March 2016

Food Recognition and Detection with Minimum Supervision

Renfeng Liu
The University of Western Ontario

Supervisor
Dr. Charles X. Ling
The University of Western Ontario

Graduate Program in Computer Science

A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

Abstract

Food recognition can help people keep track of and analyze their eating habits conveniently by snapping a photo on their smartphones. A lot of existing work has been published in food recognition using computer vision and deep learning techniques. However, most previous work assumed that one food image contains only one food item, thus can not handle images which contain multiple food items. In real-life scenarios, it is more common for a food image having more than one food item. Existing methods for multiple-food detection have various limitations: they either require certain kinds of additional user operations (such as drawing bounding boxes on food items) or are still in primitive stage with low accuracy rate. Current state-of-the-art object detection models usually utilize the power of convolutional neural networks (CNNs) which require a lot of training data with ground truth bounding boxes. Those limitations narrow down the usability of existing methods in real life.

To solve these problems, we propose a novel method which jointly detects food item combinations and their locations in the image with minimal supervision. We create a food image dataset with 10 categories for our experiments. In our dataset, images with only one food item are used as the training set while images with multiple food items are used as the test dataset. Our method does not require any ground truth bounding boxes in the training set.

At the training stage, we firstly use region proposal algorithms to generate candidate regions and extract the CNN features of all regions. Secondly, we perform region mining to select discriminative positive regions for each food category using maximum cover by submodular optimization. With these mined positive regions, we train a binary SVM classifier for each food category and further refine these classifiers with hard negative examples.

At the testing stage, we firstly generate a set of candidate regions. For each region, a classification score is computed based on its extracted CNN features. Then we select the regions using non-maximum suppression and output the locations and predicted food names of the selected regions.

Our experiments show very promising results. We obtain the average recall rate of 80.18% and precision rate of 83.78% on our test dataset. Our method achieve minimum supervision in the sense that it uses only “weakly labeled” images for training and does not require any ground truth bounding boxes for training dataset, thus can be easily extended to a larger dataset with more food categories.

Keywords: Food recognition, food detection, object localization, weakly supervised learning

Acknowlegements

I would like to express my sincere gratitude to my supervisor Prof. Charles Ling for the continuous support and guidance of my study and research, for his patience, motivation, enthusiasm, and immense knowledge. I could not have finished this thesis without the guidance of him.

I would also like to thank my colleagues here in our research group. Yan Luo, you gave me so many help both in life and study. Shuang Ao and Xiang Li, you gave me so many insightful suggestions. Chang Liu, you are such a good friend that can help in every aspect.

Finally, I would like to thank my parents, my sisters and my wife for their continuous love and support.

Contents

Certificate of Examination **i**

Abstract **i**

Acknowlegements **ii**

List of Figures **vi**

List of Tables **viii**

List of Appendices **ix**

1 Introduction **1**

1.1 Problem Statement 1

1.2 Thesis Motivation and Objectives 2

1.3 Thesis Contributions 4

1.4 Thesis Outline 6

2 Related Work **7**

2.1 Related Work in Food Image Recognition 7

2.1.1 Classification Using Classical Approach 8

2.1.2 Classification Using Deep Learning Approach 9

2.2 Related Work in Multiple-Food Recognition 10

2.3 Related Work in Multiple-Object Detection 12

2.3.1 Supervised Multiple-Object Detection 13

2.3.2 Weakly Supervised Multiple-Object Detection 14

3 Overview of Our Proposed Method **15**

4 Multiple-Food Detection **18**

4.1 Detection Region Proposals 18

4.1.1 Introduction of Detection Region Proposal Methods 19
4.1.2 SelectiveSearch Region Proposal 20
4.2 Region Representation 22
4.2.1 Convolutional Neural Network 22
4.2.2 Feature Extraction with Convolutional Neural Network 24
4.3 Region Mining 27
4.3.1 The Goal of Region Mining 27
4.3.2 Maximum-Cover Region Selection 27
Maximum-Cover by Submodular Optimization 29
4.4 Classifier Training 32
4.4.1 SVM Optimization 32
The Loss Function 33
Optimization of SVM Loss Function 34
4.4.2 Hard Negatives Mining 37
4.5 Non-Maximum Suppression 38
4.5.1 Greedy Non-Maximum Suppression 39
4.5.2 Non-Maximum Suppression on Top-N Regions 39

5 Experiments and Results **41**
5.1 The Dataset 41
5.2 Detector Evaluation Criterion 43
5.3 Experiment Results 44
5.3.1 Comparison Experiment 47
5.4 Discussion 50
5.4.1 Discussion on the Proposed Processing Pipeline 50
5.4.2 Discussion on Inaccurate Cases 51
5.4.3 Discussion on Methods Comparison 51
5.4.4 Discussion on Potential Improvements 51

6 Conclusions and Future Work **53**
6.1 Conclusions 53
6.2 Future Work 54

Bibliography **55**

A More Detection Results on Test Images **61**

Curriculum Vitae **66**

List of Figures

1.1 Food images contain only one food item 2
1.2 Real life food images contain multiple food items 3

2.1 Discrimitive component mining using Random Forest processing flow [7] . . . 8
2.2 Multiple food recognition processing flow of Matsuda et al.'s work [41] 11
2.3 FoodCam system process flow 12
2.4 Overview of RCNN 13

3.1 Overview of the training pipeline of Multiple-Food detection 15
3.2 Overview of the testing pipeline of Multiple-Food detection 16
3.3 Sample detection results of our proposed method on test images 17

4.1 Proposed regions computed by SelectiveSearch 21
4.2 A biological neuron (left) and its corresponding mathematical model (right). [1] . 22
4.3 Architecture of LeNet5 [2]. 23
4.4 Architecture of AlexNet [3] 24
4.5 Visualization of the activation of layers in AlexNet 25
4.6 Feature values of $fc7$ layer (4096 channels) 26
4.7 Regions with Maximum Cover 28
4.8 Submodular Set Function 29
4.9 The top-10 regions in set S for the training dataset of the burger category 31
4.10 Example SVM Hpyerplains 32
4.11 SVM Maximum-margin hyperplane 33
4.12 Examples of predictions without and with Non-Maximum Suppression 38
4.13 Examples of predictions with greedy version of Non-Maximum Suppression . . 39
4.14 Examples of predictions with improved Non-Maximum Suppression 40

5.1 Examples of training and testing images 42
5.2 Some successful detection results 45
5.3 Some inaccurate detection results 46
5.4 Overview of the training pipeline using whole image 47

5.5 Comparison of the recall and precision for different methods 49

A.1 Detection results on the test images (cont. 1) 61
A.2 Detection results on the test images (cont. 2) 62
A.3 Detection results on the test images (cont. 3) 63
A.4 Detection results on the test images (cont. 4) 64
A.5 Detection results on the test images (cont. 5) 65

List of Tables

5.1 The training dataset 41
5.2 The testing dataset 41
5.3 The recall rate and the precision rate on the test dataset with our proposed pipeline 44
5.4 Result of comparison experiment: recall and precision rate on the test dataset without region mining 48

List of Appendices

Appendix A 61

Chapter 1

Introduction

1.1 Problem Statement

With the abundance of food available, more and more people have dietary problems. People either eat too much more than what they need or eat unhealthily. Obesity is a direct consequence of dietary problems, which can cause many diseases such as diabetes and heart diseases. American Medical Association has officially announced obesity as a disease in 2013 [1]. A mobile app for keeping track of diet is becoming more important for people to keep healthy. Apps such as MyFitnessPal[2] and LoseIt[3] for this purpose are quite popular. These apps ask users to manually type keywords in the search box and select the food in the returned list. Usually it takes several steps to record just one food item which discourages constant uses of those apps.

With the prevalence of smartphone cameras, taking a photo of the food is an ideal method to conveniently log diet data. There has been existing work [43, 60, 61, 59, 42] using image-based food recognition techniques to analyze the food images and help record the nutrition facts. However, most existing approaches are either requiring user input or are still in primitive stage with low accuracy rate. Furthermore, although a few existing approaches [40, 42] addressed the problem of recognizing multiple food items, they assumed the availability of a lot of training data with ground truth bounding boxes. Most existing work assumed that one food image contained only one food item, thus could not handle images which contained two or more food items. Therefore, a method that can automatically classify multiple food items in a camera image taken under real-world conditions is essential for the success of a food recording system.

[1]http://www.ama-assn.org/ama/pub/news/news/2013/2013-06-18-new-ama-policies-annual-meeting.page
[2]https://www.myfitnesspal.com/
[3]https://www.loseit.com/

1.2 Thesis Motivation and Objectives

The problems in food recognition have not been fully addressed yet.

On one hand, most previous work assumed that one food image contained only one food item as reviewed in Chapter 2 (examples shown in Figure 1.1). However, in real-life scenarios, a food image of a meal plate is more common to include multiple food items (examples shown in Figure 1.2). Despite the importance of multiple food recognition, there are only four research papers on this problem as reviewed in Chapter 2 Section 2.2. Moreover, these four existing studies either need user interactions or require a lot of training data with ground truth bounding boxes of food items.

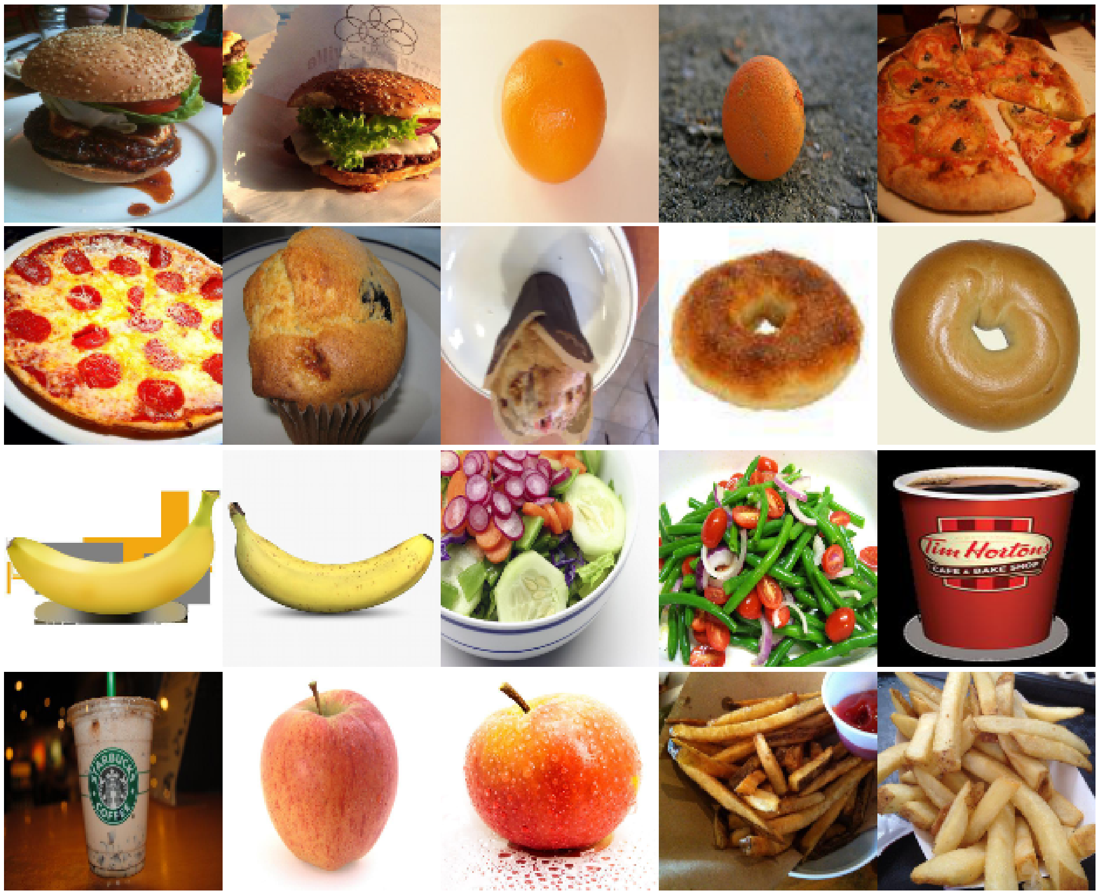


Figure 1.1: Food images contain only one food item

On the other hand, food images have high intra-class variations due to lighting conditions and different realization of a recipe. Thus, identifying discriminative regions which help distinguish each type of food from the others is essential to the success of food recognition. Since

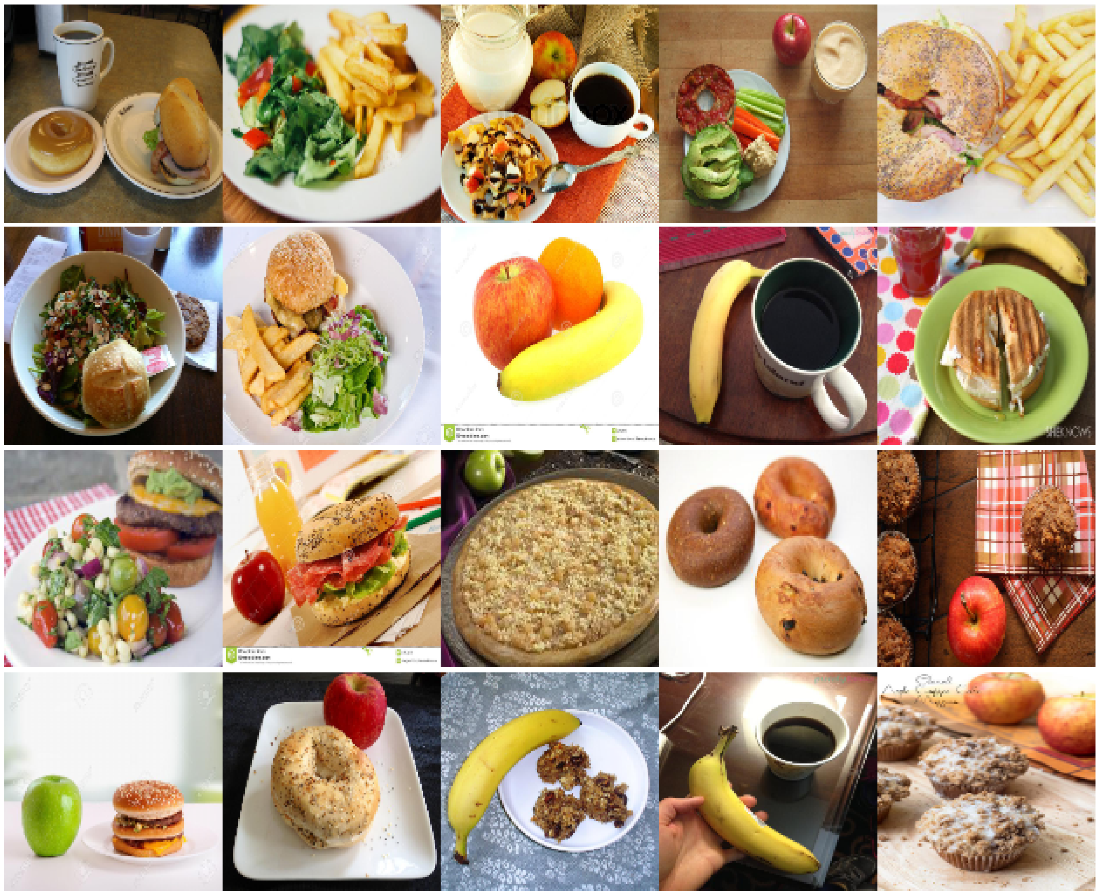


Figure 1.2: Real life food images contain multiple food items

large scale food image dataset with annotated food bounding boxes is not available, weakly supervised learning is considered to be a promising choice. Bossard et al. [7] addressed weakly supervised discriminative region mining in food recognition based on Random Forest. However, similar to most previous work, they still use the classical visual features like SURF and color histogram instead of the state-of-art CNN (Convolutional Neural Networks) features. Generic feature descriptors obtained from deep learning with convolutional nets are very powerful, outperforming classical visual features. Comparative studies [44] have suggested that CNN features should be the primary candidate in visual recognition and detection tasks.

In this thesis, we aim to solve the task of multiple food items recognition and detection using CNN features and weakly supervised learning. A classical processing pipeline to learn object detection models consists of: 1) proposing candidate regions from the image; 2) computing feature descriptors for all candidate regions; 3) training classifiers based on annotated object bounding boxes and their corresponding feature vectors. To train such an object detector, ground truth bounding boxes on target object instances are required in training. However, this hand labeling approach is very costly and error-prone for large scale training datasets. We believe that weakly supervised learning which only requires image level labels is a promising solution for multiple food items detection.

1.3 Thesis Contributions

We propose an algorithm which jointly detects food item combinations and their locations in the image, with only "weakly labeled" images for training. Without ground truth bounding boxes in training, our method can automatically detect the food items in the images and also localize their positions with promising accuracy rate. In our experiments, we have achieved the average recall and precision rate of 80.18% and 83.78% respectively on our test dataset when detecting 10 kinds of food. As far as we know, it is the first work in multiple food items recognition which use CNN features and weakly supervised learning. It is a novel algorithm with "minimal supervision".

Our contributions can be summarized in detail as follows:

1. We collect a dataset of 1792 food images, each of which contains one of the ten popular food categories, as our training data set. For the test set, we collect food images, each being a combination of food items from these 10 categories. Figure 1.1 and 1.2 show some examples of our training and testing images.

2. We compute a 4096-dimension CNN feature descriptor for each region proposed by Selective Search [53] on training images. The CNN architecture of AlexNet [35] is used for CNN feature extraction.

3. For each proposed region, we compute its Euclidean distances in 4096-d feature domain to all other candidate regions in the training set including positive regions (from the same class) and negative regions (from all other 9 classes); then we sort this distance list and retain top-N nearest regions.

4. Based on each region and its top-N nearest regions, we integrate the relevance for positive regions versus negative regions into the submodular cover framework to discover discriminative regions for each class.

5. With the discriminative regions for each class, we train a linear binary Support Vector Machine(SVM) for each class. For training, we use these discriminative regions as positive sets while all the other regions as negative sets.

6. To speed up the training process which has a much larger negative sets than positive sets, we perform iterative hard-negative mining. We also apply non-maximum suppression over the class scores calculated by SVM classifiers to make predictions at test stage.

7. We perform test experiments on images with combination of these 10 kinds of food. The recall and precision rate are computed on the test sets. Moreover, we perform a comparative test using the whole image as proposed region without any region mining, and compare its recall and precision rate with that obtained by our method. Our results suggest that region mining is essential to improve the detection results.

1.4 Thesis Outline

In Chapter 2, we review previous work in food recognition, including single food item recognition (Section 2.1), multiple food items recognition (Section 2.2) and multiple objects detection (Section 2.3).

In Chapter 3, we describe the motivation behind our proposed method, and present the processing pipelines of our method at the training stage and the testing stage.

In Chapter 4, we briefly introduce region proposal methods in object detection. Specifically, the strength of the Selective Search method is described in Section 4.1. For feature representation, we describe different convolutional neural networks (CNN) architectures and how we compute CNN features for proposed regions in Section 4.2. we present our method for discriminative region mining using maximum cover by submodular optimization in Section 4.3. In Section 4.4, we explain the training of SVM classifiers with hard-negative mining. In Section 4.5 we explain the non-maximum suppression algorithms we used on the detection results from the trained classifier.

In Chapter 5, we demonstrate our experiment results by figures and tables. We also compare the results using our proposed processing pipeline with the results obtained using the whole image as the proposed region.

In Chapter 6, we discuss the results and highlight the conclusion that could be drawn from the results. We also discuss possible future work in Section 6.2.

Chapter 2

Related Work

Food image recognition is an active research area and there are several existing approaches that have been previously published. The task to recognize what the food is in one image could be categorized into several problems. At a coarse level, it is enough to classify the food image, assuming that only one food item is in the image; in a more delicate way, the algorithm needs to figure out what food items are in the images and where they are located, which is a much more complicated problem.

2.1 Related Work in Food Image Recognition

Most existing research work in food recognition assumes that only one food item was present in one image. Thus, food recognition can be solved as an image classification problem.

Image classification is a core problem in computer vision. Classical approaches exploit interest point descriptors like SIFT [38] extracted locally or on a dense grid, then pool the features into a vectorial representation e.g., bag of words [36] and Fisher Vectors [49] to use SVM for classification. A very recent and successful trend in image classification is to use convolutional neural networks (CNN), a deep learning approach.

Concerning food recognition, most previous work uses classical approaches with differences in feature combinations of specified food image dataset. There are also a handful of recent food recognition approaches using CNN, yielding much higher recognition accuracy rate. We elaborate the most representative research work in food recognition for each method in the following two sections.

2.1.1 Classification Using Classical Approach

Bossard et al. [7] created a food database named *Food101*[1] containing 101 food categories. For each category, there were 1000 images. Firstly, they extracted the color and SURF [3] features for superpixels on each image. Then all the superpixels were clustered into groups using Random Forest based on their respective Fisher vector encoded feature vectors to obtain discriminative components across all the images, as shown in Figure 2.1. For each category, only the top N mined components were used to train a binary SVM, with the component in the leaf treated as positive set and all others as negative sets. Since there were much more negative samples in the training set, the hard-negative mining technique was used iteratively to speed up the training process. In this way, there were $K(classes) * N(TopNcomponents)$ SVM classifiers and for each image there would be a $K * N$ score vector. This score vector was used as the source of training data to train the multiple-class SVM for image classification. They achieved a test accuracy rate of 50.8%.

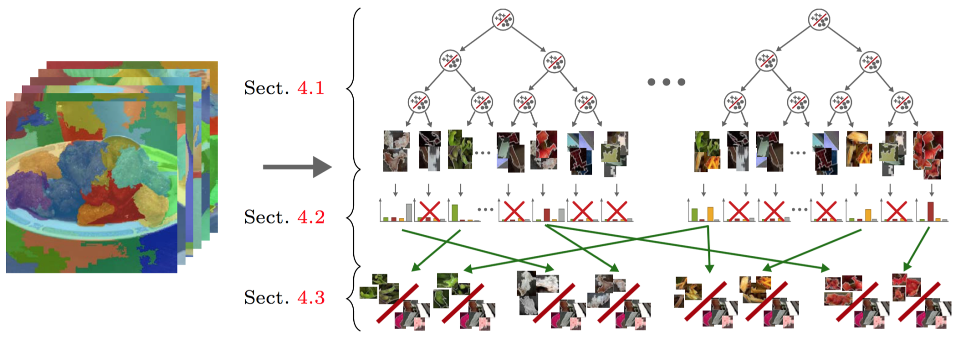


Figure 2.1: Discriminative component mining using Random Forest. For each superpixel, the LAB color feature and SURF feature are encoded by Fisher Vector. Random Forest is used to hierarchically cluster the feature vectors of superpixels of the training images. The discriminative clusters in the leaves are selected and used to train the component model. [7]

Joutou et al. [31] created a private Japanese food dataset with 50 classes and each class included 100 images downloaded from the Internet. They proposed a Multiple Kernel Learning(MKL) method using combined features including SIFT-based bag-of-features, color histogram and Gabor Texture features. The MKL method could determine the weights for each type of features, and they trained a SVM classifier based on the combined features with adaptive weights. They achieved an accuracy rate of 61.3% on their dataset. Hoashi et al.[26] continued this research work and achieved an accuracy rate of 62.5% using the same method on an extended dataset of 85 classes.

[1]http://www.vision.ee.ethz.ch/datasets/food-101/

Chen et al. [11] created the Pittsburgh food database which contained 101 classes of American fast food images taken in a controlled environment. For each class, there were 24 images taken in 3 instances. Yang et al. [58] defined eight basic food materials and learned spatial relationships between these ingredients in a food image using pairwise features. They achieved 28.2% classification accuracy rate on 61 food categories which was a subset of Pittsburgh [11] dataset.

Bettadapura et al. [4] incorporated the geological information of where the food picture was taken, they used this information to get the information about the restaurant and then downloaded the menu online. And they assumed that the food image must be one of the items in the menu. They combined 6 feature descriptors (2 color-based and 4 SIFT-based) together and used SMK-MKL Sequential Minimal Optimization to train a SVM classifier. They created a training food dataset with 3750 food images of 75 categories (50 images per category). They reported an accuracy of 63.33% on their test dataset.

2.1.2 Classification Using Deep Learning Approach

Kagaya et al. [32] trained convolutional neural network for food recognition and also non-food detection. They built a food database of 170,000 images containing 10 popular food items. Using 6-fold cross-validation, the optimal CNN hyper parameters related to the number of layers, pre-processing and training were decided. Examples of the optimal convolutional kernels in CNN after training show that the features extracted for food images were almost color-specific, which was in agreement with the conclusion obtained by Bosch et al's [5] food recognition research work using classical hand-crafted global and local features. Experiments showed that CNN outperformed all the other baseline classical approaches by achieving an average accuracy rate of 73.7% for 10 classes.

Kawano et al. [34] used CNN as a feature extractor and achieved state-of-the-art top-1 accuracy of 72.3% on the UEC-FOOD100 [2] dataset containing 100 classes of Japanese food. Since the UEC-FOOD100 dataset only contained 100 images for each class, the pretrained AlexNet proposed by Krizhevsky et al. [35] was used as a feature extractor. For each food image, a 4096-dimension CNN feature vector were obtained by taking the output of the layer just before the last layer (classifier layer) in AlexNet. By integrating both the CNN feature and Fisher Vector encoded conventional SIFT and color features, they trained one-vs-rest linear SVM classifiers for 100 food classes. Experiments showed that CNN features alone achieved 57.9% top-1 accuracy, while combination of conventional features and CNN features achieved 72.3% top-1 accuracy.

[2]http://foodcam.mobi/dataset

Bossard et al. [7] trained a deep CNN on *Food101* from scratch using the architecture of AlexNet [35] and achieved 56.4% top-1 accuracy on test set after 450,000 iterations.

Yanai et al. [57] fine-tuned the AlexNet which was pretrained on 2000 categories of the ImageNet dataset including 1000 food-related categories. They achieved the best results on public food datasets so far, with a top-1 accuracy of 78.8% for UEC-FOOD100 dataset and 67.6% for UEC-FOOD256. Their experiments showed that accuracy rate on a small set of food images like UEC-FOOD256 and UEC-FOOD100 (both of which contained 100 images for each class) can be boosted by fine-tuning the pretrained CNN net which was trained on a large data set of similar objects.

Most recently, Myers et al. [42] presented the Im2Calories system for food recognition which extensively used CNN-based approaches. They used the architecture of GoogLeNet [52] and fine-tuned the pretrained model on *Food101*. By replacing the last layer of GoogLeNet with different loss functions, problems in food recognition including food recognition, non-food detection and multiple food recognition could be solved efficiently. The resulting model has a classification top-1 accuracy rate of 79% on *Food101* test set.

2.2 Related Work in Multiple-Food Recognition

There are four research works on multiple-food recognition so far.

Matsuda et al. [40] presented the first work in multiple-food recognition and Figure 2.2 shows the processing flow of the method they proposed. Firstly, candidate regions generated by the whole image, deformable part model, circle detector and JSEG region segmentation were integrated to create a candidate set of bounding-box regions. Secondly, various kind of image features including bag of features (BoW), histogram of oriented gradient (HOG), Gabor texture features and color histograms were computed for each bounding box region in the candidate set. Evaluation values of each region belonging to all the given classes were computed using trained SVM classifiers by multiple kernel learning. Then, the evaluation values were sorted and the system will only output the top 10 classes for the user to choose. They further extended their work in [41] by considering multiple-food co-occurrence statistics. The sorted evaluation values were re-ranked by manifold ranking using a co-occurrence probability matrix. Their proposed system only listed all the possible food items and did not associate the location of bounding boxes with the food items names. They trained the SVM models on their own dataset which contained 9132 food images of 100 food categories. A top-10 accuracy rate of 65.6% was reported on their test set.

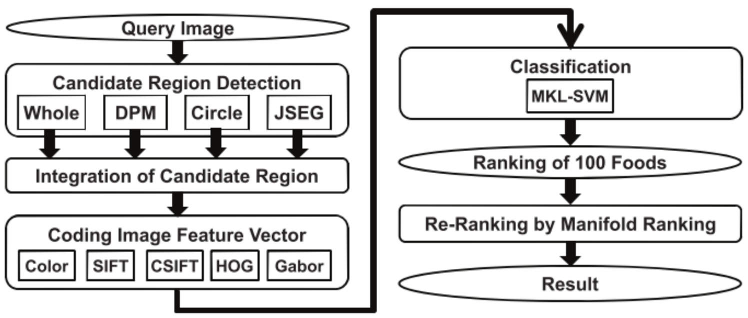
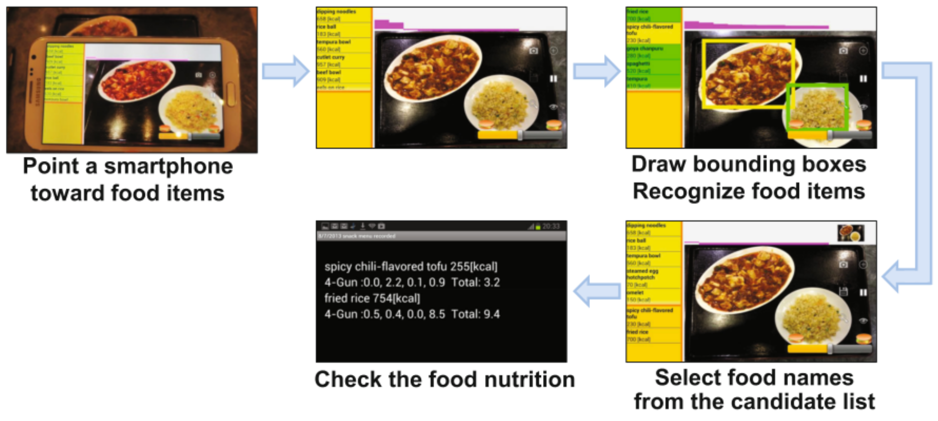

Query Image
Candidate Region Detection
Whole
DPM
Circle
JSEG
Integration of Candidate Region
Coding Image Feature Vector
Color
SIFT
CSIFT
HOG
Gabor
Classification
MKL-SVM
Ranking of 100 Foods
Re-Ranking by Manifold Ranking
Result

Figure 2.2: Multiple-food recognition processing flow [41]. SVM classifiers with chi-square RBF kernel were trained by multiple-kernel learning. The evaluation values computed by SVM classifiers were re-ranked by manifold ranking using co-occurrence statistics. Names of top 10 classes of food items were provided for users to choose from.

Kawano [33] et al. developed a mobile application FoodCam for real-time multiple food recognition. The processing flow of FoodCam is shown in Figure 2.3. In this system, users needed to draw bounding boxes over food items on the screen. This bounding box was adjusted to the food region by a GrubCut-based segmentation method. For each segmented region within the bounding box, Fisher Vector encoded feature vectors of color histogram and histogram of gradient were computed and passed into SVM classifiers to predict the food item. The system would show the top 5 predicted food names of each bounding box region. All the food recognition processing was computed on the mobile CPU. The SVM classifiers of 100 food categories were trained offline with a nonlinear RBF kernel on 12,905 food images. A top-1 accuracy rate of 51.9% and top-5 accuracy rate of 79.2% were reported on their test set.

Zhang et al. [59] developed a mobile application for multiple-food recognition of 15 food categories on the phone without any user intervention. Firstly, the user took a photo of the food plate and a cropped 400×400 food image was uploaded to the server for food recognition. On the server side, the food image was firstly segmented into possible salient regions, and these regions were further grouped based on the similarity of their color, HOG and SIFT feature vectors. Normally a typical food image yielded about 100 salient segment regions. They collected 2000 training images for 15 classes with mobile phone's camera as they found that the model trained with images downloaded from Internet could not generalize well for images taken on mobile phones. They trained a linear multiple-class SVM classifier for each class using the Fisher vector encoded feature vectors (including SIFT and color features) of salient regions. They reported a top1 accuracy rate of 85% when detecting 15 different kinds of foods in their experiments.

In the Im2Calories system Myers et al. [42] proposed, multiple-food recognition were


Point a smartphone toward food items
Draw bounding boxes Recognize food items
Select food names from the candidate list
Check the food nutrition

```
spicy chili-flavored tofu 255[kcal]
4-Gun :0.0, 2.2, 0.1, 0.9  Total: 3.2
fried rice 754[kcal]
4-Gun :0.5, 0.4, 0.0, 8.5  Total: 9.4
```

Figure 2.3: FoodCam system process flow. Users draw bounding boxes over each food item and image classification is performed for each bounding box region. Top-5 candidate food items are displayed on the screen for users to choose from.

solved by replacing the last layer named *softmax* classifiers of GoogLeNet by a multiple-label classifier named *logistic nodes*. They trained the CNN on an extended dataset of *Food101* which contained 201 food items. An average top1 accuracy rate of 50% were achieved, with mean top1 accuracy of 80% for classes in *Food101* and 20% for classes outside *Food101*. They argued that the number of training images for classes outside *Food101*, which were much smaller compared with classes in *Food101* which had 1000 images for each categories, was the reason for the low accuracy rate.

2.3 Related Work in Multiple-Object Detection

In real life scenarios, it is common that people have multiple food items in their plate. In order to address this problem, the system need to be able to localize and identify multiple food items in the image.

We aim to learn multiple food detectors to localize food items where only binary image labels that encodes whether an image contains this target object or not is provided. Thus our work is strongly related to the research area of multiple-object detection. Generally, the detectors are trained in two ways: the supervised way with ground-truth bounding boxes and the weakly supervised way with labels only for whole image. In this section, we review the most representative work of those two approaches.

2.3.1 Supervised Multiple-Object Detection

The state-of-the-art object detectors are trained with ground truth bounding box. Usually, there are three main steps in training an object detector:

1. Region Proposals: proposing candidate regions which may contains target objects for each image;

2. Region Representation: computing feature representation (eg. SIFT, HOG and CNN) for each proposed region.

3. Classifier training and bounding boxes location regression: training the classifier for bounding boxes localizing and content predictions.

The last step varies from one approach to another. Efficient approaches often combine the bonding box location regression and classifier training together to speed up the training process.

In the following section, we review the most representative work in this category.

Girshick [23] et al. proposed an algorithm which significantly improved the accuracy rate of object detection. Figure 2.4 [3] shows the pipeline of their method. They first used the method SelectiveSearch proposed by Uijlings et al. [53] to propose regions in the image which may contains objects. In their work, they used about two thousand regions in the image as the candidate regions. Then they used the AlexNet [35] to extract features of each of those regions.

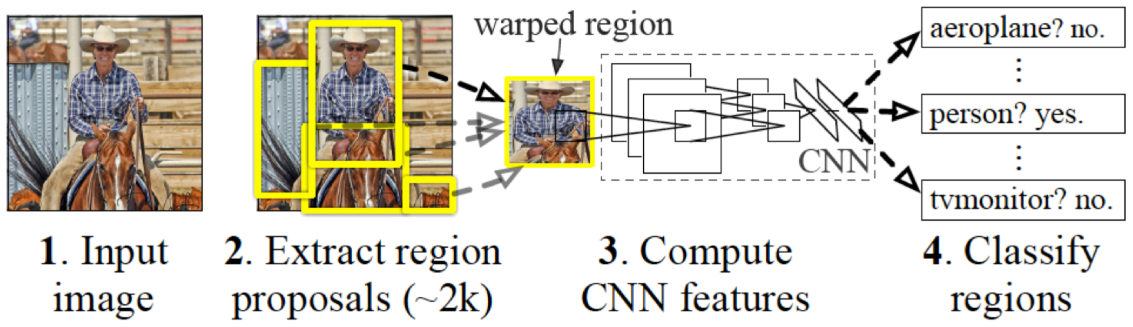


Figure 2.4: Overview of RCNN

Since the ground-breaking work of [23], many CNN based approaches achieved better performance and needed less time to train. Successors such as [22], [45], and [24] improved the detection accuracy rate significantly.

[3] Image is from [23]

2.3.2 Weakly Supervised Multiple-Object Detection

In object detection, weakly supervised learning refers to methods that rely on image-level labels which indicate the absence or presence of object instances in the images. Such methods do not require ground truth bounding box annotations on training images and can potentially be widely used on large amount of textually tagged images on the Web. There have been intensive studies in weakly supervised object detection in recent years. Most of these approaches adopt a similar framework, which shares following main steps:

1. Region Proposals: proposing candidate regions which may contains target objects for each image;

2. Region Representation: computing feature representation (eg. SIFT, HoG and CNN) for each proposed region.

3. Region Mining: finding out the most representative object regions (positive samples) among all the candidate regions.

4. Classifier training and refining: training a classifier for each object category based on the mined regions, and then applying hard negatives mining to refine the classifier.

Song et al. [50] proposed a method to learn object detectors from weakly labeled images. Images with only labels of whether the object was presented or not were called as weakly labeled images. They solved the detection problem by training latent SVM classifiers, where the mined bounding boxes positions in the positive images were the latent variables. They used the submodular cover algorithm to get a good initialization of candidate bounding boxes. They got a set of guessed bounding boxes in this step that were as good as manually labeled by human. They created a smooth formulation of LSVM, which allowed to use efficient algorithms for solving the optimization problem. They achieved the state-of-the-art result on the PASCAL VOC 2007 dataset among the approaches of weakly supervised learning.

Wang et al [55] proposed a similar procedure for weakly supervised object localization as [50]. However they focused more on the cluttered backgrounds and treated this information as the latent variable. They claimed that using the co-occurrence of the object and background could improve the detection accuracy rate.

Weakly supervised learning for object detection is becoming a hot topic and generally follows the four steps as described in the beginning of section 2.3.2.

Chapter 3

Overview of Our Proposed Method

We propose an algorithm which jointly detects food item combinations and their locations in the image, with only “weakly labeled” images for training. In other words, each of the training images contains only a single food item with no bounding box and its label, while the testing images may contain multiple food items. Our proposed algorithm will predict the labels of food items and their corresponding locations in the test images. Thus, it is a novel algorithm with “minimal supervision”. We overview the processing flows of our proposed method in Figure 3.1 and Figure 3.2.

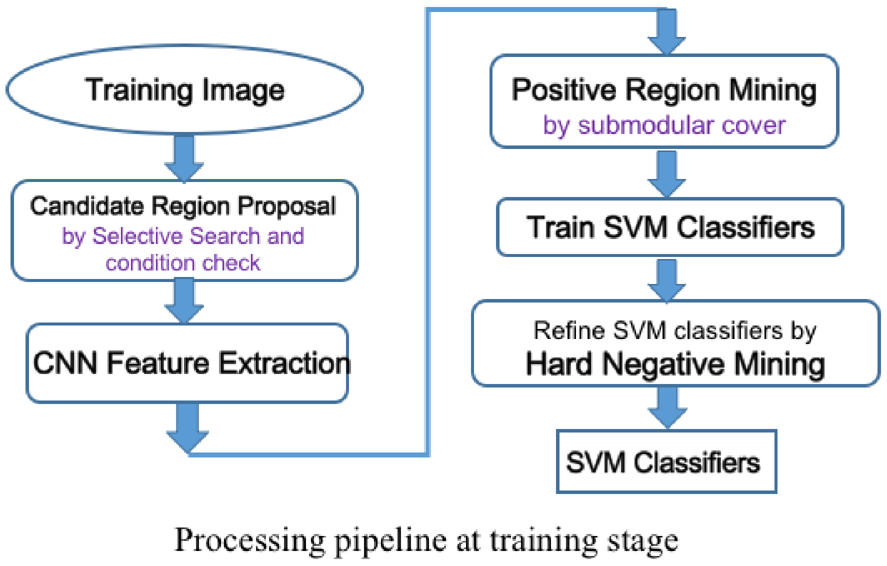


Figure 3.1: Overview of the training pipeline of Multiple-Food detection

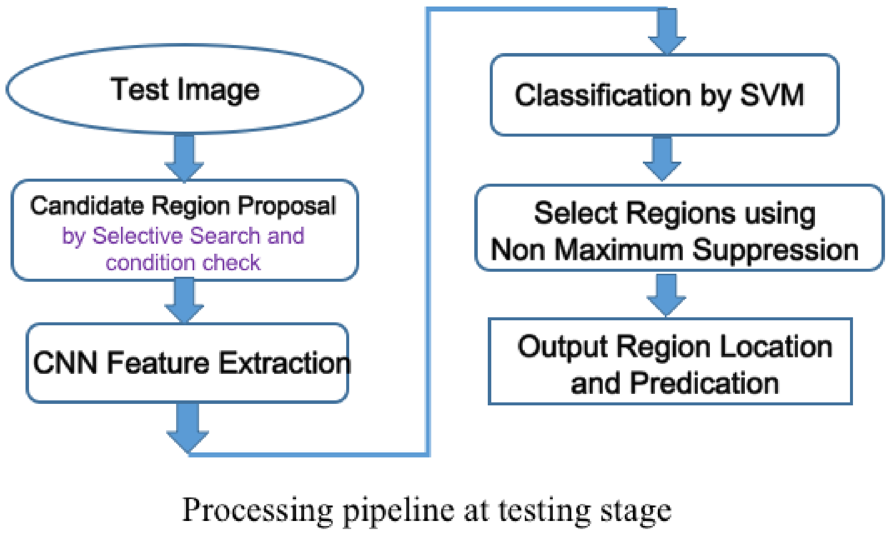

Test Image
Candidate Region Proposal by Selective Search and condition check
CNN Feature Extraction
Classification by SVM
Select Regions using Non Maximum Suppression
Output Region Location and Predication
Processing pipeline at testing stage

Figure 3.2: Overview of the testing pipeline of Multiple-Food detection

Given an input image, our method firstly generates candidate regions which may contain objects, using the segmentation based region proposal method named Selective Search [53].

Since each training image in our experiment contains only one food item, the sizes of these regions which contain the food item should not be too small and the aspect ratio of width and height of these regions should be in a reasonable range. Thus, we exclude irrelevant proposed regions regarding their size and aspect ratio to construct a much smaller set of candidate regions. In this thesis, we also use the whole image as the proposed region for comparison in our experiments.

Next, we extract the CNN features of each region in the candidate set for every training image. We use the pretrained AlexNet [35] model to represent each candidate region using the output of the model's $fc7$ layer, which is second fully-connected layer in AlexNet and contains 4096 neurons. Therefore, the feature vector of each region has the dimension of 4096.

Then, we propose a novel strategy to discover the discriminative regions for each food category. The idea is based on the fact that regions come from the same class (positive samples) share similar feature appearance and thus are close to each other in the feature space. We examine the discriminative power of each region by its *maximum cover*. The *maximum cover* of each region is computed by counting how many positive training images it *covers*. A region *covers* a positive training image if it is one of the top K nearest neighbors of regions in the image. Outputs of this procedure are a set of positive discriminative regions for each class.

With these mined positive regions, one binary linear SVM classifier is trained for each food category. A total number of 10 classifiers are trained from the mined regions. These classifiers are further refined iteratively with hard negative examples.

At the testing stage (shown in Figure 3.2), we first select the regions based on the classification scores, then we apply our improved Non Maximum Suppression (NMS) algorithm on these regions to generate the final results. The outputs of our proposed method are the region locations and its predicated food names.

Some of the results of our proposed method on test images are shown in Figure 3.3.

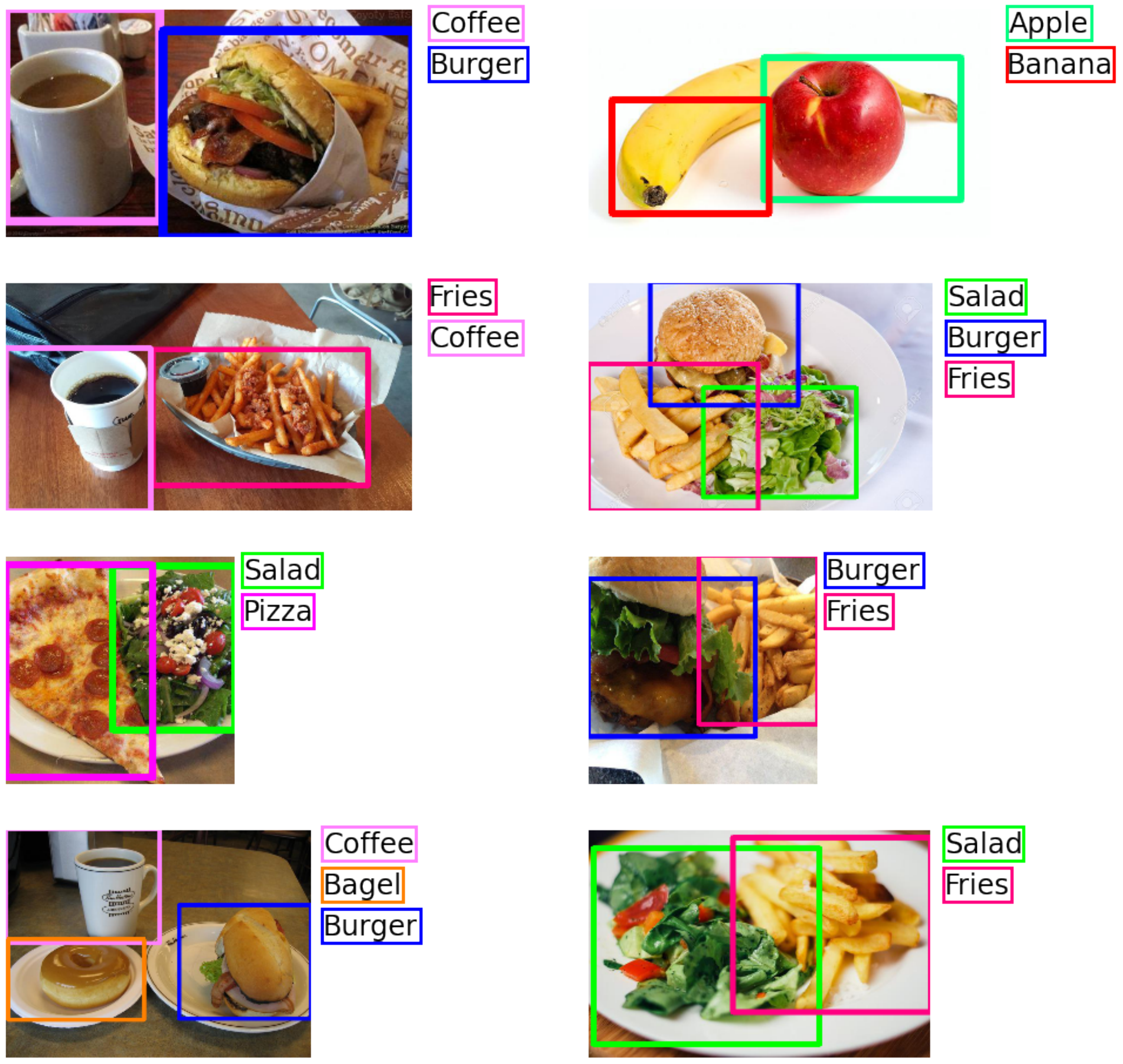


Figure 3.3: Sample detection results of our proposed method on test images

Chapter 4

Multiple-Food Detection

To recognize multiple food items and localize them separately in a single food image is a very challenging task, especially when there are no ground-truth bounding boxes provided during training. To the best of our knowledge, no previous work has addressed the problem of multiple food items recognition and detection in a weakly supervised manner.

Our goal is to learn a detector for each food category from a set of training images with only binary labels indicating whether the image contains this food category or not. We model an image as a set of overlapping rectangular windows, therefore, we can reduce the detection problem to the problem of binary classification of image rectangular windows. We model each image as a bag of instances (rectangular windows). The binary image label $y = 1$ specifies that the image contains at least one instance of the target category, and the label $y = -1$ specifies that the image contains no instances of the category. Since instance labels are not provided, we perform region mining with submodular cover optimization to find the positive instances which are rectangular windows containing the target food category. With the positive instances, we can train a classifier for each food category and further refine the classifier by mining the hard negatives .

In this chapter, we describe each step of our proposed algorithm in detail and offer a brief rationale for why certain methods and parameters have been chosen. We also show the intermediate results of each step at the training stage and the testing stage.

4.1 Detection Region Proposals

To detect objects in an image, candidate regions where objects are expected to exist are usually extracted to guide the object detection.

Region proposal methods can generate a small number of semantic regions where objects are most likely present, avoiding exhaustive sliding window search across the whole image. Good region proposal methods can propose high quality regions, all of which contains potential target objects. Meanwhile, the number of generated regions is also relatively small so that we can mine the regions effectively. There are many region proposal methods. We briefly overview them in Section 4.1.1 and discuss the region proposal method we choose in our proposed pipeline.

4.1.1 Introduction of Detection Region Proposal Methods

A method based on sliding window is an intuitive way to propose regions. This approach exhaustively searches across the whole image, yielding $10^4 - 10^5$ rectangular windows [27] on a typical image. In multiple-scale detection, the number of proposed windows grows by orders of magnitude with the number of different size of detection windows.

One approach to alleviate the computing burden is to propose regions only on objects of interest. One observation is that objects of interest share common visual properties most of time. Based on this observation, a much smaller number of regions which have high probability to contain objects of interest can be proposed to be candidate regions. Current state-of-the-art object detectors often employ region proposal methods to speed up the search for objects. Those approaches are more efficient than the sliding window approach as fewer windows are examined.

Currently there are two categories of region proposal methods [27]. One category is *window scoring* and the most popular method in this category is *Objectness*. The other category is *grouping method* and the most popular algorithm in this category is known as *SelectiveSearch* [53].

Objectness is proposed by Alexe et al. [1] in 2010. It is among the earliest well known region proposal methods. In this method, the salient locations in an image are used for initial proposals. Each of the salient regions are then scored based on the probability of containing an object [2]. The score estimation is based on a combination of multiple information such as color contrast, edge, location and size, saliency and its overlap with superpixels.

SelectiveSearch is proposed by Uijlings et al. [54] [53] in 2011. This method greedily merge similar regions together to generate candidate regions. Since the author designed similarity functions manually, there is no learned parameters. Because of its high recall rate and no customized parameters, it is the most widely used region proposal method and most recent object detection methods use SelectiveSearch as region proposal method.

4.1.2 SelectiveSearch Region Proposal

Good region proposal methods typically depends on cheap features and quick inference model [45]. SelectiveSearch greedily merge superpixels to generate proposals where features and similarity functions used for merging superpixels can be manually designed. It proposes high quality candidate regions at different scales using multiple grouping criteria with relatively small computational cost.

SelectiveSearch has been broadly used as the detection proposal method by many state-of-the-art object detectors. The idea of this method is grouping superpixels to generate a hierarchy of small to large regions. The processing procedures of this algorithm can be summarized as follows:

1. Generate initial regions. In this step, SelectiveSearch use the superpixel extraction method described by Felzenszwalb et al. [20] to generate as many regions as possible, and each of them belongs to at most one object.

2. Recursively merge similar regions into larger ones greedily until only one region left. This procedure yields a hierarchy of successively larger regions. The merge criteria is based on the similarity in color and texture between regions. The similarity is measured by the similarity metric, which is defined as follow:

   (a) Choose the color space from RGB, HSV,and Lab color space.

   (b) Compute the color similarity based on the selected color space using histogram intersection.

   (c) Compute the HOG-like features for texture similarity measurement.

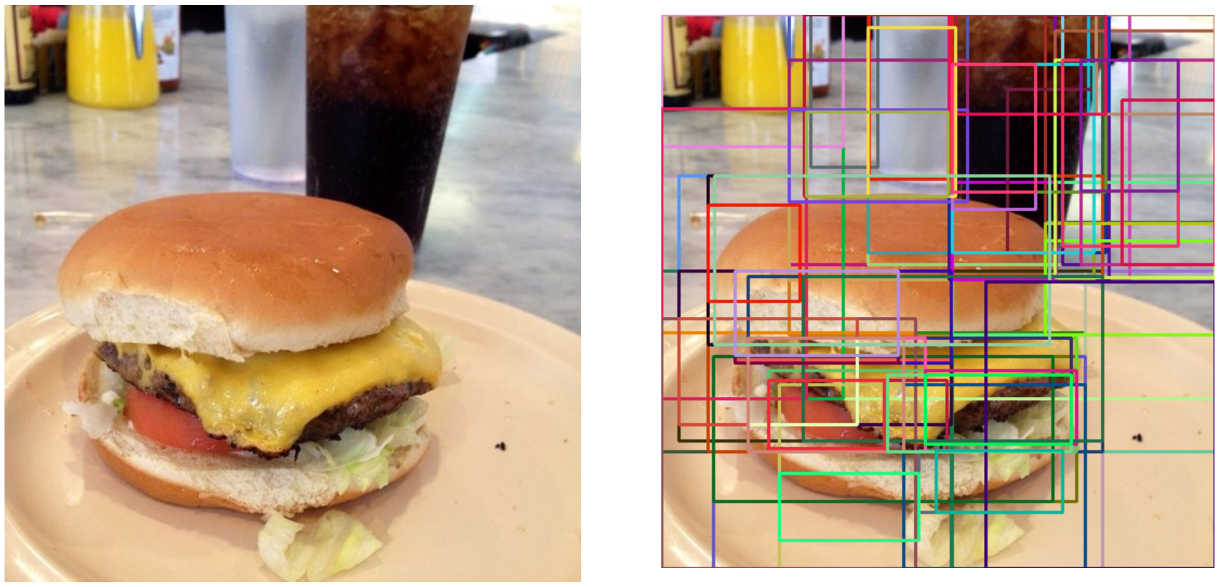
   (d) Add a size component in the similarity metric so that all smaller regions are more similar to each other.

   (e) Shape compatibility also measures similarity of two regions.

   The function of similarity measure is computed as a linear combination of all the metrics described above.

3. Lastly, output regions which likely contain candidate objects.

We adopt SelectiveSearch to generate region proposals for multiple-food detection. Figure 4.1 (a) shows the original image; Figure 4.1 (b) shows the regions proposed by SelectiveSearch overlaid on the original image, and only a small number of proposed regions are shown. Usually more than 2000 regions can be generated for a single image by SelectiveSearch. We further filter the candidate regions based on their size and aspect ratio. Regions that are either too small (area is less than 1/100 of the original image) or the ratio of length to width is too large (larger than 20) or too small (less than 1/20) are removed. Because of this simple and reasonable filtering, we successfully decrease the number of proposed regions per image from about 2000 to about 150. This significantly improves the speed and accuracy of the subsequent procedures.



(a). The Food image (b). Proposed region windows by SelectiveSearch

Figure 4.1: Example of a food image (left) with proposed regions computed by SelectiveSearch overlaid (right)

4.2 Region Representation

Extracting features to represent the regions in feature space is an important component of object detection. Traditional feature extractors are based on hand-engineered methods, such as HOG[14] and SIFT[38]. Recently large performance improvements on detection benchmarks have been realized by training deep convolutional neural networks. Generic feature vectors extracted from CNN are very powerful and have been reported to outperform all of other feature extracting methods [44]. It is suggested that features obtained from deep learning with CNN should be the primary candidate in most visual recognition tasks [44].

In this section, we briefly describe convolutional neural networks and some of the most popular CNN architectures. Then we explain how to extract feature vectors from convolutional neural networks in our experiments.

4.2.1 Convolutional Neural Network

Neural network is inspired by biological visual cortex. In the central nervous system, each neuron receives input signals from the output of its dendrites, and then combines all the input with some kind of computation to generate an output signal to another neuron. In this model, the neuron will fire if the combination of all the input signals is strong enough. The activation function f is used to calculate the strength of the output signal. Sigmoid function is a common choice of activation function. Figure 4.2 shows a biological neuron and its corresponding mathematical model.

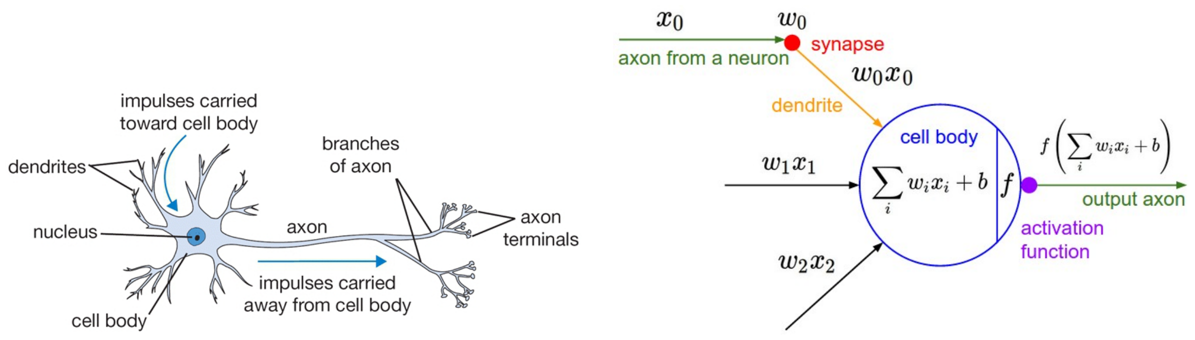


Figure 4.2: A biological neuron (left) and its corresponding mathematical model (right). [1]

Convolutional neural network is very similar to ordinary neural network and it is explicitly designed for inputs which are images. CNN is make up of layers and each CNN layer transforms an input 3D volume of activation neurons to an output 3D volume of neurons with

[1]Images are from http://cs231n.github.io/neural-networks-1/

some differentiable function that may or may not have parameters. There are four main types of layers to build CNN architectures: Convolutional Layer, Pooling Layer, RELU layer and Fully-Connected Layer.

There are several popular CNN architectures and we briefly describe three most popular CNN architectures below.

Yann LeCun applied CNN computer vision tasks in his ground-breaking work [37]. In this work, he trained a CNN name LeNet5 for recognition of handwritten digits. His architecture (Figure 4.3) achieved the highest accuracy rate (error rate less than 1%) compared to other traditional method at the time. The LeNet5 contains 2 convolutional layers, 2 subsampling(pooling) layers, 2 fully connected layers, and one output layers. Figure 4.3 shows the architecture of this network.

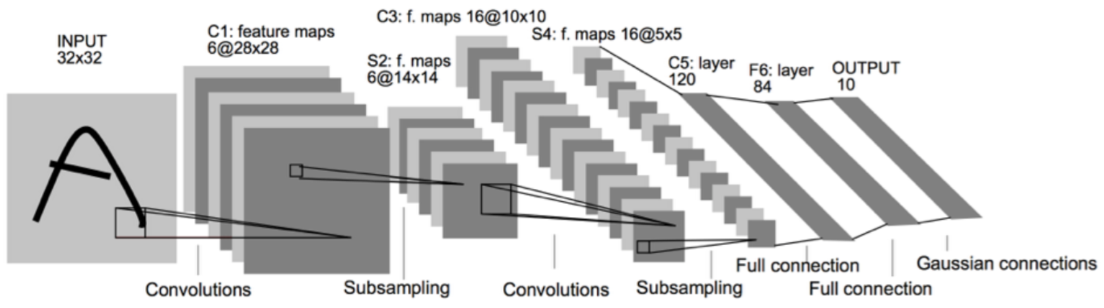


Figure 4.3: Architecture of LeNet5 [2].

Krizhevsky et al. [35] unveiled the power of deep convolutional neural network for the task for image classification. The ImageNet organization holds an ImageNet Large Scale Visual Recognition Challenge (ILSVRC) on their large image dataset [15] yearly since 2010. There are 15 million images in the dataset spread in 22000 categories. Every year many teams submit their algorithms try to improve the classification accuracy rate.

In 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC2012), [35] proposed a large deep convolutional neural network which outperformed far away than many teams and won the competition. They achieved top-1 and top-5 error rates of 37.5% and 15.3% respectively, while the second place achieved a top-5 error rate of 25.7%. This was a major breakthrough in image classification. Since then, lots of better classification results are reported. The AlexNet contains 60 Million parameters, and even just to store the parameters will require 230 Megabytes. Figure 4.4 shows the architecture of the AlexNet.

[2]Image is from [37]
[3]Image is from [35]

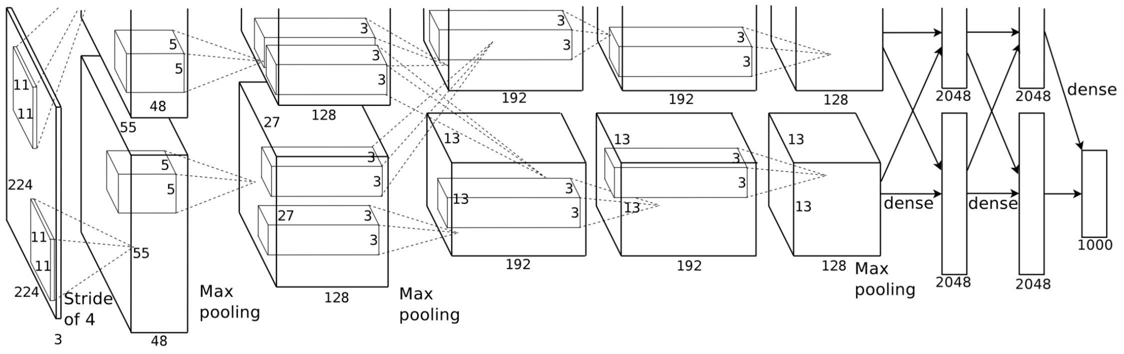


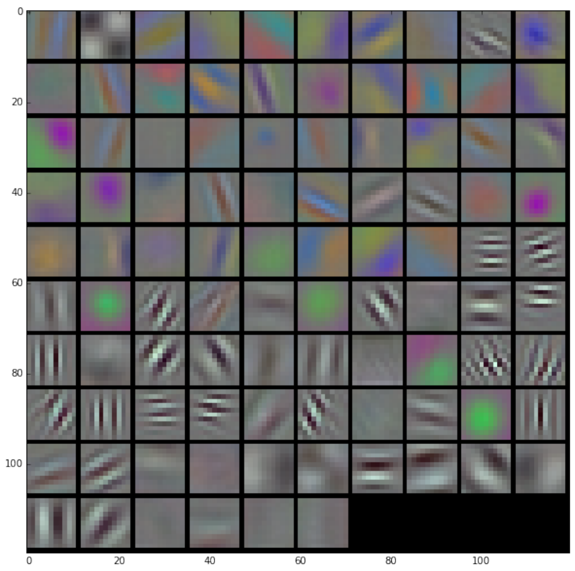
Figure 4.4: Architecture of AlexNet [3]

4.2.2 Feature Extraction with Convolutional Neural Network

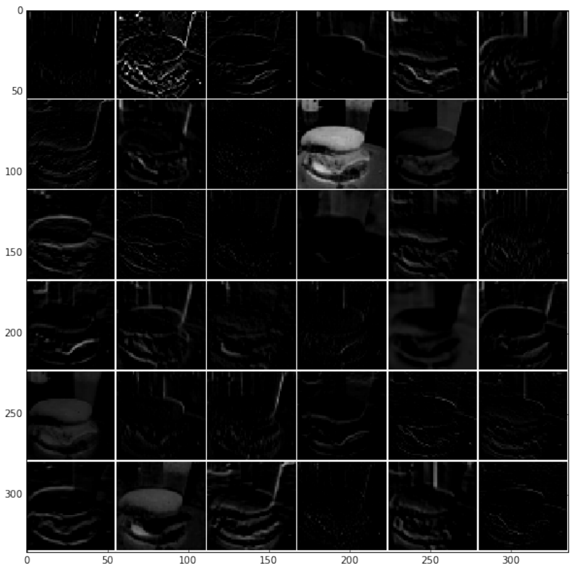
Using features extracted from the convolutional neural network is becoming the standard feature extraction method. In 2012 with the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [15] [16] classification breakthrough obtained by convolutional neural network, the power of CNN has been explored extensively. Donahue et al. [17] showed that the winner (AlexNet [35]) of ILSVRC12 could be used as a powerful black-box feature extractor. Currently the best-performing methods for detection also employed CNN to extract high level features. Girshick et al. [23] used features extracted from AlexNet in their ground-breaking work on object detection. Figure 4.5 visualizes the features extracted from a burger image using the $f7$ layer of AlexNet.

In our experiments, we use the pre-trained AlexNet model as our feature extraction model. We use the Caffe [29] deep learning framework and its pre-trained AlexNet model to compute the feature vectors of the proposed regions. For each region, we first resize it to 227 by 227 pixels to meet the input requirement of AlexNet. Then the pixel values of the region are subtracted by the mean and then normalized to between $[0.0, 1.0]$. The normalized pixel values are then forward propagate through the neural network and a 4096-dimension feature is extracted by using the output of the last convolutional layer ($fc7$ layer) in the model. As we have decreased the number of proposed regions from about 2000 to about 150 per image (every image has different number of proposed regions), after extracting features for all these regions, a feature matrix of 150×4096 is computed for each image. This feature matrix is used as the input data for the region mining process.

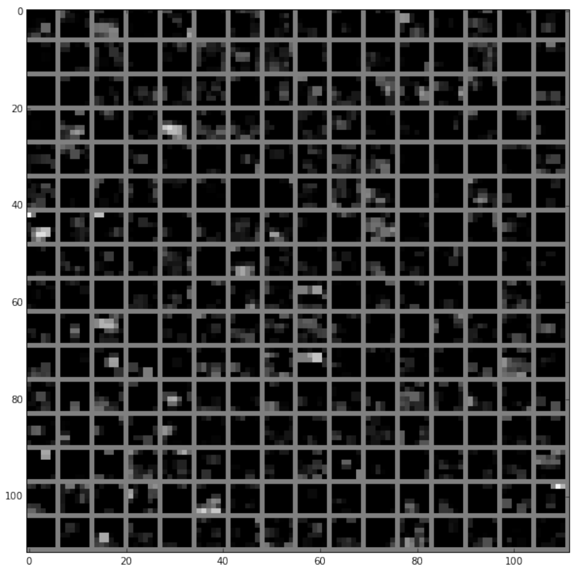
(a). Sample input image



(b). The filter of first convolutional layer



(c). The output of first convolutional layer



(d). The output of last (the 5th) pooling layer

Figure 4.5: Visualization of a burger go through the AlexNet. (a) Sample input image to the neural network; (b) Visualization of the filters in the first convolutional layer; (c) The activation output of the second fully connected layer $fc7$ by sample input image; (d) The activation output of the 5th pooling layer, which is the input of the second fully connected layer $fc7$.

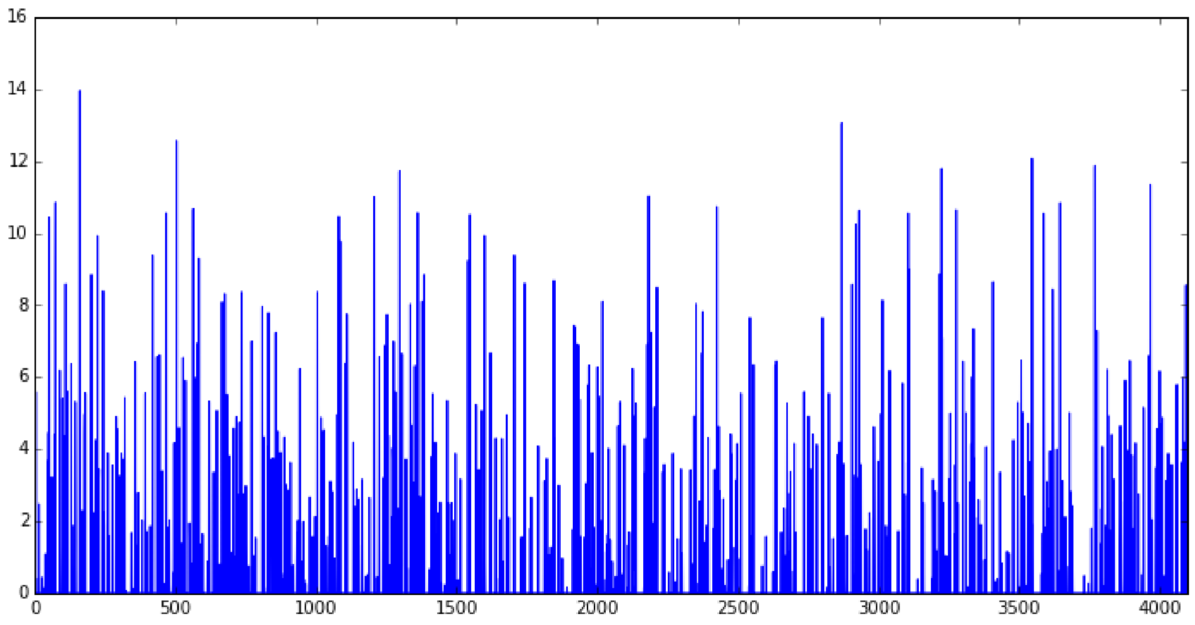


Figure 4.6: Values of the 4096-d feature vector extracted from one proposed region of the burger image shown in Figure 4.5.

4.3 Region Mining

With the features of all candidate regions extracted, we do not know which region contains the object and can be used as positive samples to train classifiers because class labels of regions are not provided. Figure 4.1 shows one example of the result of the proposed regions generated by the SelectiveSearch algorithm. We can observe that although the SelectiveSearch algorithm do propose many regions which contain objects, most of those regions do not contain or just contain part of the target objects (burger) we want to detect. Thus we need to find out *which region contains the target object?*

In this section, we describe an effective region mining strategy to discover a set of positive region samples for each class.

4.3.1 The Goal of Region Mining

Region mining aims to find two types of regions from all the proposed regions of training images.

1. Positive regions: Based on the fact that different proposed regions have different discrimination to the target object class, region mining procedure evaluates the discrimination to discover the positive regions which best differentiate the target object class and backgrounds.

2. Hard Negative regions: Negative regions consist of two parts: the background regions from positive images belong to the same class and the regions from negative images belong to other classes. Hard negative regions are samples that are similar to positive regions and falsely detected by the classifiers.

In this section, we focus on positive regions mining with the algorithm described below.

4.3.2 Maximum-Cover Region Selection

Based on the assumptions that: i) the correct rectangular windows are similar across training images which belong to the same food category in the feature space and must be clustered well, and ii) the correct rectangular windows do not occur in the negative images (training images belong to all other food categories), we formalize the intuition to mine the positive regions as a submodular cover problem.

For each region, we integrate its relevance to regions coming from positive images (belong to the same object class) versus negative images. Similar to [50], we create a bipartite graph

$G = (R, I, e)$ to represent this relationship. Each image has a set $R_I = \{r_1, ..., r_n\}$ of proposed regions from previous step. Set I contains all regions of positive images R_I of a food category. Set R contains all the regions of all the images. An edge e exists between R and I if region r_i from R is one of the top-k nearest neighbors of region in I. The idea is that, if a region r_i is the nearest neighbor in both positive and negative regions, then the number of edges stem from this region would be small since we only count edges connected to positive image regions. This is reasonable because if the region is close to regions in both positive and negative images, it is very possible that this region is background.

As shown in Figure 4.7, the regions in the top row are the individual proposed regions, and those at bottom are their nearest neighbors with respect to other training images belong to the same class in the feature space. All training images from the same class are treated as positive images and images from all other categories are treated as negative images. For each proposed region, its distances to all other regions proposed on both positive and negative training images are computed and ranked. Based on the top-k nearest neighbors, we count the number of regions coming from positive images and used it as a score for next step.

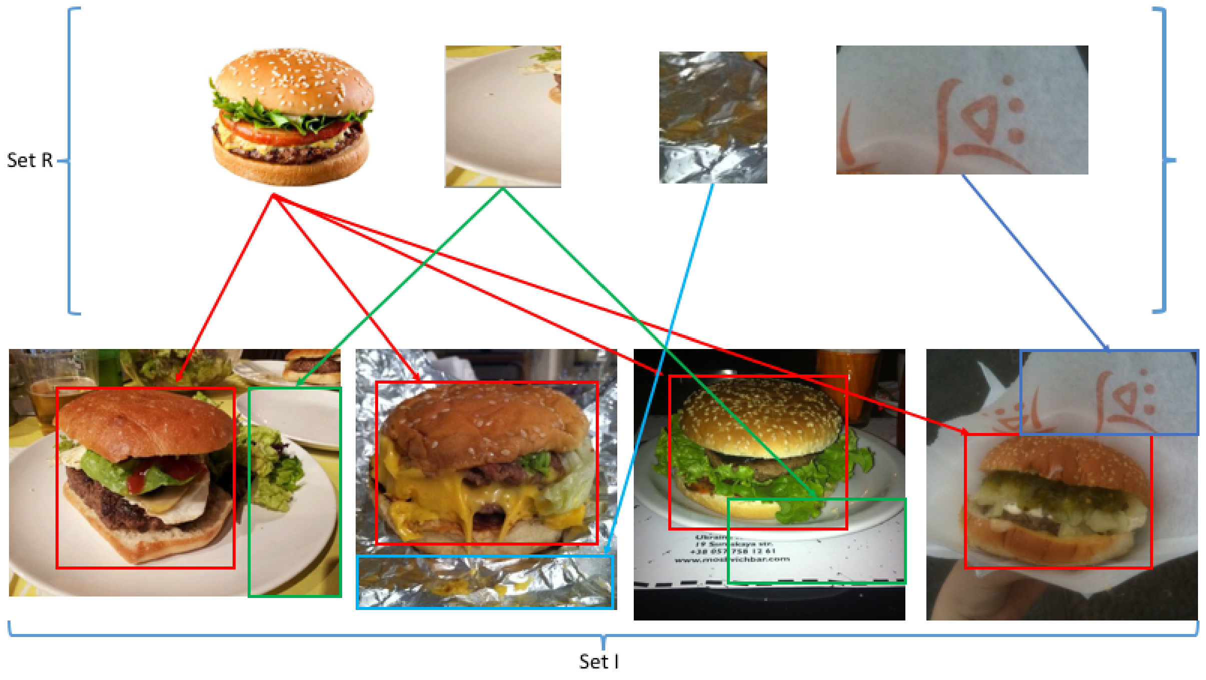


Figure 4.7: Regions with Maximum Cover

Maximum-Cover by Submodular Optimization

After we get the scores of each region, we want to select a set of regions $S \subseteq R$, that (i) can well represent the category; (ii) is small enough for fast training. We have two reasons for that: i) there are too many regions from the region proposal step and, ii) only a small portion of these regions contain the target food item. How many regions are needed to represent one food category is a problem with submodularity: the representation margin diminishes by adding one more region to the set S compared to previous added region. We define the function $N(S)$ as the neighbors of a set of regions $S \subseteq R$.

A set function $F : 2^{\Omega} \rightarrow \mathbb{R}$ is **submodular** if it satisfies *diminishing gains* [21]. That is, for any $S \subseteq T \subseteq \Omega \setminus \{v\}$, it holds equation 4.1

$$F(T \cup \{v\}) - F(T) \leq F(S \cup \{v\}) - F(S), \forall v \in \Omega \tag{4.1}$$

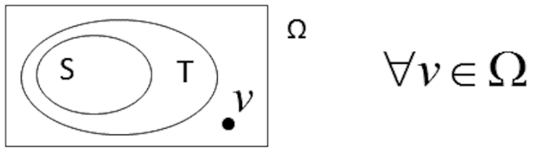


Figure 4.8: Submodular Set Function. Set T get less margin by adding v compare to the margin get by adding v to set S.

We define a covering score $C_{I,t}(S)$ for each image I. The score is determined by a covering threshold t and the number of regions from R_I that are in the set of neighbors of S, that is, $N(S)$.

$$C_{I,t}(S) = \min\{t, |N(S) \cap R_I|\} \tag{4.2}$$

The covering score $C_{I,t}(S)$ measures how many regions in R_I are the near neighbors of S, if there are many regions in R_I which are the near neighbors of S, then set S can well represent the image I, and we call that set S covers image I. The covering score $F(S)$ from equation 4.3 for the whole set s is then defined as sum of scores on all positive images P of that category.

$$F(S) = \sum_{I \in P} C_{I,t}(S) \tag{4.3}$$

Our target is to find a set of regions S from R that minimize equation 4.3, while having enough representation ability. We want regions in S come from each of the positive images, and at the same time to keep S as small as possible. This is indirect controlled by the threshold parameter t. If t is small, than regions in S will come from regions from many different images and the set S is relatively small. on the other hand, if t is large, there will be many more regions in set S, which *covers* each image more.

To minimize $F(S)$, we want regions in S are those with the most number of edges stem from positive images. An edge e exists between R and I if region r_i from R is the top-k nearest neighbors of region in I. That means the more edges stem from one region, the more positive images it can *covers*, the higher representation ability it has.

Function $F(S)$ is a submodular function and can be optimized via a greedy algorithm. This is because:

Function $F(S) : 2^{\Omega} \rightarrow \mathbb{R}$ defined in Equation 4.3 is a submodular function.

A set function that satisfies decreasing gains is submodular. That is, for $v \in \Omega$ and $S \subseteq T \subseteq \Omega \setminus \{v\}$, it holds $F(S \cup v) - F(S) \leq F(T \cup v) - F(T)$.

Let $Cov = |N(S) \cap R_I|$, then function Cov is a covering function. For $S \subset T \subseteq \Omega \setminus r$, we can have

$$N(S) \subseteq N(T) \tag{4.4}$$

$$\begin{aligned} |N(T \cup \{r\})| - |N(T)| &= |N(r) \setminus N(T)| \\ &\leq |N(r) \setminus N(S)| \\ &= |N(S \cup \{r\})| - |N(S)| \end{aligned} \tag{4.5}$$

Thus function Cov is a non-decreasing submodular function. Function $F(S)$ is the sum over Cov, thus also is a non-decreasing submodular function.

Optimization Our goal is to select a representative minimum subset $S \subseteq R$, which can be stated as:

$$\min_{S \subseteq R} |S| \text{ s.t. } F(S) \leq \alpha F(R), \text{ for } \alpha \in (0, 1]\,. \tag{4.6}$$

Submodular cover is an old problem. Wolsey et al. [56] analyzed the greedy algorithm for the submodular set covering problem in 1982. We adopt the greedy optimization algorithm. Let $S_0 = \emptyset$ and in each following iteration step τ, add the region r that can *cover* the largest number of *uncovered* regions. More formally, add the region that could maximize the marginal gain $F(S_\tau \cup \{r\}) - F(S_\tau)$. Figure 4.9 shows the top-10 regions selected from the greedy algorithm for the burger training images with the $\alpha = 0.95$ [50].

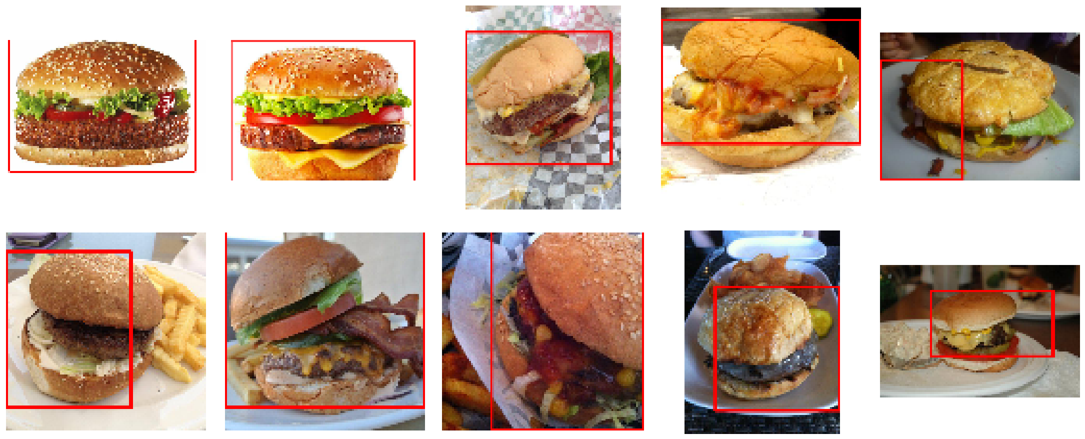


Figure 4.9: The top-10 regions in set S for the training dataset of the burger category

4.4 Classifier Training

After we get the candidates of positive and negative regions, we use the features extracted from those regions to train an initial classifier. As we have explained in the begin of this chapter, we chose to use Support Vector Machine (SVM) [6] [13] to train the binary classifier for each class.

4.4.1 SVM Optimization

Support Vector Machine (SVM) is one of the most important algorithms in machine learning since it was introduced by Boser et al in 1992 in [6].

Usually, a linear classifier can be expressed by $y_i = f(x) = w^T x_i + b$, where x_i is the feature vector of the sample i. In the training phase, we have the ground truth label y_i, we use the pairs (x_i, y_i) of the training data to obtain the weights vector w and bias b. After that, we could use this model to predict on test data. In the testing phase, the feature vector x_i is first obtained from the sample, and then we compute the prediction by $y_i = w^T x_i + b$. The function $f(x) = w^T x_i + b$ is one hyperplane to separating the training data into categories. As Figure 4.10 shows, H_2, H_3 are the separating hyperplanes for the training data, in which H_3 has the largest margin between two classes.

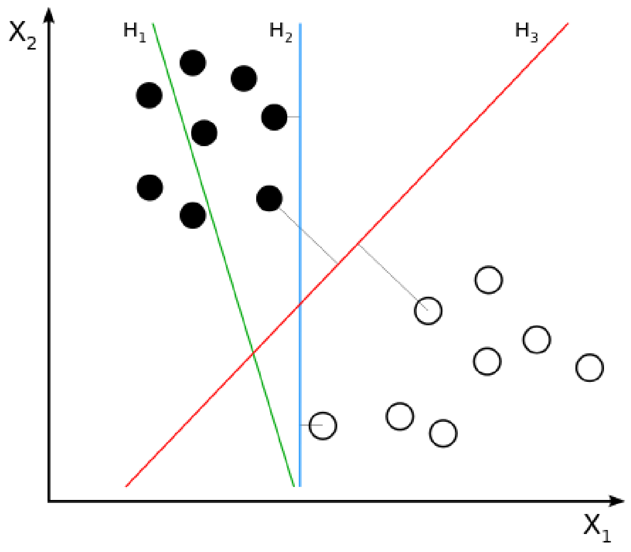


Figure 4.10: Example hyperplanes. H_1 does not separate the two classes. H_2, H_3 can separate them but H_3 with the largest margin. [5]

[5]Image is from https://en.wikipedia.org/wiki/Support_vector_machine
[7]Image is from https://en.wikipedia.org/wiki/Support_vector_machine

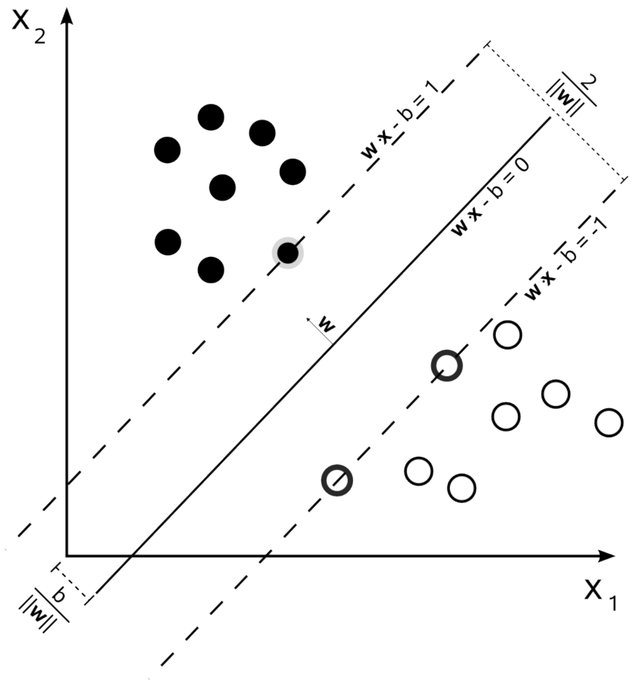


Figure 4.11: Maximum-margin hyperplane for an example SVM with two classes. Samples on the two marginal hyperplanes are the support vectors. The margin is the distance between two marginal hyperplanes. [7]

The idea of SVM is to find a discriminative hyperplane that represents the largest margin between the two classes in the training data. Figure 4.11 shows the hyperplane for a linear SVM and its support vectors. The margin the hyperplane has between two classes can be expressed as $\frac{2}{||w||^2}$. Thus target of SVM is to maximize $\frac{2}{||w||^2}$ to get the maximum margins between the two classes. That is:

$$\min ||w||^2, \text{ subject to: } \begin{cases} (w^T x_i + b) \geq 1 & \text{if } y_i = 1 \\ (w^T x_i + b) \leq -1 & \text{if } y_i = -1 \end{cases}$$

This can be combined to:

$$\min ||w||^2 \text{ subject to: } y_i(w^T x_i + b) \geq 1 \tag{4.7}$$

The Loss Function

To deal with non-separable case, we can add some slack variable ξ_i and the problem becomes as:

Minimize:

$$||w||^2 + C \sum_{i=1}^{m} \xi_i \tag{4.8}$$

Subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \tag{4.9}$$

for all *m* training instances in the training data.

Thus the loss function of SVM is:

$$L(w, b) = \frac{1}{m} \sum_{i=1}^{m} \ell(w^T x_i + b, y_i) + ||w||^2 \tag{4.10}$$

With the slack variable ξ_i, the loss ℓ for one sample is no longer the zero-one loss, but is the "hinge-loss":

$$\ell(y, \hat{y}) = \max(0, 1 - y\hat{y}) \tag{4.11}$$

The final task is to minimize:

$$\min L(w, b) = \frac{1}{2}||w||^2 + C \sum_{i=1}^{m} \max(1 - y_i(w^T x_i + b), 0) \tag{4.12}$$

Optimization of SVM Loss Function

The target loss function defined in Equation 4.12 can be optimized by the BroydenFletcher-GoldfarbShanno (BFGS) algorithm. We use the SciPy package [30] which implements this algorithms. The code listed in 4.1 is our implementation using the *fmin_bfgs* function from the module *optimize* in the SciPy package.

Listing 4.1: Python code to optimize the loss function with SciPy optimize module

```
def update_model(class_id, pos_feat,neg_feat,
    w, b, pweight=2, bias_mult=10, C=0.001, hard_iter=1):

    i = 0
    while i < hard_iter:
        hard_threshold = -1.0001
        feat_dim = pos_feat.shape[1]
        if w is None:
            w =  np.zeros(feat_dim + 1)

        x, y = get_taining_set_with_neg_mining(pos_feat,
            neg_feat, pweight, w, hard_threshold)

        def lost_func(w):
            return get_svm_l1_hinge_cost(class_id, w, x, y, C)

        def grad_func(w):
            return get_svm_l1_hinge_grad(w, x, y, C)

        pos_num = pos_feat.shape[0]
        neg_num = x.shape[0] - pos_num*pweight

        w = fmin_bfgs(lost_func, w, fprime=grad_func,
            disp=1, maxiter=3)
        i += 1
    return w
```

The key for the optimization is to get the loss and gradient. The gradient of the hinge loss is:

$$\frac{\partial l_{hinge}}{\partial w} = \begin{cases} 0 & y_i w \cdot x \geq 1 \\ -y_i x & y_i w \cdot x < 1 \end{cases} \tag{4.13}$$

The sum of the gradient is:

$$\frac{\partial L_S^{hinge}}{\partial w} = \sum_i \frac{\partial l_{hinge}}{\partial w} \tag{4.14}$$

Based on the mathematics equation, we implemented the procedure in the Python. The code listed in 4.2

Listing 4.2: Python code to compute the gradient and the loss for each category

```
# calculate the hinge loss for the class
def get_svm_l1_hinge_cost(class_id, w, x, y, C):
    scores = np.dot(x, w.T).reshape(-1, 1)
    margins = y*scores
    loss = np.maximum(0, 1 - margins)
    n = np.linalg.norm(w)
    cost = 0.5 * n*n + C * np.sum(loss)
    return cost


# calculate the sum of gradient for the class
def get_svm_l1_hinge_grad(w, x, y, C):
    scores = np.dot(x, w.T)
    scores = scores.reshape(-1, 1)
    margins = y*scores
    margin_test = margins < 1
    loss_grad_mat = np.dot((margin_test * ( -1 * y)).T , x)
    grad = w + C * loss_grad_mat.reshape(-1)
    return grad
```

4.4.2 Hard Negatives Mining

Our training dataset is imbalanced. This is because we use the regions from other categories as negative samples. Therefor, the number of negative sample is hundreds of times of the positive samples (we have 10 categories, each have about 150 regions proposed, and less than 10 percent of the regions are selected to be the positive instances). The impact of imbalanced dataset on the learning algorithm has been investigated since decades ago [28] [10]. In our setting, we have two problems about this imbalanced data. First, imbalanced dataset harms the accuracy rate of the classifier [28] [10] [25]. Second, it is costly to train such a large dataset.

Dalal and Triggs et al. [14] developed the method of data-mining for hard negative examples which is based on the bootstrapping methods used by [48] and [51]. In the work of Pedro F. et al. [19], the authors extensively used this method to improve their classifier. In our experiment, we also used this method to speed up the training process and to improve our classification accuracy rate.

Initially, when there's no "hard negatives", we select the k negative regions from all negative regions of a food category, where k is the number of mined positive regions. During previous mining step, we have already calculated the distance between negative and positive regions. With this information, we select top-k negative regions whose distances are the closest to those positive regions, those regions then become the initial "hard negative", which are used to train the initial linear SVM classifier.

With the initial classifier, we get the parameter w and b of the linear SVM. We can get the prediction from this initial classifier with the feature vector x_i of region i by equation 4.15. For negative regions, an accurate SVM classifier should get $f(x) \leq -1.0$. However, for the initial classifier, there are many negative regions are wrongly classified. To improve this initial classifier, we need to iteratively refine the classifier by re-train it with emphasize on the wrongly classified instances.

$$f(x) = w^T x_i + b \tag{4.15}$$

The refine process mainly focus on the negative regions for two reasons. First, the number of negative instances is hundreds times more than positive instances. Selecting only the "hard" ones can increase the distance between positive and negative "support vectors", thus increasing the accuracy rate. Second, the number of positive regions is small. If we focus only on the wrongly classified ones, then we will tend to have very few training instances, which will seriously under-fit the classifier, thus decreasing the accuracy rate. Focusing only on negative instances will not have the over-fit classifier problem due to its abundance of instances.

To mine the hard negatives, we compute the $f(x)$ for all the negative regions and then sort the results by descending order, so the instances with the largest values are in the front (they are the "hard" ones). Then the top-*k* regions are selected base on the sorted results. Those top-*k* regions are then become the "hard negatives" for the next iteration to refine our classifier.

4.5 Non-Maximum Suppression

One major problem for object detector is that it almost always detects multiple bounding boxes surrounding the object in the image. Figure 4.12 shows one example of this problem. This is because the region proposal method proposes many windows near the target object area. An efficient solution to this problem is to use the non-maximum suppression (NMS) method to select only representative regions as the output region.

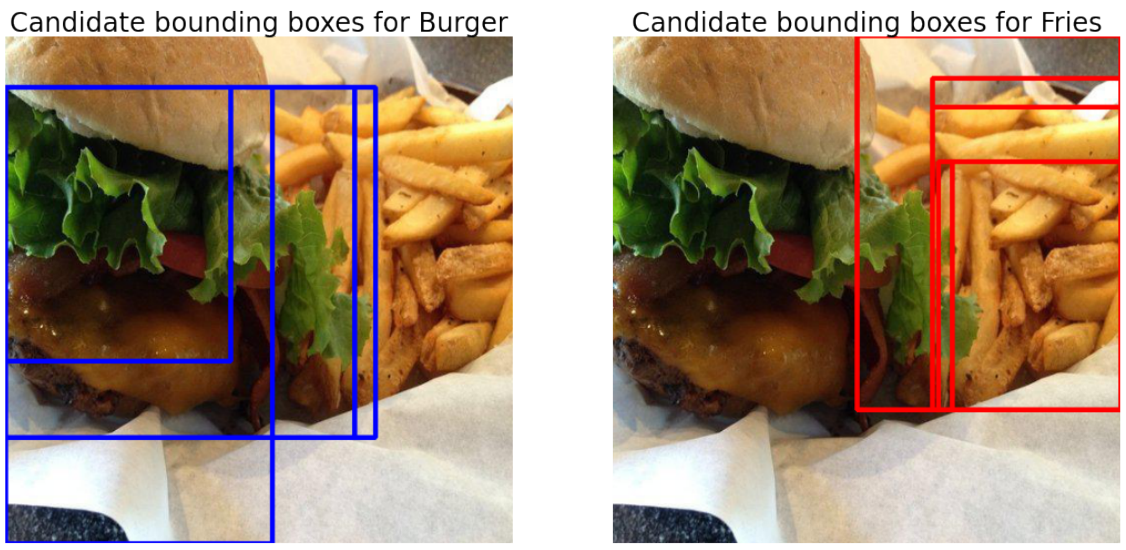

Candidate bounding boxes for Burger
Candidate bounding boxes for Fries

Figure 4.12: Results without Non-Maximum Suppression. Left side are 4 candidate bounding boxes and their predictions (burger). Right side are 5 bounding boxes and their predictions (fries).

NMS has been widely used in object detection tasks and became a part of many detection pipelines. It was first used by Canny et al [9] for edge detection. In the area of object detection, Dalal et al. [14] used non-maximum suppression for human detection and made this method popular. Their method is based on the Mean-Shift algorithm [12]. More recent object detectors such as [19] and [39] also employed this method to deal with multiple bounding box for one single object.

4.5.1 Greedy Non-Maximum Suppression

Initially, we used the greedy iterative NMS algorithm similar to Felzenszwalb et al. [19]. We start by selecting the highest score region, the regions which classified to be the same food and are too close to the top score window then are suppressed. Then the next highest score region which has not been processed is selected and the previous steps are repeated until all the regions are processed. The distance between regions is defined by the ratio of overlapping area to the area of the current processing region, if it is larger than 0.5, then there's overlap, and the current regions will be suppressed.

However this approach has some drawbacks. It selects the regions with the highest score as the output. This approach does not always select the best region [46]. Usually the regions contain only the center area of the target object received the highest score, which makes the output region not covering the whole object. Based on this observation, we make a small modification on the greedy approach which improves the accuracy of our bounding box.

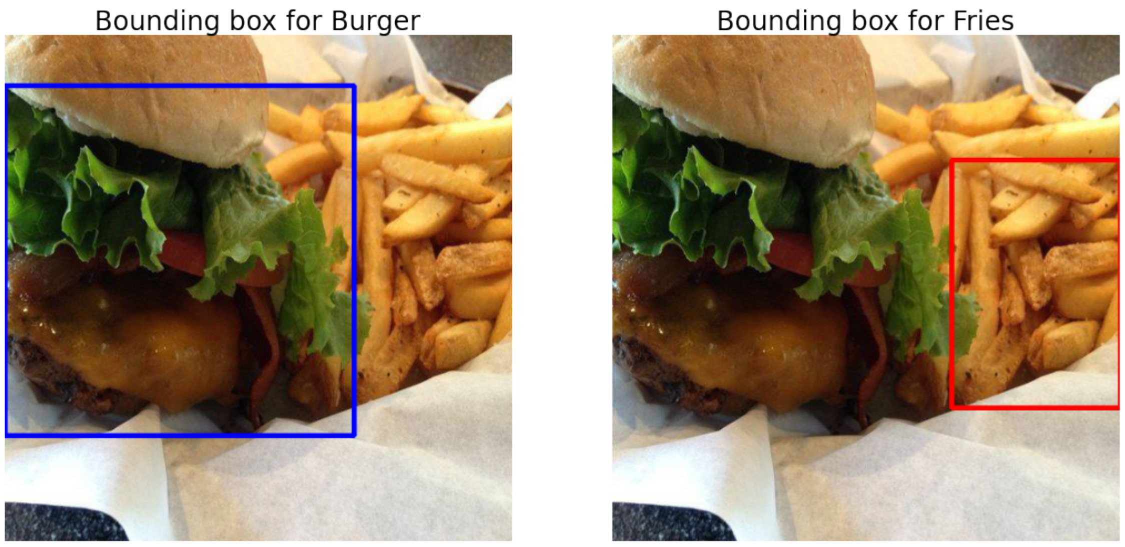


Figure 4.13: Results with greedy version of Non-Maximum Suppression with top-5 regions. Left side is the final bounding box and its prediction (burger). Right is the final bounding box and its prediction (fries).

4.5.2 Non-Maximum Suppression on Top-N Regions

Normally, there are several proposed regions around the target food area. Those regions usually have the same predicted label as showed in Figure 4.12. With this information on hand, we could fuse the top-N predictions on that area to generate one output bounding box instead of just using the top-1 score.

In our experiment, we set N to be 5. This has the advantage to capture all the strong responses from the classifier and drop those not so confident prediction. In the selected top-N regions, we first run the greedy version of the NMS algorithm to get the initial location of the object. Then we loop through the other four candidate bounding boxes, to see if any of them significantly overlaps with the one selected by the greedy NMS algorithm. *Significantly overlap* means the ration of the overlapped area of the two regions with the candidate region exceed at certain threshold α (In our experiment $\alpha = 0.8$). If the candidate bounding box is significantly overlap with the selected one and is larger than the selected one, then it becomes the selected bounding box, we repeat this process until all the regions have been processed. After the process, the selected region then become the final output of the detection. Figure 4.14 shows the resulting regions for the detection.

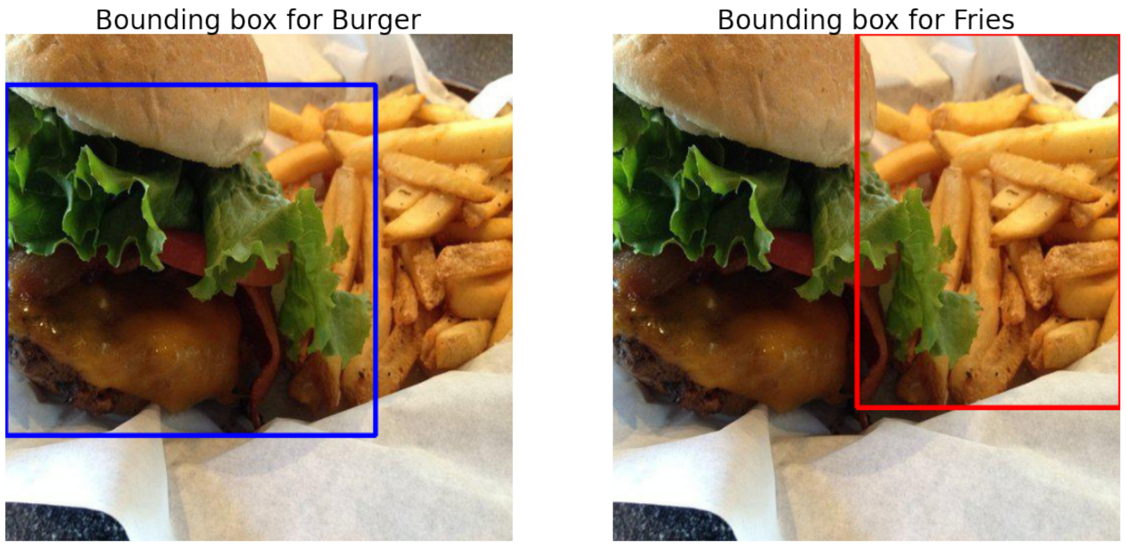


Figure 4.14: Results with Non-Maximum Suppression with top-5 regions. Left side is the final bounding box and its prediction (burger). Right is the final bounding box and its prediction (fries).

Chapter 5

Experiments and Results

In this chapter, we describe how we train and evaluate the proposed method on a manually collected food dataset. Detailed experiment setup is given in the following sections.

5.1 The Dataset

The food dataset used for our experiments is composed of a training set and a test set.

For training, we collected 1792 images of 10 categories of food. Each one of the training images only contains one food item. A detailed number of training images for each food type is listed in Table 5.1.

For testing, we collected 153 food images and each image contains at least two different food items from the 10 classes. In Table 5.2, we summarize the number of instances of each food category shown in the test dataset.

Table 5.1: The training dataset

Categories	**Number of Images**
Apple	145
Orange	128
Banana	95
Salad	201
Bagel	205
Pizza	202
Burger	204
Fries	325
Muffin	200
Coffee	87
Total	1792

Table 5.2: The testing dataset

Categories	**Number of Instances**
Apple	47
Orange	17
Banana	19
Salad	18
Bagel	17
Pizza	14
Burger	28
Fries	23
Muffin	15
Coffee	40
Total	135

Figure 5.1 shows some examples of the training and testing images.

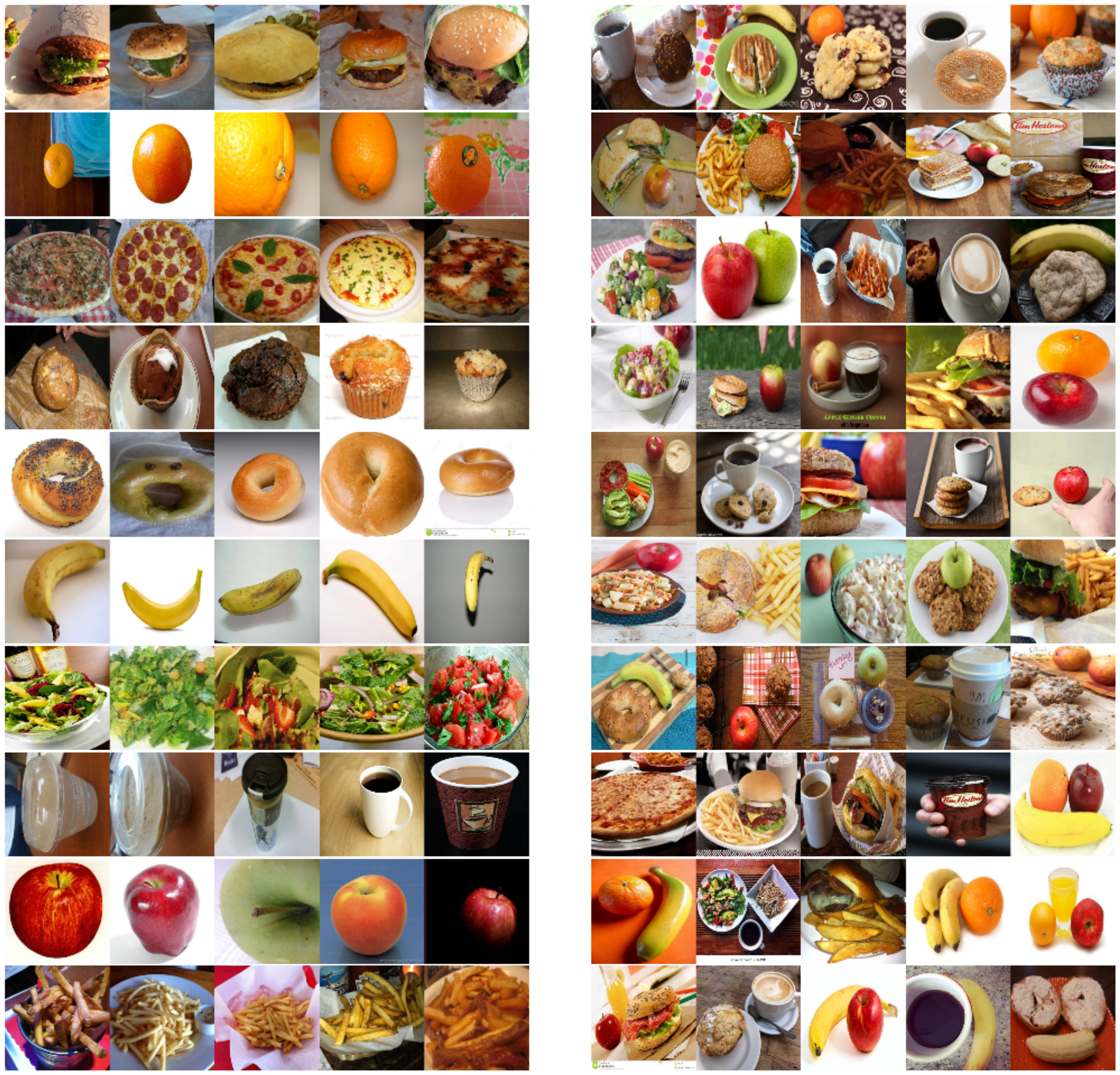


Figure 5.1: Examples of training in the food dataset and testing (right) images. The training dataset contains images with only single food item and the testing dataset contains images only with multiple food items.

5.2 Detector Evaluation Criterion

Mean Average Precision (mAP) The PASCAL Visual Object Classes (VOC) [18] defined a criterion to compute the mean average precision (mAP) for object detection evaluation, which has become the standard evaluation criterion. The mAP metric is used to compare a detection d_i to a ground truth bounding box g_i by computing the ratio of the area of the intersection $|d_i \cap g_i|$ over the area of union $d_i \cup g_i$ of the two regions. When the overlap exceeds 0.5, the detection is considered to be successful. Then the Average Precision (AP) on each class is calculated based on the detection result. This is an accurate evaluation criterion on tasks such as object detection. However, this criterion requires the ground truth bounding boxes of the test dataset to compute the ratio, which are not available in our experiment.

Classification Rate (CR) Matsuda et al. [40] developed a criterion to evaluate the results of multiple food detection:

$$CR = \frac{\text{num. of correctly detected food items in top-N candidates}}{\text{num. of all the food items in all the test image}} \tag{5.1}$$

This criterion does not consider the location information and evaluate on top N classification results.

Average Recall and Precision We report detection accuracy as average top-1 recall and precision on the test dataset to evaluate our proposed algorithm instead of top-N. This is meaningful for our experiment since recall rate measures the ability to detect food items and precision rate measures how accurate the detection is. The recall rate is computed by Equation 5.2 and the precision rate is computed by Equation 5.3 as follows:

$$recall = \frac{\text{num. of correctly detected items in the category}}{\text{ground truth num. of items in the category}} \tag{5.2}$$

$$precision = \frac{\text{num. of correctly detected items in the category}}{\text{num. of predicted items in the category}} \tag{5.3}$$

5.3 Experiment Results

Following the proposed processing pipeline, we train multiple-food detectors on the training images and test their performance on the test dataset. An average recall rate of 80.18% and precision rate of 83.78% is obtained on the test dataset. Detailed recall rate and precision rate for each food category is listed in Table 5.3.

Table 5.3: The recall rate and the precision rate on the test dataset with our proposed pipeline

Categories	**Num**	**Recall**	**Precision**
Coffee	40	95.00%	92.68%
Apple	47	78.26%	94.74%
Salad	18	100.00%	85.00%
Muffin	15	73.33%	64.71%
Fries	23	76.19%	94.12%
Bagel	17	50.00%	70.00%
Burger	28	80.77%	77.78%
Orange	17	82.35%	87.50%
Banana	19	81.25%	86.67%
Pizza	14	84.62%	84.62%
Average	24	80.18%	83.78%

Figure 5.2 shows some successful recognition and localization by our proposed method. More results are attached in Appendix A. Although food items vary a lot in appearance, size and occlusion, our proposed method correctly recognizes and localizes most of the samples. Promising results are shown in Figure 5.2. However, our proposed algorithm also generates inaccurate detection on some of the test images. Figure 5.3 shows some inaccurate detection examples. We discuss the reasons for those inaccurate detections in chapter 5.4.2.

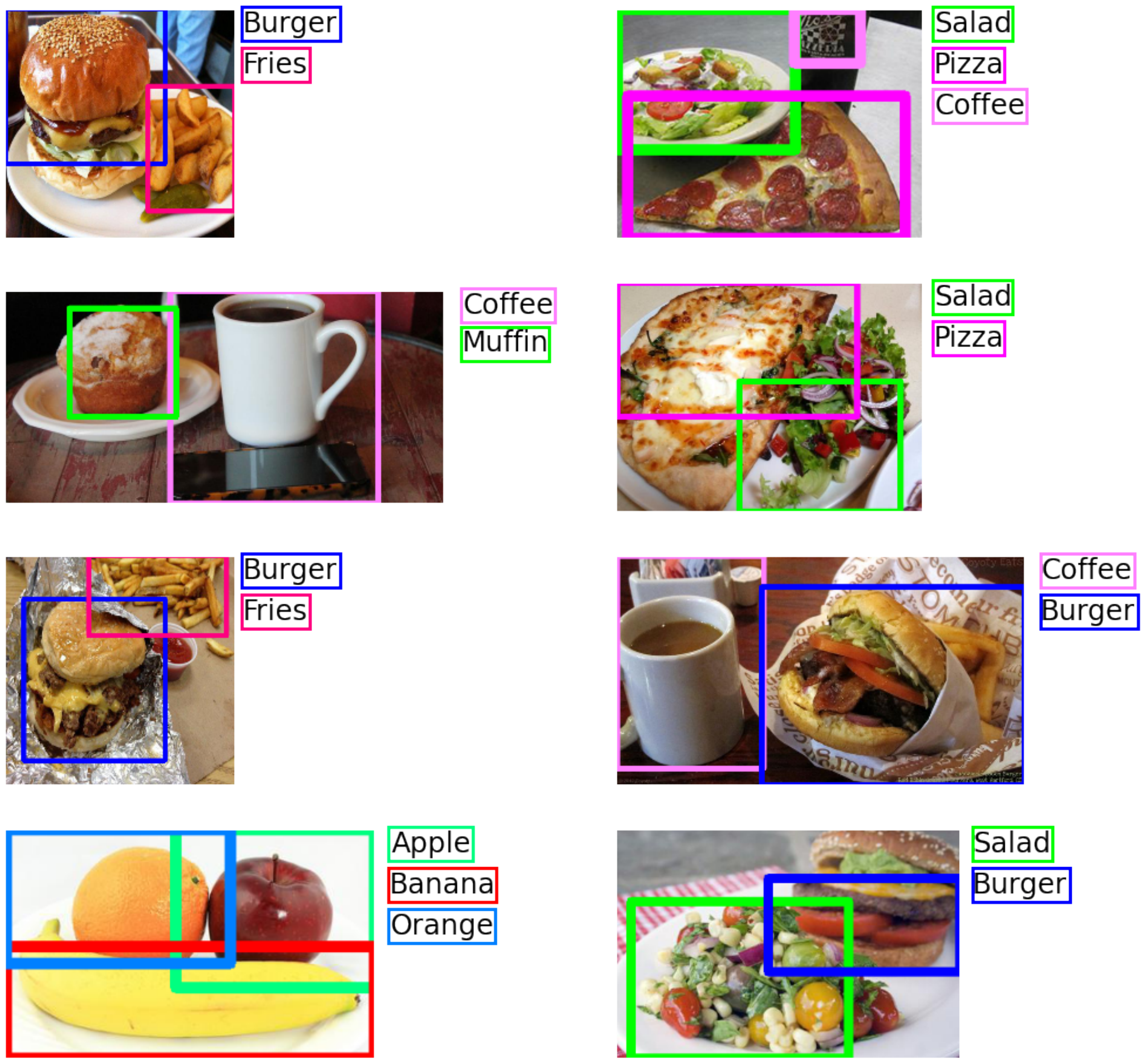


Figure 5.2: Example of successful detection results on test images. Our proposed algorithm can recognize the food items and also localize them in the image correctly.

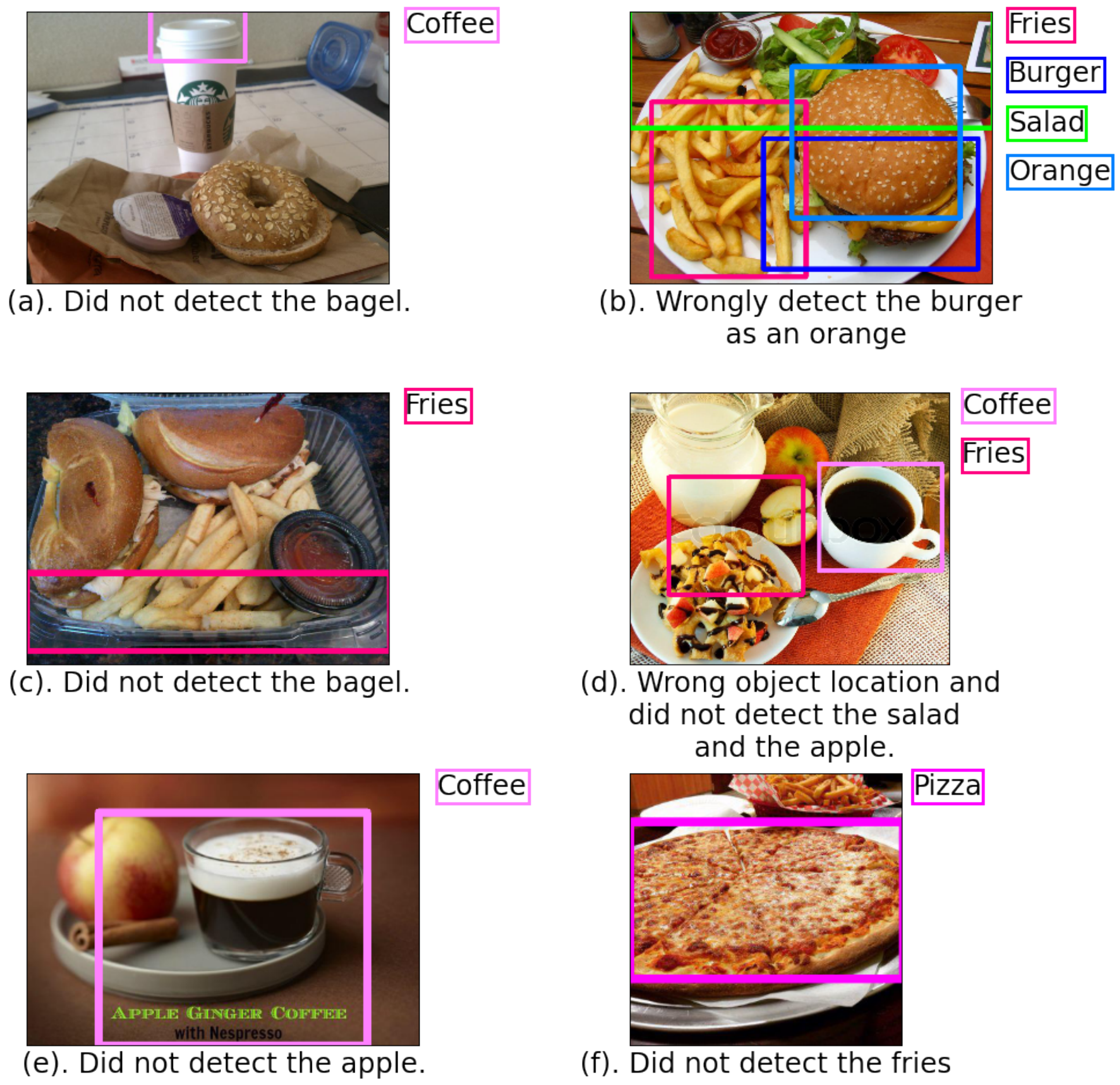

Coffee
(a). Did not detect the bagel.
Fries
Burger
Salad
Orange
(b). Wrongly detect the burger as an orange
Fries
(c). Did not detect the bagel.
Coffee
Fries
(d). Wrong object location and did not detect the salad and the apple.
Coffee
APPLE GINGER COFFEE
with Nespresso
(e). Did not detect the apple.
Pizza
(f). Did not detect the fries

Figure 5.3: Some inaccurate detection results on the test images. (a). Did not detect the bagel in the image. (b). Wrongly detect the burger as an orange. (c). Did not detect the bagel. (d). Wrong object location and did not detect the salad and the apple.(e). Did not detect the apple. (f). Did not detect the fries.

5.3.1 Comparison Experiment

To evaluate the significance of region mining procedure in object detection, we conduct a comparison experiment which uses the whole image region to train multiple food detectors without any region mining involved during training. Because the training dataset only contains a single food item per image, the whole image region can be used as a positive sample in training. An overview of the processing procedures at training stage is shown in Figure 5.4. At training stage, SVM classifiers are trained using all the training images themselves as positive samples. The rest procedures are exactly the same, with hard negative mining during training and non maximum suppression at testing stage.

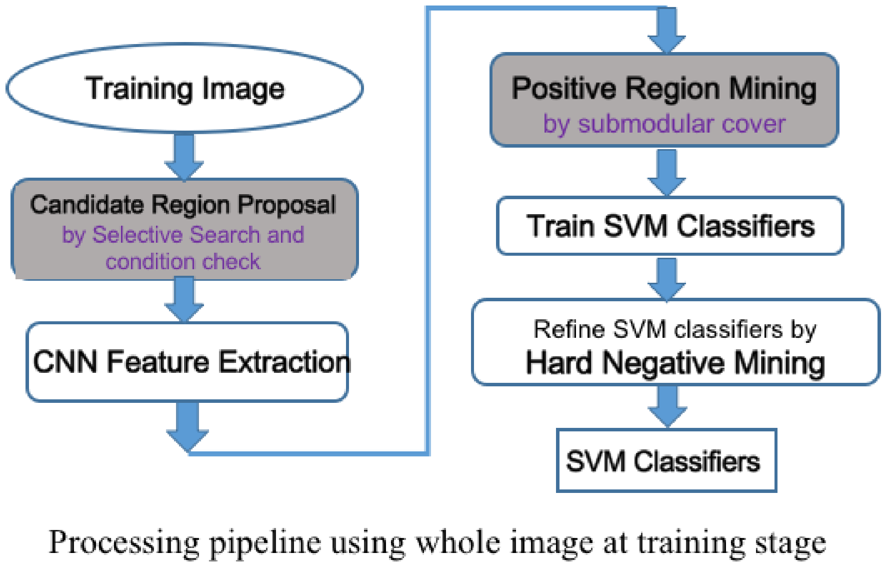

Training Image
Candidate Region Proposal
by Selective Search and condition check
CNN Feature Extraction
Positive Region Mining
by submodular cover
Train SVM Classifiers
Refine SVM classifiers by
Hard Negative Mining
SVM Classifiers
Processing pipeline using whole image at training stage

Figure 5.4: Overview of the training pipeline using whole image

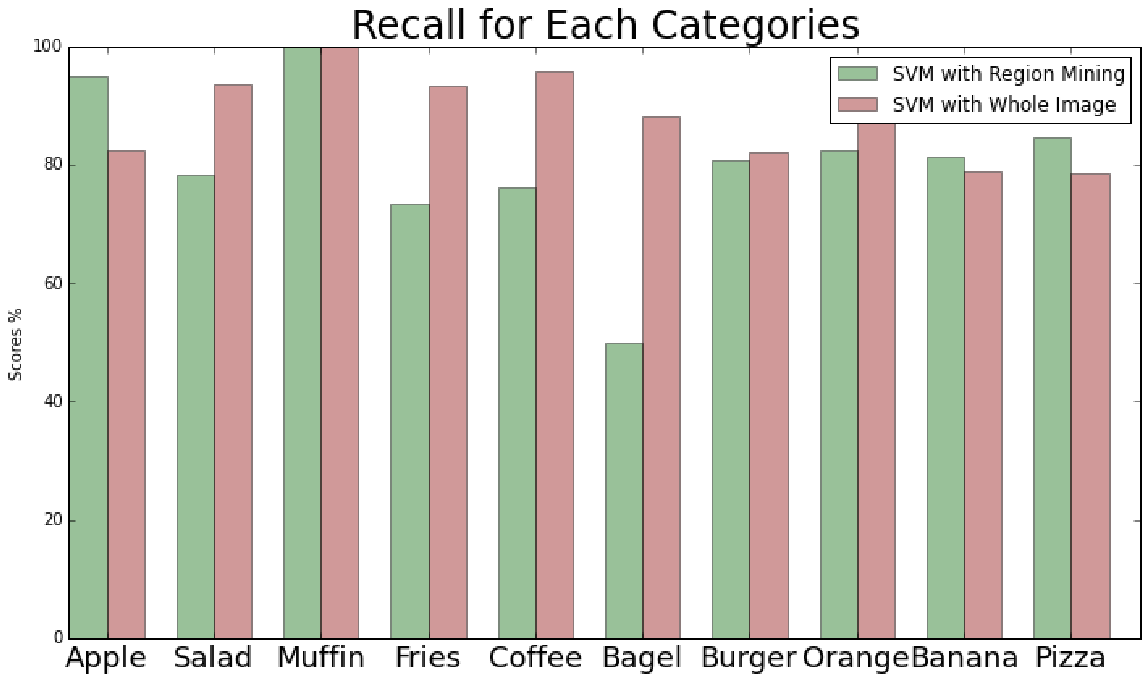
An average recall rate of 88.71% and precision rate of 56.31% is obtained by the comparison method. Compared to the results obtained with region mining, it has a higher recall rate but a much lower precision rate. We argue that the high recall rate is mostly due to the classifiers which always output many labels for one image and therefore increase its recall rate according to Equation 5.1. The classifiers trained without region mining produce low precision rate because many false predictions are computed.

We compare the recall rate and the precision rate of our proposed pipeline and the method using whole images as positive regions without region mining. The scores for each food category obtained are listed in Figure 5.5. From this figure we could see that even though the recall

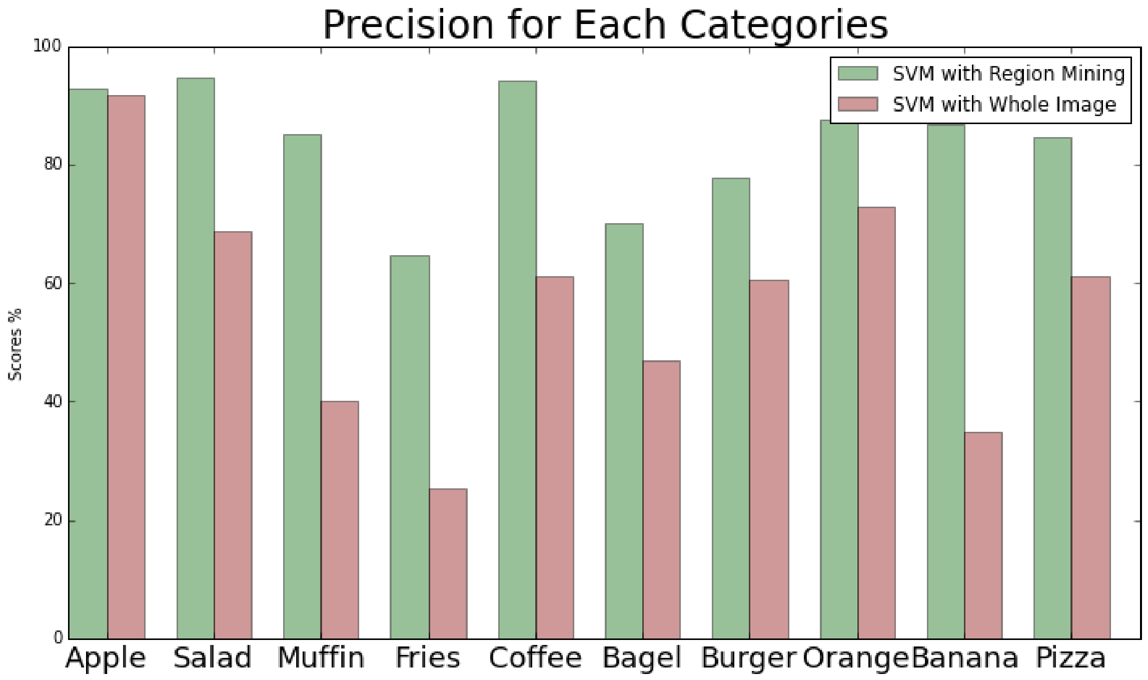
Table 5.4: Result of comparison experiment: recall and precision rate on the test dataset without region mining

Categories	**Num**	**Recall**	**Precision**
Coffee	40	82.50%	91.67%
Apple	47	93.62%	68.75%
Salad	18	100.00%	40.00%
Muffin	15	93.33%	25.45%
Fries	23	95.65%	61.11%
Bagel	17	88.24%	46.88%
Burger	28	82.14%	60.53%
Orange	17	94.12%	72.73%
Banana	19	78.95%	34.88%
Pizza	14	78.57%	61.11%
Average	24	88.71%	56.31%

rates of the two methods are comparable, the precision rates of our method are much higher than the method using a whole image as one region.



The recall rates of our pipeline and the method using whole image as one region



The precision rates of our pipeline and the method using whole image as one region

Figure 5.5: Comparison of the recall (top) and the precision (bottom) of our proposed pipeline and the method using whole image as one region.

5.4 Discussion

In this section, we discuss the performance of our proposed method and address the importance of region mining and hard negative mining in our processing pipeline. We also discuss the potential improvement that can be made on our proposed method.

5.4.1 Discussion on the Proposed Processing Pipeline

Our experiments show that the proposed method described in Chapter 3 works effectively well on both localizing and predicting the food items in the image. Each step in our processing pipeline plays its own importance.

Region mining is an important step in object detection. It discovers discriminative regions which helps distinguish each type of food from the others. Regions that belong to the background or would possible confuse the classifier are discarded. Classifiers trained on regions with high discriminative power are much more robust and accurate. In our proposed method, we use the submodular cover algorithm to select the discriminative regions from a large amount of proposed regions. Region mining improves the accuracy of classifiers for each food category because the classifiers are trained on carefully selected positive samples. On the contrary, when training classifiers using the whole image as the positive region without region mining, the precision accuracy rate is very low. This is because the individual classifier for each category trained on the whole image is not accurate. Thus, there are many false positive detections at the testing phase.

Mining the hard negative regions is another important step in our proposed method. This processing step solves the problem of imbalanced positive and negative samples at training stage. As a result, it helps to improve the recall rate of our detection. Classifiers trained with the data most of which are negative examples will predict “NO” most of the times, resulting in very low recall rate. The step of mining the hard negatives only select the hardest samples from the negative sample set and thus can balance the positive/negative samples in training set.

Non maximum suppression is also very important in accurately localizing the bounding box. We improve the standard NMS by combining the scores of the bounding boxes and the area information to get more accurate location of the bounding boxes for each food item. We select regions with top 5 SVM scores, and only keep the one with the largest area to be the final bounding box.

5.4.2 Discussion on Inaccurate Cases

Figure 5.3 shows some examples of inaccurate detection. Based on our observation, there are two main reasons for these inaccurate detection results:

1. Too much variance in food appearance. For example, the color, size, shape of muffin vary too much. In some images muffin appears to be round shaped and dark brown color, some appears to be square shaped and black chocolate color. Region mining procedure in our proposed pipeline is based on the assumption that object regions of images belonging to the same class will cluster closely together in feature space. If the appearance varies too much, it is difficult to obtain meaningful positive regions from region mining. Without discriminative positive regions, highly robust and accurate classifiers can not be learned, resulting in inaccurate detection.

2. Feature extractor is not fine-tuned on food dataset. The feature extractor we used is the $fc7$ layer of AlexNet pretrained on ImageNet rather than on food dataset. Previous work [32] has pointed out that color feature is more important to food recognition. Thus, a feature extractor fine-tuned on food dataset would greatly improve the feature to represent the most important information for recognition.

5.4.3 Discussion on Methods Comparison

Based on the experiments, we show that the approach using the whole image as one region to train a SVM classifier does not work well. The precision of the classifier is very low due to its inaccurate classification for each category. This can be explained by two reasons. Firstly, even though the collected training images contains only one food item, there are many other objects such as tables, wrapping paper and various other backgrounds in those images. Features extracted with those noise included could significantly decrease the performance of the SVM classifier as we show later in our experiments. Secondly, we have only around 200 training images per category, this number of training data is not big enough to train an accurate classifier. Even if we could collect a lot of training data to mitigate this, we still could not get an accuracy classifier due to the reason mentioned.

5.4.4 Discussion on Potential Improvements

One issue with our proposed method is that it requires a lot of computations, compared with the simpler using the whole image approach. The extensive computation happens mainly in the region mining step, which needs to compute the distance of features between proposed regions. The distance can be computed independently, thus it would be easy to parallelize the computation. We use the vectorized code for effective computation. This could be further optimized with the parallel computation on GPU.

The other issue is that we could use better evaluation criterion on our results if we have the ground truth bounding boxes on the test dataset. This evaluation method actually favors more for using the whole image as one region, as the evaluation criterion is insensitive to locations. Actually we get very promising results on locating the bounding boxes of the food items, which could be see in the Figure 5.2 and Figure A.1 to A.5.

Chapter 6

Conclusions and Future Work

We develop a method for multiple food detection with minimum supervision. In this chapter, we summarize our work and highlight the conclusions that can be drawn from our experiment. We also discuss the possible future work which may extend from this thesis.

6.1 Conclusions

In this thesis, we propose an effective method to detect multiple food items with minimum supervision. Our experiments are conducted on a dataset of 10 food categories, where images containing only a single food item are used as training set and images containing multiple food items are used as test set. Our proposed method learns from single food item images using techniques including CNN features, region mining, classifier training and hard negative mining. At training stage, our method discovers the discriminative regions to construct a set of positive regions for initial detector training and then refines the detector by mining hard negatives. At testing stage, we use non-maximum suppression to refine the detection results. We obtain an average precision rate of 83.78% of the prediction. Without any ground-truth bounding boxes at training stage, our algorithm can localize the objects on test images reasonably well, demonstrated by figures with detected bounding boxes shown in Appendix A. We also conduct experiments to demonstrate the importance of region mining in our method. Classifiers trained by the whole image without region mining have much lower precision rate on test dataset, indicating that object detectors should be trained on selected discriminative regions to obtain good results.

6.2 Future Work

Our proposed method can easily extend for a larger dataset containing more food categories for real application on smartphone. Since only CPU computation is needed, porting this implementation to mobile application is possible. By integrating the code to mobile health application, user could easily log their meal by just taking one picture of all the food items they eat.

Another future work is on food image segmentation, which can be used to estimate the portion size of food items. With the predicted bounding boxes obtained by our method, it could be a good start point for segmentation algorithms such as Graph Cuts [8] and Grabcut [47] to precisely segment the food items. Based on the segmentation, we can estimate the volume of food item to approximate the calories of food items in the meal photo.

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE, 2010.

[2] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2189–2202, 2012.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

[4] Vinay Bettadapura, Edison Thomaz, Aman Parnami, Gregory D Abowd, and Irfan Essa. Leveraging context to support automated food recognition in restaurants. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 580–587. IEEE, 2015.

[5] Marc Bosch, Fengqing Zhu, Nitin Khanna, Carol J Boushey, and Edward J Delp. Combining global and local features for food identification in dietary assessment. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1789–1792. IEEE, 2011.

[6] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[7] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014*, pages 446–461. Springer, 2014.

[8] Yuri Y Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Computer Vision, 2001. ICCV 2001.*

Proceedings. Eighth IEEE International Conference on, volume 1, pages 105–112. IEEE, 2001.

[9] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

[10] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6, 2004.

[11] Mei Chen, Kapil Dhingra, Wen Wu, Lei Yang, Rahul Sukthankar, and Jie Yang. Pfid: Pittsburgh fast-food image dataset. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 289–292. IEEE, 2009.

[12] Yizong Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799, 1995.

[13] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[14] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[17] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[18] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[19] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[20] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[21] Satoru Fujishige. *Submodular functions and optimization*, volume 58. Elsevier, 2005.

[22] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015.

[23] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[24] Bharath Hariharan. Beyond bounding boxes: Precise localization of objects in images. 2015.

[25] Haibo He, Edwardo Garcia, et al. Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284, 2009.

[26] Hajime Hoashi, Taichi Joutou, and Keiji Yanai. Image recognition of 85 food categories by feature fusion. In *Multimedia (ISM), 2010 IEEE International Symposium on*, pages 296–301. IEEE, 2010.

[27] Jan Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals? *arXiv preprint arXiv:1502.05082*, 2015.

[28] Nathalie Japkowicz et al. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI workshop on learning from imbalanced data sets*, volume 68, pages 10–15. Menlo Park, CA, 2000.

[29] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014.

[30] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2015-08-26].

[31] Taichi Joutou and Keiji Yanai. A food image recognition system with multiple kernel learning. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 285–288. IEEE, 2009.

[32] Hokuto Kagaya, Kiyoharu Aizawa, and Makoto Ogawa. Food detection and recognition using convolutional neural network. In *Proceedings of the ACM International Conference on Multimedia*, pages 1085–1088. ACM, 2014.

[33] Yoshihiro Kawano and Katsuki Yanai. Real-time mobile food recognition system. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 1–7. IEEE, 2013.

[34] Yoshiyuki Kawano and Keiji Yanai. Food image recognition with deep convolutional features. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 589–593. ACM, 2014.

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

[36] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006.

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

[38] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[39] Tomasz Malisiewicz, Abhinav Gupta, Alexei Efros, et al. Ensemble of exemplar-svms for object detection and beyond. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 89–96. IEEE, 2011.

[40] Yuji Matsuda, Hajime Hoashi, and Keiji Yanai. Recognition of multiple-food images by detecting candidate regions. In *Multimedia and Expo (ICME), 2012 IEEE International Conference on*, pages 25–30. IEEE, 2012.

[41] Yuji Matsuda and Keiji Yanai. Multiple-food recognition considering co-occurrence employing manifold ranking. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2017–2020. IEEE, 2012.

[42] Austin Meyers, Nick Johnston, Vivek Rathod, Anoop Korattikara, Alex Gorban, Nathan Silberman, Sergio Guadarrama, George Papandreou, Jonathan Huang, and Kevin P Murphy. Im2calories: towards an automated mobile vision food diary. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1233–1241, 2015.

[43] Parisa Pouladzadeh, Shervin Shirmohammadi, and Rana Al-Maghrabi. Measuring calorie and nutrition from food image. *Instrumentation and Measurement, IEEE Transactions on*, 63(8):1947–1956, 2014.

[44] Ali S Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

[45] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[46] Rasmus Rothe, Matthieu Guillaumin, and Luc Van Gool. *Non-maximum suppression for object detection by passing messages between windows*. Springer, 2015.

[47] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3):309–314, 2004.

[48] Henry A Rowley, Shumeet Baluja, Takeo Kanade, et al. *Human face detection in visual scenes*. Carnegie-Mellon University. Department of Computer Science, 1995.

[49] Jorge Sanchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fish vector: Theory and practice. *International Journal of Computer Vision*, 105(3):222–245, 2013.

[50] Hyun Oh Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, and Trevor Darrell. On learning to localize objects with minimal supervision. *arXiv preprint arXiv:1403.1024*, 2014.

[51] Kah-Kay Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(1):39–51, 1998.

[52] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[53] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[54] Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886. IEEE, 2011.

[55] Chong Wang, Weiqiang Ren, Kaiqi Huang, and Tieniu Tan. Weakly supervised object localization with latent category learning. In *Computer Vision–ECCV 2014*, pages 431–445. Springer, 2014.

[56] Laurence A Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

[57] Keiji Yanai and Yoshiyuki Kawano. Food image recognition using deep convolutional network with pre-training and fine-tuning. In *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*, pages 1–6. IEEE, 2015.

[58] Shulin Yang, Mei Chen, Dean Pomerleau, and Rahul Sukthankar. Food recognition using statistics of pairwise local features. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2249–2256. IEEE, 2010.

[59] Weiyu Zhang, Qian Yu, Behjat Siddiquie, Ajay Divakaran, and Harpreet Sawhney. snap-n-eat food recognition and nutrition estimation on a smartphone. *Journal of diabetes science and technology*, 9(3):525–533, 2015.

[60] Fengqing Zhu, Marc Bosch, TusaRebecca Schap, Nitin Khanna, David S Ebert, Carol J Boushey, and Edward J Delp. Segmentation assisted food classification for dietary assessment. In *IS&T/SPIE Electronic Imaging*, pages 78730B–78730B. International Society for Optics and Photonics, 2011.

[61] Fengqing Zhu, Marc Bosch, Insoo Woo, SungYe Kim, Carol J Boushey, David S Ebert, and Edward J Delp. The use of mobile devices in aiding dietary assessment and evaluation. *Selected Topics in Signal Processing, IEEE Journal of*, 4(4):756–766, 2010.

Appendix A

More Detection Results on Test Images

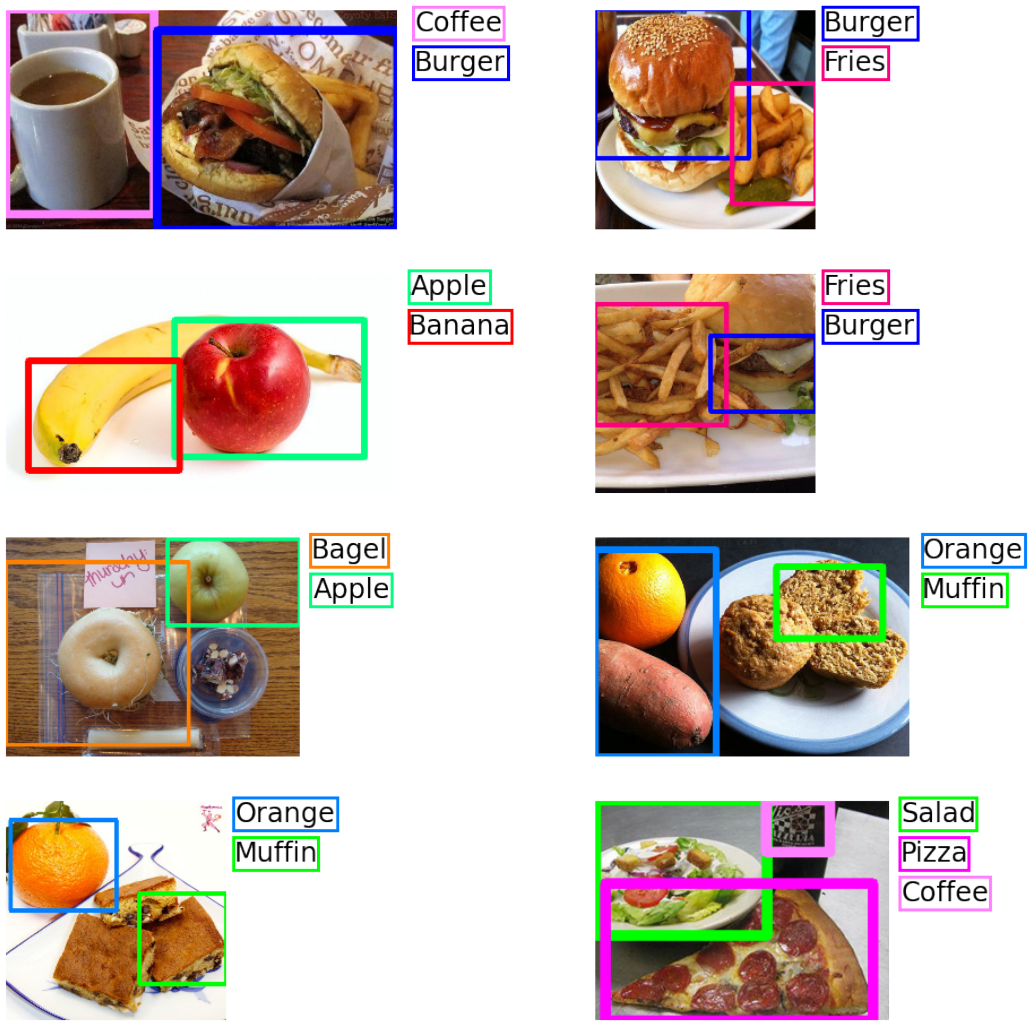


Figure A.1: Detection results on the test images (cont. 1)

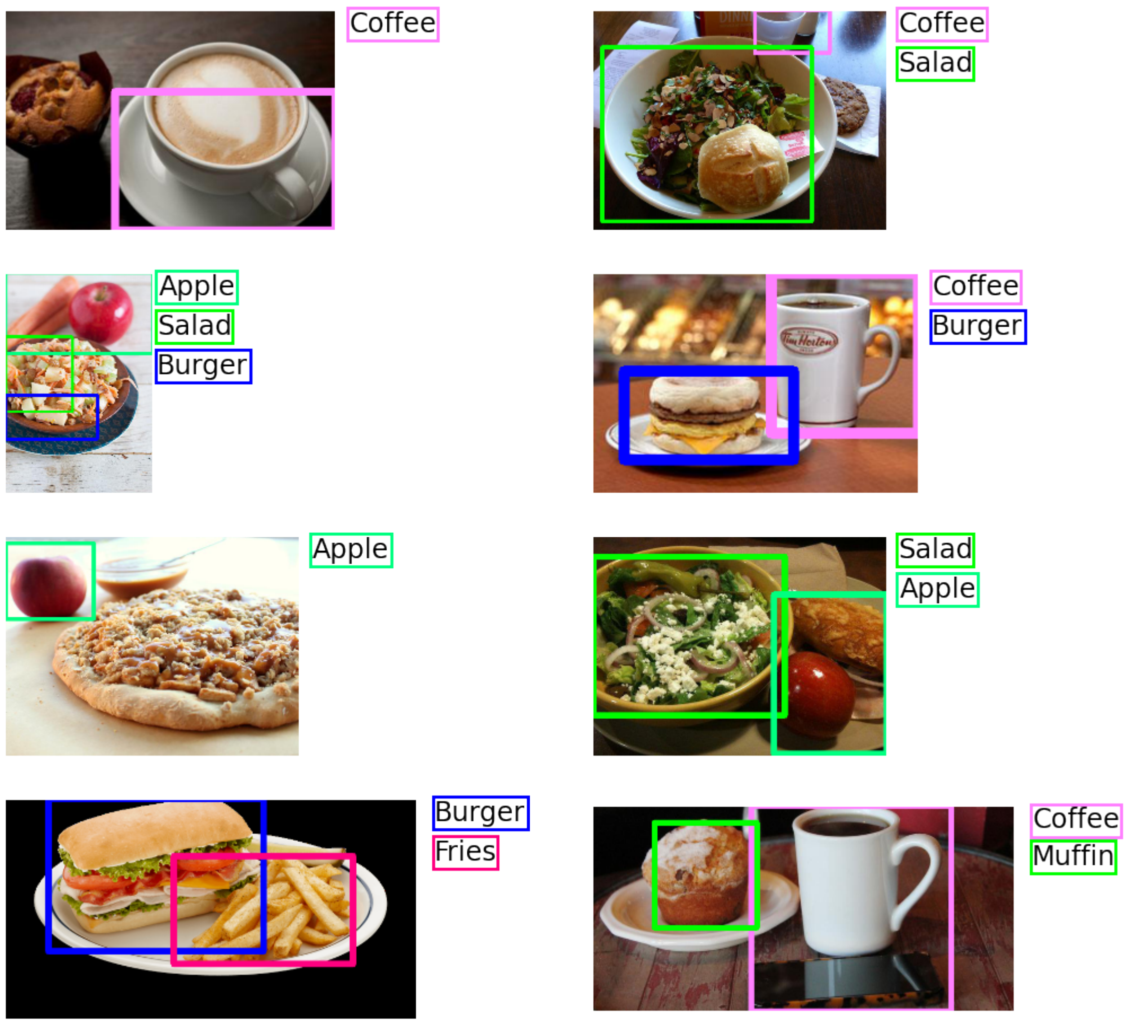


Figure A.2: Detection results on the test images (cont. 2)

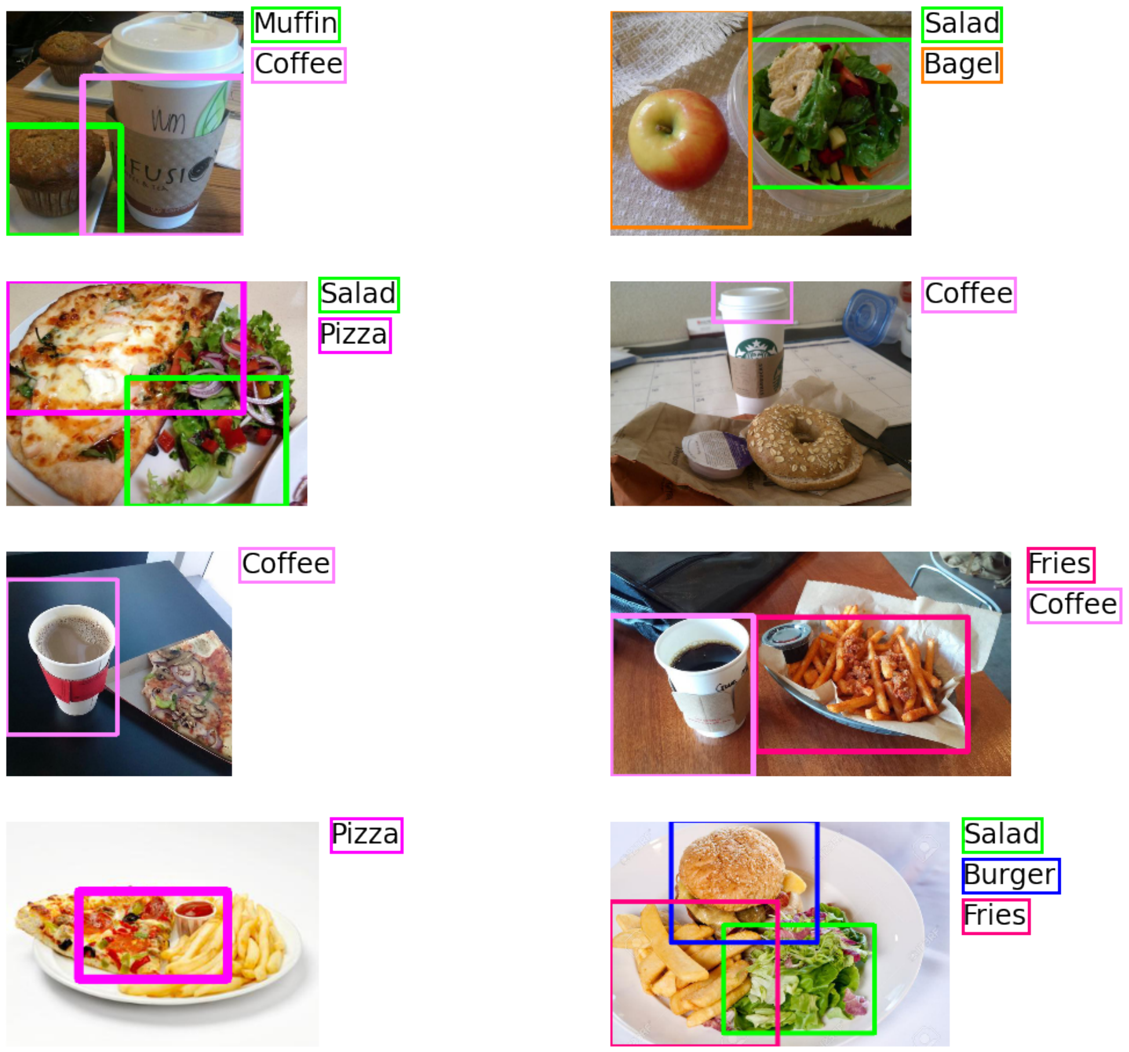


Figure A.3: Detection results on the test images (cont. 3)

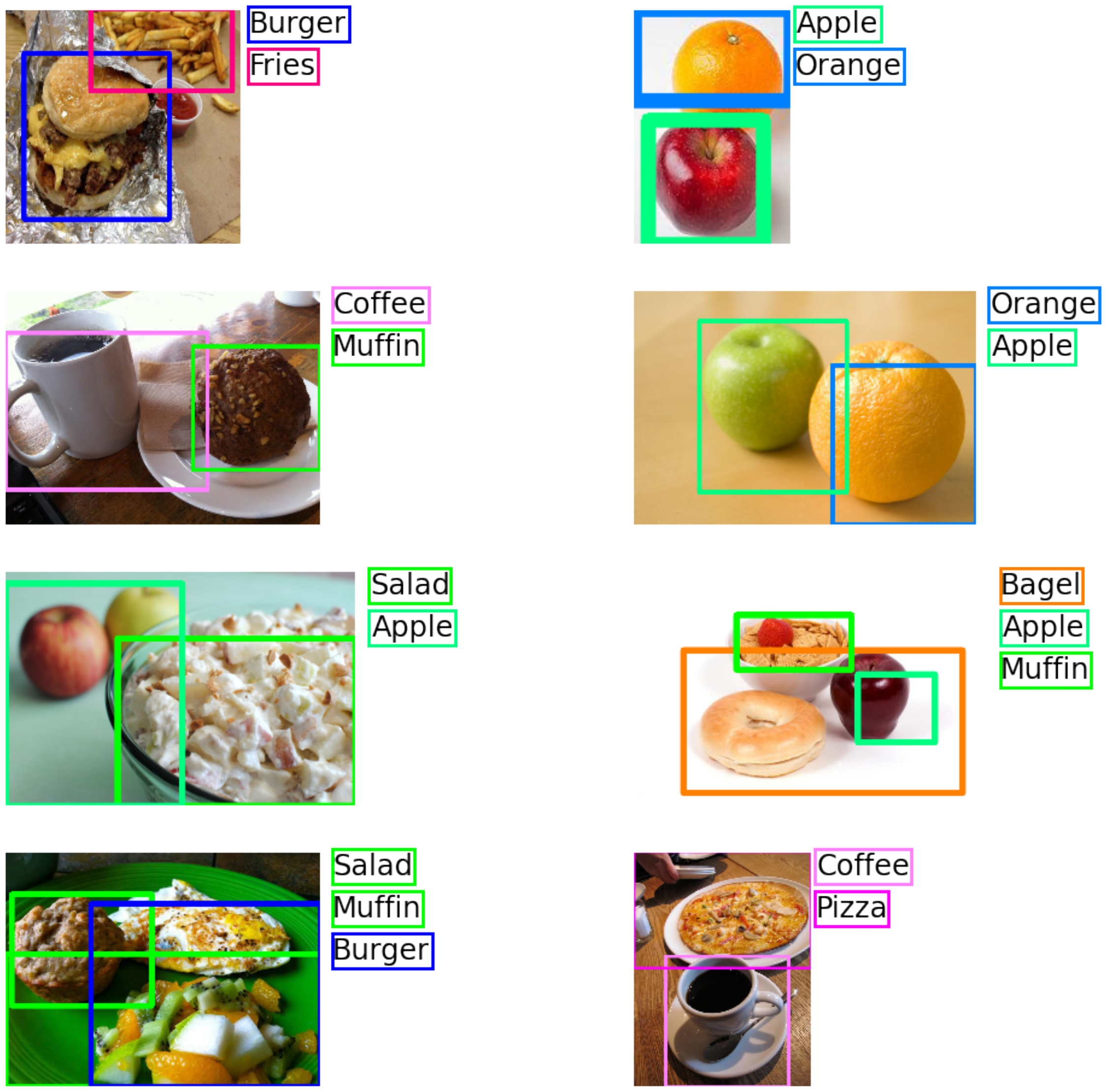


Figure A.4: Detection results on the test images (cont. 4)

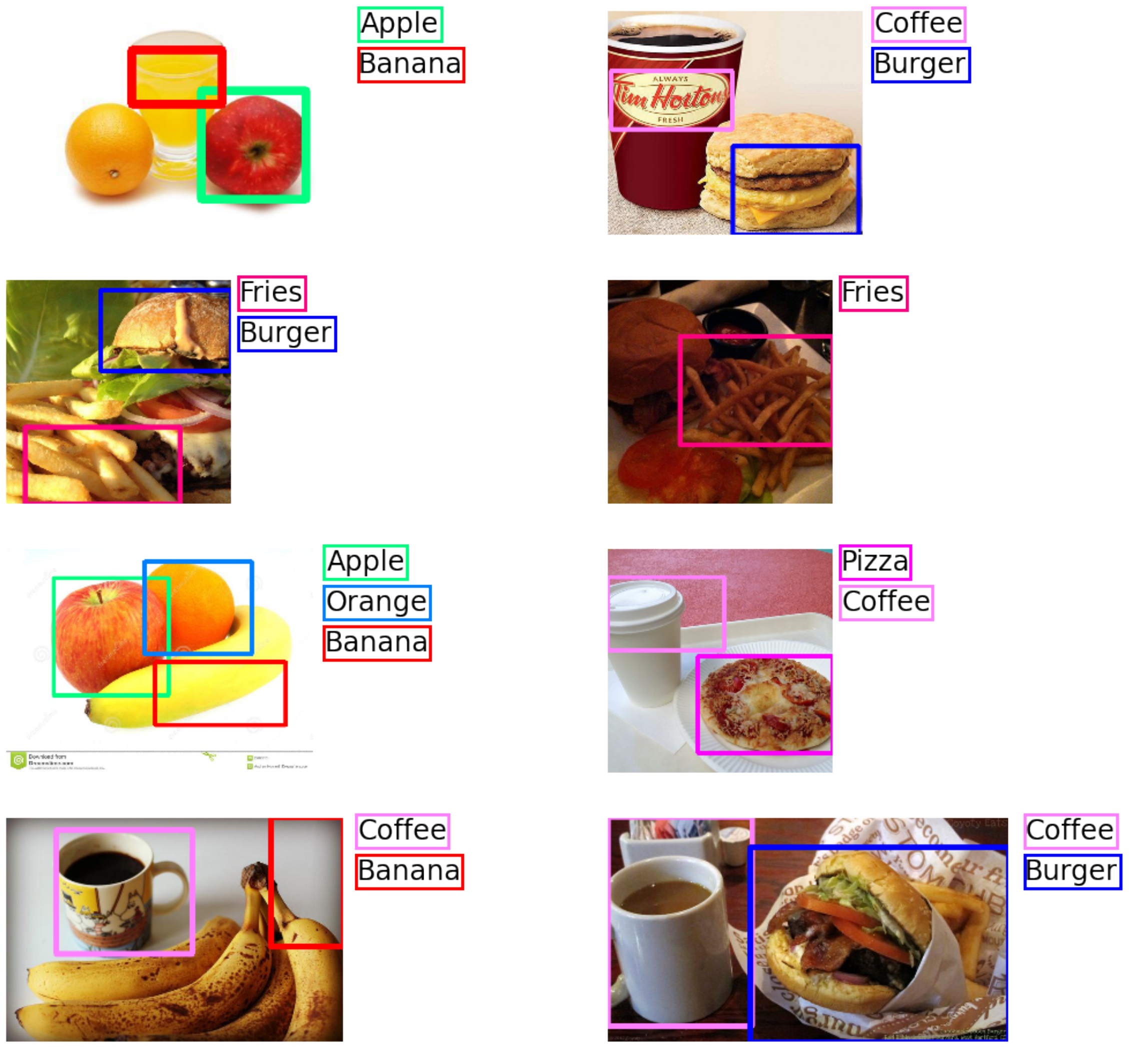


Figure A.5: Detection results on the test images (cont. 5)

Curriculum Vitae

Name:	Renfeng Liu
Post-Secondary Education and Degrees:	Huazhong University of Science and Technology Hubei, China 2002 - 2006 B.E. Software Engineering
	University of Western Ontario London, ON, Canada 2014 - 2015 M.Sc. Computer Science
Related Work Experience:	Teaching Assistant and Research Assistant The University of Western Ontario 2014 - 2015