A. Yuille, *Nddr-cnn: Layer-wise feature fusing in multi-task cnn by neural discriminative dimensionality reduction. corr abs/1801.0 (2018)*, 2018.

[53] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, “Digital Image Processing Using MATLAB”, *Pearson Prentice Hall*, 2004.

[54] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples”, in *International Conference on Learning Representations*, 2015. [Online]. Available: `http://arxiv.org/abs/1412.6572`.

[55] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset”, 2007.

[56] C. Güngör, F. Baltacı, A. Erdem, and E. Erdem, “Turkish cuisine: A benchmark dataset with turkish meals for food recognition”, in *2017 25th Signal Processing and Communications Applications Conference (SIU)*, May 2017, pp. 1–4. DOI: `10.1109/SIU.2017.7960494`.

[57] F. Han, R. Guerrero, and V. Pavlovic, “Cookgan: Meal image synthesis from ingredients”, in *The IEEE WACV*, 2020, pp. 1450–1458.

[58] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Munari, M. Mordonini, and S. Cagnoni, “Food image recognition using very deep convolutional networks”, in *International Workshop on Multimedia Assisted Dietary Management*, 2016, pp. 41–49.

[59] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition”, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[60] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, “Bounding box regression with uncertainty for accurate object detection”, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2888–2897.

[61] L. Herranz, S. Jiang, and R. Xu, “Modeling Restaurant Context for Food Recognition”, *IEEE Trans. Multimedia*, vol. 19, no. 2, pp. 430–440, 2017.

[62] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network”, *arXiv preprint arXiv:1503.02531*, 2015.

[63] D. Hoiem, Y. Chodpathumwan, and Q. Dai, “Diagnosing error in object detectors”, *Eur. Conf. on Comput. Vision*, pp. 340–353, 2012.

[64] D. Horita, W. Shimoda, and K. Yanai, “Unseen food creation by mixing existing food images with conditional stylegan”, in *Proc. of MADiMa*, 2019, pp. 19–24.

[65] D. Horita, R. Tanno, W. Shimoda, and K. Yanai, "Food category transfer with conditional cyclegan and a large-scale food image dataset", in *Proc. of CEA/MADiMa*, 2018, pp. 67–70.

[66] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks", in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[67] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex", *The Journal of physiology*, vol. 160, no. 1, pp. 106–154, 1962.

[68] Y. Ito, W. Shimoda, and K. Yanai, "Food image generation using a large amount of food images with conditional gan: Ramengan and recipegan", in *Proc. of CEA/MADiMa*, 2018, pp. 71–74.

[69] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The One Hundred Layers Tiramisu: Fully convolutional DenseNets for Semantic Segmentation", in *CVPR Workshops*, 2017, pp. 1175–1183.

[70] 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 *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[71] S. Jiang, W. Min, L. Liu, and Z. Luo, "Multi-scale multi-view deep feature aggregation for food recognition", *IEEE Transactions on Image Processing*, vol. 29, pp. 265–276, 2019.

[72] I. T. Jolliffe, "Principal components in regression analysis", in *Principal component analysis*, Springer, 1986, pp. 129–155.

[73] T. Joutou and K. Yanai, "A food image recognition system with multiple kernel learning", in *2009 16th IEEE International Conference on Image Processing (ICIP)*, IEEE, 2009, pp. 285–288.

[74] A. Jsang, *Subjective Logic: A formalism for reasoning under uncertainty*. Springer Publishing Company, Incorporated, 2018.

[75] H. Kagaya and K. Aizawa, "Highly accurate food/non-food image classification based on a deep convolutional neural network", in *International conference on image analysis and processing*, Springer, 2015, pp. 350–357.

[76] H. Kagaya, K. Aizawa, and M. Ogawa, "Food detection and recognition using convolutional neural network", in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 1085–1088.

[77] H. F. Kaiser, "The application of electronic computers to factor analysis", *Educational and psychological measurement*, vol. 20, no. 1, pp. 141–151, 1960.

[78] G. Kang, X. Dong, L. Zheng, and Y. Yang, "Patchshuffle regularization", *arXiv preprint arXiv:1707.07103*, 2017.

[79] A. Karpathy, *Neural Networks Part 1: Setting up the Architecture*, Stanford University Course Notes. [Online]. Available: `https://cs231n.github.io/neural-networks-1/` (visited on 05/20/2020).

[80] Y. Kawano and K. Yanai, "Real-time mobile food recognition system", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 1–7.

[81] ——, "Automatic expansion of a food image dataset leveraging existing categories with domain adaptation", in *European Conference on Computer Vision*, Springer, 2014, pp. 3–17.

[82] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?", in *Advances in neural information processing systems*, 2017, pp. 5574–5584.

[83] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics", *arXiv preprint arXiv:1705.07115*, 2017.

[84] S. Khan, M. Hayat, S. W. Zamir, J. Shen, and L. Shao, "Striking the right balance with uncertainty", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 103–112.

[85] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[86] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick", in *Advances in neural information processing systems*, 2015, pp. 2575–2583.

[87] K. Kitamura, T. Yamasaki, and K. Aizawa, "Foodlog: Capture, analysis and retrieval of personal food images via web", in *Proceedings of the ACM multimedia 2009 workshop on Multimedia for cooking and eating activities*, 2009, pp. 23–30.

[88] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers", *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[89] S. Kornblith, J. Shlens, and Q. V. Le, "Do better imagenet models transfer better?", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2661–2671.

[90] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images", Citeseer, Tech. Rep., 2009.

[91] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[92] L. Kuncheva, R. Kounchev, and R. Zlatev, "Aggregation of multiple classification decisions by fuzzy templates", in *Proceedings of the Third European Congress on Intelligent Technologies and Soft Computing EUFIT*, vol. 95, 1995, pp. 1470–1474.

[93] L. I. Kuncheva, J. C. Bezdek, and R. P. Duin, "Decision templates for multiple classifier fusion: An experimental comparison", *Pattern recognition*, vol. 34, no. 2, pp. 299–314, 2001.

[94] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles", in *Advances in Neural Information Processing Systems*, 2017, pp. 6402–6413.

[95] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network", in *Advances in neural information processing systems*, 1990, pp. 396–404.

[96] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[97] K.-H. Lee, X. He, L. Zhang, and L. Yang, "Cleannet: Transfer learning for scalable image classifier training with label noise", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5447–5456.

[98] S. Li, Z.-Q. Liu, and A. B. Chan, "Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 482–489.

[99] X. Li, F. Zhao, and Y. Guo, "Conditional restricted boltzmann machines for multi-label learning with incomplete labels", in *Artificial Intelligence and Statistics*, 2015, pp. 635–643.

[100] Y. Li, Y. Song, and J. Luo, "Improving pairwise ranking for multi-label image classification", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[101] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast autoaugment", in *NIPS*, 2019, pp. 6662–6672.

[102] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation", in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2017, pp. 1925–1934.

[103] M. Lin, Q. Chen, and S. Yan, "Network in network", *arXiv preprint arXiv:1312.4400*, 2013.

[104] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, and Y. Ma, "Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment", in *International Conference on Smart Homes and Health Telematics*, Springer, 2016, pp. 37–48.

[105] W. Liu and I. W. Tsang, "Large margin metric learning for multi-label prediction.", in *AAAI*, vol. 15, 2015, pp. 2800–2806.

[106] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks.", in *ICML*, vol. 2, 2016, p. 7.

[107] C. Louizos and M. Welling, "Multiplicative normalizing flows for variational bayesian neural networks", in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 2218–2227.

[108] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[109] C. Luo, X. Li, J. Yin, J. He, D. Gao, and J. Zhou, "How does the data set and the number of categories affect cnn-based image classification performance?", *Journal of Software*, vol. 14, no. 4, pp. 168–181, 2019.

[110] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, "Bagan: Data augmentation with balancing gan", *arXiv preprint arXiv:1803.09655*, 2018.

[111] N. Martinel, G. L. Foresti, and C. Micheloni, "Wide-slice residual networks for food recognition", in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2018, pp. 567–576.

[112] Y. Matsuda, H. Hoashi, and K. Yanai, "Recognition of multiple-food images by detecting candidate regions", in *2012 IEEE International Conference on Multimedia and Expo*, IEEE, 2012, pp. 25–30.

[113] C. Mayer and R. Timofte, "Adversarial sampling for active learning", in *The IEEE WACV*, 2020, pp. 3071–3079.

[114] A. Meyers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. P. Murphy, "Im2calories: Towards an automated mobile vision food diary", in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1233–1241.

[115] S. Mezgec and B. Koroušić Seljak, "Nutrinet: A deep learning food and drink image recognition system for dietary assessment", *Nutrients*, vol. 9, no. 7, p. 657, 2017.

[116] S. Mezgec and B. K. Seljak, "Using deep learning for food and beverage image recognition", in *2019 IEEE Int. Conf. on Big Data (Big Data)*, IEEE, 2019, pp. 5149–5151.

[117] W. Min, B. Bao, S. Mei, Y. Zhu, Y. Rui, and S. Jiang, "You are what you eat: Exploring rich recipe information for cross-region food analysis", *IEEE Trans. Multimedia*, 2017.

[118] W. Min, S. Jiang, J. Sang, H. Wang, X. Liu, and L. Herranz, “Being a supercook: Joint food attributes and multimodal content modeling for recipe retrieval and exploration”, *IEEE Transactions on Multimedia*, vol. 19, no. 5, pp. 1100–1113, 2017.

[119] W. Min, L. Liu, Z. Luo, and S. Jiang, “Ingredient-guided cascaded multi-a ention network for food recognition”, 2019.

[120] Z.-Y. Ming, J. Chen, Y. Cao, C. Forde, C.-W. Ngo, and T. S. Chua, “Food photo recognition for dietary tracking: System and experiment”, in *International Conference on Multimedia Modeling*, Springer, 2018, pp. 129–141.

[121] D. Molchanov, A. Ashukha, and D. Vetrov, “Variational dropout sparsifies deep neural networks”, in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 2498–2507.

[122] F. J. Moreno-Barea, F. Strazzera, J. M. Jerez, D. Urda, and L. Franco, “Forward noise adjustment scheme for data augmentation”, in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2018, pp. 728–734.

[123] V. Mullachery, A. Khera, and A. Husain, “Bayesian neural networks”, *arXiv preprint arXiv:1801.07710*, 2018.

[124] N. Nag, V. Pandey, and R. Jain, “Health multimedia: Lifestyle recommendations based on diverse observations”, in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, ACM, 2017, pp. 99–106.

[125] M. Ng, T. Fleming, M. Robinson, B. Thomson, N. Graetz, C. Margono, E. C. Mullany, S. Biryukov, C. Abbafati, S. F. Abera, *et al.*, “Global, regional, and national prevalence of overweight and obesity in children and adults during 1980–2013: A systematic analysis for the global burden of disease study 2013”, *The lancet*, vol. 384, no. 9945, pp. 766–781, 2014.

[126] C. Nielsen and M. Okoniewski, “Gan data augmentation through active learning inspired sample acquisition”, in *Proceedings of the IEEE conference on computer vision and pattern recognition Workshops*, 2019, pp. 109–112.

[127] W. H. Organization, *Diet, nutrition, and the prevention of chronic diseases: report of a joint WHO/FAO expert consultation*. World Health Organization, 2003, vol. 916.

[128] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, “Scikit-learn: Machine learning in python”, *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[129] J. C. Peterson, R. M. Battleday, T. L. Griffiths, and O. Russakovsky, “Human uncertainty makes classification more robust”, in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9617–9626.

[130] F. Ragusa, V. Tomaselli, A. Furnari, S. Battiato, and G. M. Farinella, “Food vs non-food classification”, in *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*, 2016, pp. 77–81.

[131] G. Raimato, "The Design of a Smart Tray with Its Canteen Users: A Formative Study", in *Int. Conf. in Methodologies and Intelligent Systems for Technology Enhanced Learning*, vol. 617, 2017, p. 36.

[132] D. Ravi, C. Wong, B. Lo, and G.-Z. Yang, "Real-time food intake classification and energy expenditure estimation on a mobile device", in *2015 IEEE 12th Int. Conf. on Wearable and Implantable Body Sensor Networks (BSN)*, IEEE, 2015, pp. 1–6.

[133] J. Redmon, *Darknet: Open Source Neural Networks in C*, `http://pjreddie.com/darknet/`, 2016.

[134] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2016, pp. 779–788.

[135] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger", in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2017, pp. 6517–6525.

[136] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows", *arXiv preprint arXiv:1505.05770*, 2015.

[137] H. Robbins and S. Monro, "A stochastic approximation method", *The annals of mathematical statistics*, pp. 400–407, 1951.

[138] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation", in *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[139] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.", *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[140] W. Rumpler, M. Kramer, D. Rhodes, A. Moshfegh, and D. Paul, "Identifying sources of reporting error using measured food intake", *European journal of clinical nutrition*, vol. 62, no. 4, pp. 544–552, 2008.

[141] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge", *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[142] D. Sahoo, W. Hao, S. Ke, W. Xiongwei, H. Le, P. Achananuparp, E.-P. Lim, and S. C. Hoi, "Foodai: Food image recognition via deep learning for smart food logging", 2019.

[143] S. Sajadmanesh, S. Jafarzadeh, S. A. Ossia, H. R. Rabiee, H. Haddadi, Y. Mejova, M. Musolesi, E. D. Cristofaro, and G. Stringhini, "Kissing cuisines: Exploring worldwide culinary habits on the web", in *Proceedings of the 26th International Conference on World Wide Web Companion*, International World Wide Web Conferences Steering Committee, 2017, pp. 1013–1021.

[144] C. Sakaridis, D. Dai, and L. V. Gool, "Guided curriculum model adaptation and uncertainty-aware evaluation for semantic nighttime image segmentation", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7374–7383.

[145] A. Salvador, N. Hynes, Y. Aytar, J. Marin, F. Ofli, I. Weber, and A. Torralba, "Learning cross-modal embeddings for cooking recipes and food images", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[146] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty", in *Advances in Neural Information Processing Systems*, 2018, pp. 3179–3189.

[147] T. R. Shaham, T. Dekel, and T. Michaeli, "Singan: Learning a generative model from a single natural image", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4570–4580.

[148] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, 2017.

[149] J.-S. Shim, K. Oh, and H. C. Kim, "Dietary assessment methods in epidemiologic studies", *Epidemiology and health*, vol. 36, 2014.

[150] W. Shimoda and K. Yanai, "CNN-based food image segmentation without pixel-wise annotation", in *Int. Conf. on Image Analysis and Processing*, 2015, pp. 449–457.

[151] ——, "Foodness Proposal for Multiple Food Detection by Training of Single Food Images", in *Int. Workshop on Multimedia Assisted Dietary Management*, 2016, pp. 13–21.

[152] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains", *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[153] A. Singla, L. Yuan, and T. Ebrahimi, "Food/non-food image classification and food categorization using pre-trained googlenet model", in *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*, ACM, 2016, pp. 3–11.

[154] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[155] H. Su, T.-W. Lin, C.-T. Li, M.-K. Shan, and J. Chang, "Automatic recipe cuisine classification by ingredients", in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, ACM, 2014, pp. 565–570.

[156] M. Subedar, R. Krishnan, P. L. Meyer, O. Tickoo, and J. Huang, "Uncertainty-aware audiovisual activity recognition using deep bayesian variational inference", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6301–6310.

[157] M. A. Subhi, S. H. Ali, and M. A. Mohammed, "Vision-based approaches for automatic food recognition and dietary assessment: A survey", *IEEE Access*, vol. 7, pp. 35 370–35 381, 2019.

[158] C. Summers and M. J. Dinneen, "Improved mixed-example data augmentation", in *2019 IEEE WACV*, IEEE, 2019, pp. 1262–1270.

[159] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[160] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[161] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks", in *International Conference on Machine Learning*, 2019, pp. 6105–6114.

[162] R. Tanno, K. Okamoto, and K. Yanai, "Deepfoodcam: A dcnn-based real-time mobile food recognition system", in *Proceedings of the 2nd International Workshop on MADiMa*, ACM, 2016, pp. 89–89.

[163] R. E. Uhrig, "Introduction to artificial neural networks", in *Proceedings of IECON'95-21st Annual Conference on IEEE Industrial Electronics*, IEEE, vol. 1, 1995, pp. 33–37.

[164] G. Waltner, M. Schwarz, S. Ladstätter, A. Weber, P. Luley, M. Lindschinger, I. Schmid, W. Scheitz, H. Bischof, and L. Paletta, "Personalized Dietary Self-Management using Mobile Vision-based Assistance", in *Int. Workshop on Multimedia Assisted Dietary Management*, 2017.

[165] J. Wang and L. Perez, "The effectiveness of data augmentation in image classification using deep learning", *Convolutional Neural Networks Vis. Recognit*, p. 11, 2017.

[166] Y. Wang, J.-j. Chen, C.-W. Ngo, T.-S. Chua, W. Zuo, and Z. Ming, "Mixed dish recognition through multi-label learning", in *Proceedings of the 11th Workshop on Multimedia for Cooking and Eating Activities*, ser. CEA '19, Ottawa ON, Canada: Association for Computing Machinery, 2019, pp. 1–8, ISBN: 9781450367790. DOI: 10.1145/3326458.3326929. [Online]. Available: https://doi.org/10.1145/3326458.3326929.

[167] A. Waxman, "Who global strategy on diet, physical activity and health", *Food and nutrition bulletin*, vol. 25, no. 3, pp. 292–302, 2004.

[168] A. Waxman and K. R. Norum, "Why a global strategy on diet, physical activity and health? the growing burden of non-communicable diseases", *Public Health Nutrition*, vol. 7, no. 3, p. 381, 2004.

[169] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning", in *Advances in Neural Information Processing Systems*, 2017, pp. 4148–4158.

[170] H. Wu, M. Merler, R. Uceda-Sosa, and J. R. Smith, "Learning to make better mistakes: Semantics-aware visual food recognition", in *Proceedings of the 24th ACM international conference on Multimedia*, ACM, 2016, pp. 172–176.

[171] Y. Xia, F. Liu, D. Yang, J. Cai, L. Yu, Z. Zhu, D. Xu, A. Yuille, and H. Roth, "3d semi-supervised learning with uncertainty-aware multi-view co-training", in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 3646–3655.

[172] R. Xu, L. Herranz, S. Jiang, S. Wang, X. Song, and R. Jain, "Geolocalized modeling for dish recognition", *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1187–1199, 2015.

[173] K. Yanai and Y. Kawano, "Food image recognition using deep convolutional network with pre-training and fine-tuning", in *2015 IEEE International Conference on Multimedia And Expo Workshops (ICMEW)*, IEEE, 2015, pp. 1–6.

[174] H. Yang, J. T. Zhou, and J. Cai, "Improving multi-label learning with missing labels by structured semantic correlations", in *European conference on computer vision*, Springer, 2016, pp. 835–851.

[175] S. Yang, M. Chen, D. Pomerleau, and R. Sukthankar, "Food recognition using statistics of pairwise local features", in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, 2010, pp. 2249–2256.

[176] R. Yasarla and V. M. Patel, "Uncertainty guided multi-scale residual learning-using a cycle spinning cnn for single image de-raining", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8405–8414.

[177] B. Yegnanarayana, *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[178] J. Yim, H. Jung, B. Yoo, C. Choi, D. Park, and J. Kim, "Rotating your face using multi-task deep neural network", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 676–684.

[179] X. Yin and X. Liu, "Multi-task convolutional neural network for pose-invariant face recognition", *IEEE Transactions on Image Processing*, 2017.

[180] T. Yu, D. Li, Y. Yang, T. M. Hospedales, and T. Xiang, "Robust person re-identification by modelling feature uncertainty", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 552–561.

[181] S. Zagoruyko and N. Komodakis, "Wide residual networks", in *Proceedings of the British Machine Vision Conference (BMVC)*, 2016, pp. 87.1–87.12.

[182] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks", in *European conference on computer vision*, Springer, 2014, pp. 818–833.

[183] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan++: Realistic image synthesis with stacked generative adversarial networks", *IEEE Trans. on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1947–1962, 2018.

[184] X.-J. Zhang, Y.-F. Lu, and S.-H. Zhang, "Multi-task learning for food identification and analysis with deep convolutional neural networks", *Journal of Computer Science and Technology*, vol. 31, no. 3, pp. 489–500, 2016.

[185] M. M. Zhang, "Identifying the cuisine of a plate of food", *University of California San Diego, Tech. Rep*, 2011.

[186] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms", *IEEE transactions on knowledge and data engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[187] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning", in *European Conference on Computer Vision*, Springer, 2014, pp. 94–108.

[188] Z. Zhang, A. Romero, M. J. Muckley, P. Vincent, L. Yang, and M. Drozdzal, "Reducing uncertainty in undersampled mri reconstruction with active acquisition", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2049–2058.

[189] H. Zhao, K.-H. Yap, A. C. Kot, and L. Duan, "Jdnet: A joint-learning distilled network for mobile visual food recognition", *IEEE J. of Selected Topics in Signal Processing*, 2020.

[190] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network", in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2017, pp. 2881–2890.

[191] J. Zheng, R. Yu, J.-C. Chen, B. Lu, C. D. Castillo, and R. Chellappa, "Uncertainty modeling of contextual-connections between tracklets for unconstrained video-based face recognition", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 703–712.

[192] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation", *arXiv preprint arXiv:1708.04896*, 2017.

[193] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[194] B. Zhu, C.-W. Ngo, J. Chen, and Y. Hao, "R2gan: Cross-modal recipe retrieval with generative adversarial network", in *Proc. of the IEEE Conf. on CVPR*, 2019, pp. 11 477–11 486.