Food Recognition and Nutrition Analysis using Deep CNNs

Bojia Qiu

School of Computer Science,
McGill University, Montreal, Canada

A thesis submitted to McGill University
in partial fulfillment of the requirements of the degree of

Master of Science

July 2019

2019/07/03

Abstract

Food is essential for human life. People nowadays would love to keep track of what they eat everyday in order to understand eating habits and patterns. New nutrition assessment tools based on modern techniques enable a new world of possibilities for understanding the culinary culture, exploring human nutrition patterns, guiding their daily recipe and improving their health. In the last few years, both recent breakthroughs in computer vision and deep learning field are transforming the way we analyze food data. However, due to the multiformity of food items as well as the inefficiency of detection algorithm, food related image is usually hard to recognize and slow to detect.

In this thesis, we develop an automatic nutrition analysis system to detect and recognize food items from daily meal images. Specifically, we build a three-step system to recognize multiple-food images by detecting candidate regions and classifying them using deep convolutional neural networks. In the first step, the automatic system generates multiple region of proposals from the input images by applying the Region Proposal Network (RPN) from Faster R-CNN model. Then it classifies each region of proposals by mapping them into feature maps, and classify them into different food categories, as well as locating them in the original images. Finally, we analyze the nutritional ingredients from the images and generate a dietary assessment report by calculating the amount of calories, fat, carbohydrate and protein.

Additionally, with the state-of-the-art and proposed models, we conduct thorough experiments using two popular datasets, which are UEC-FOOD100 and UEC-FOOD256. We also generate a new type of dataset about food items based on FOOD101 with bounding. The model is evaluated through different evaluation metrics. The experimental results show that our system is able to recognize the food items accurately and generate the dietary assessment report efficiently, which will benefit the users with a clear insight of healthy dietary and feasible advice.

Résumé

La nourriture est essentielle à la vie humaine. Avec la croissance rapide des journaux de nutrition, les gens aimeraient de nos jours garder une trace de ce qu'ils mangent tous les jours afin de comprendre les habitudes et les habitudes alimentaires. De nouveaux outils d'évaluation de la nutrition basés sur des techniques modernes ouvrent de nouvelles perspectives pour comprendre la culture culinaire, explorer les modèles de nutrition humaine, guider leur recette quotidienne et améliorer leur santé. Ces dernières années, les avancées récentes en vision par ordinateur et en apprentissage en profondeur ont transformé la façon dont nous analysons les données sur les aliments. Cependant, en raison de la multiplicité des produits alimentaires et de l'inefficacité de l'algorithme de détection, l'image liée aux aliments est généralement difficile à reconnaître et lente à détecter.

Dans cette thèse, nous développons un système d'analyse de nutrition automatique pour détecter et reconnaître les aliments à partir d'images de repas quotidiennes. Plus précisément, nous visons à construire le système en trois étapes pour reconnaître des images multi-aliments en détectant les régions candidates et en les classant à l'aide de réseaux de neurones à convolution profonde. Dans la première étape, le système automatique génère plusieurs régions de propositions à partir des images d'entrée en appliquant le réseau de proposition de région (RPN) à partir du modèle Faster R-CNN. Ensuite, il classe chaque région de proposition en les cartographiant respectivement dans des cartes de caractéristiques, puis en les classant dans différentes catégories d'aliments, ainsi qu'en les localisant dans les images originales. Après cette étape, nous avons terminé notre tâche de détection et reconnu les aliments dans l'image. La dernière étape consiste à analyser les ingrédients nutritionnels à partir des images et à générer un rapport d'évaluation alimentaire en calculant la quantité de calories, de lipides, de glucides et de protéines.

De plus, avec les modèles avancés et proposés, nous menons des expériences approfondies en utilisant deux jeux de données populaires, UEC-FOOD100 et UEC-FOOD256. Nous générons également un autre jeu de données sur les produits alimentaires avec des boîtes de délimitation et effectuons plusieurs expériences. Les résultats sont évalués à l'aide de différents paramètres d'évaluation. Les résultats expérimentaux montrent que notre système est capable de reconnaître les aliments avec précision et de générer efficacement le rapport d'évaluation alimentaire, ce qui profitera aux utilisateurs grâce à des informations claires sur un régime alimentaire sain et des conseils réalistes.

Acknowledgement

I would like to thank all persons who have ever supported me during my graduate studies. Foremost, I would express my sincere gratitude to my supervisor, Professor Xue Liu, for his expert advice and continuous support during my study at McGill. Without his help, I cannot have such wonderful graduate life and meet such brilliant people here. I am grateful for the inspiration he provided that helped me choosing this interesting research topic. His advices and guidance helped me progress and led me to conduct better research.

I would also like to thank Landu Jiang and other members in CPSlab, for taking time to help me and offering me innumerable valuable suggestions on statistical inference, neural network and deep learning.

To my collage friends and colleagues, I feel genuinely blessed and fortunate to have you all. I received too much and learned a lot from you.

Last but not least, I would like to thank my mum, who supports me unconditionally and endlessly. Thank you mum, thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter first presents the background and possible challenges of food recognition and nutrition analysis problems. Then it summarizes the contributions we made in this thesis. In the end, it provides the organization of the remaining chapters.

1.1 Background

The World Health Organization (WHO) defines obesity as an "abnormal or excessive fat accumulation that may impair health", and states that "the fundamental cause of obesity and overweight is an energy imbalance between calories consumed and calories expended". The obesity epidemic has been growing steadily across the whole world, and so far not a single country has been able to reverse it. Worldwide, there are more than 1.9 billion adults were overweight in 2016. From [1] we know that the prevalence of obesity was 39.8% and affected about 93.3 million of US adults in 2015 to 2016. The estimated annual medical cost of obesity in the United States was 147 billion in 2008 US dollars; the medical cost for people who have obesity was $1,429 higher than those of normal weight [2]. And obesity-related conditions include heart disease, stroke, type 2 diabetes and certain types of cancer that are some of the leading causes of preventable, premature death. The causes for obesity could be various, such as genetics reason, certain medications, emotional factors like stress, less exercise and poor sleep. Among all the causes, weight gain is largely a result of what they eat and how they eat, which is their eating behavior.

According to [3, 4], calories as well as other nutrition ingredients like fat, carbohydrate and protein are measures of energy. The amount of having them each day decides whether a person is having a healthy diet. Therefore, more and more people want to keep track of what they eat and the amount of nutrition contents they have everyday. Accurate estimation of dietary caloric intake will be more important than ever to live a healthy style.

Besides, the explosion of data enhances the rapid development of Internet of Things (IoT) [5], social network and mobile networks. Nowadays, people are willing to record, upload and share food images on the websites like Yelp [6], Dianping [1] and Yummly [2], leading to huge-scale food data. It is now more convenient than ever to find a huge amount of images and videos related to food, and more necessary than ever to develop tools for detecting and recognizing food from images automatically in order to take advantage of the boost of data. What's more, these tools can also promote the analysis of the nutritional ingredients from receipts and track the personal habit of eating and drinking.

Currently, there are three kinds of most common methods [7] manually used to assess dietary intake.

[1]http://www.dianping.com/
[2]https://www.yummly.com/

The first method uses diet records. Subjects record all food and beverage consumed over three consecutive day (two weekdays and one weekend day). The consumed items can be measured using a scale or other household items. Trained staff must provide detailed instructions on how to record intake and the completed records need to be entered into a software program, such as Nutrition Data System for Research (NDSR) [8], for analysis. The second one is called 24-hour recall. Subjects are asked to report all foods and beverages consumed in the past 24 hours. This can be done via telephone or face-to-face interview. Trained staff must conduct the interview to prompt for detail, such as cooking methods and portion sizes. The data need to be entered into a software program for analysis [9]. The third one is food frequency questionnaire (FFQ). Subjects report how frequently certain food and beverage items were consumed over a specific period of time (typically 1 year). Most FFQ versions ask portion size questions of every food item, as well as general questions about common cooking practice (e.g. type of fat typically added while cooking). Most FFQs are available in paper or electronic format and take about 1 hour to complete. Computerized software programs calculate nutrient intake by multiplying the reported frequency of each food by the amount of nutrient in a serving of that food. There are no data entry requirements for the study team. FFQs are usually validated for a period of 6 months or 1 year and repeat administration is not recommended for a period shorter than 6 months [10].

Although these methods are now the gold standard for reporting, there is at least one drawback that the user can not ignore - these methods still suffer from bias as the subject is required to estimate their dietary intake. Assessment of dietary intake by the participant can result in underreporting and underestimating of food intake [11]. In order to reduce participant bias and increase the accuracy of self-report, enhancements are needed to supplement the current dietary recalls.

In order to get rid of the bias, many semi-automatic or fully-automatic food analysis systems have been built. Due to the rapid spreading of mobile devices, there are more and more applications built in mobile system to enable food analysis. For example, a segmentation assisted food classification for dietary assessment system was proposed by Zhu et al. [12]. Their goal is to automatically determine the regions in an image where a particular food is located and correctly identify them, using traditional computer vision techniques. Another mobile cloud-based food calorie measurement system was proposed by Pouladzadeh et al. [13]. They used cloud Support Vector Machine (SVM) to train their system to recognize food and calculate the calories for each food.

With the rapid development of hardware device, different algorithms and machine learning techniques, especially in deep learning field, are enhanced in not only many industrial and professional processes, but also in everyday living. Nowadays, there are more self-driving cars on the road, artificial intelligence players who can play computer games, and even digital robots with empathy as our friends. Now we can almost rely on software to do our jobs since they are more accurate, more efficient and making less mistakes. Food recognition and nutrition analysis systems based on state-of-the-art neural networks show up in the same way as in other industries. The work from Pouladzadeh et al. [14] provided an assistive calorie measurement system to patients and doctors, to record and measure their daily calorie intake in an convenient and intelligent way.

In this thesis, we try to build an automatic food recognition and nutrition analysis system. With just one single photo taken by mobile devices, our system can recognize the food items and analyze the nutrition contents from the meal, as well as providing the user with a thorough dietary assessment report on what they have. With this automatic system, it will help the user to measure their daily intake and enable them in a long-term way to ensure their health.

1.2 Challenges

Although the idea of our project sounds not complicated, many challenges arise in real food image recognition and analysis. There are three main challenges in the research field of food analysis.

- **Classify one food per image.**

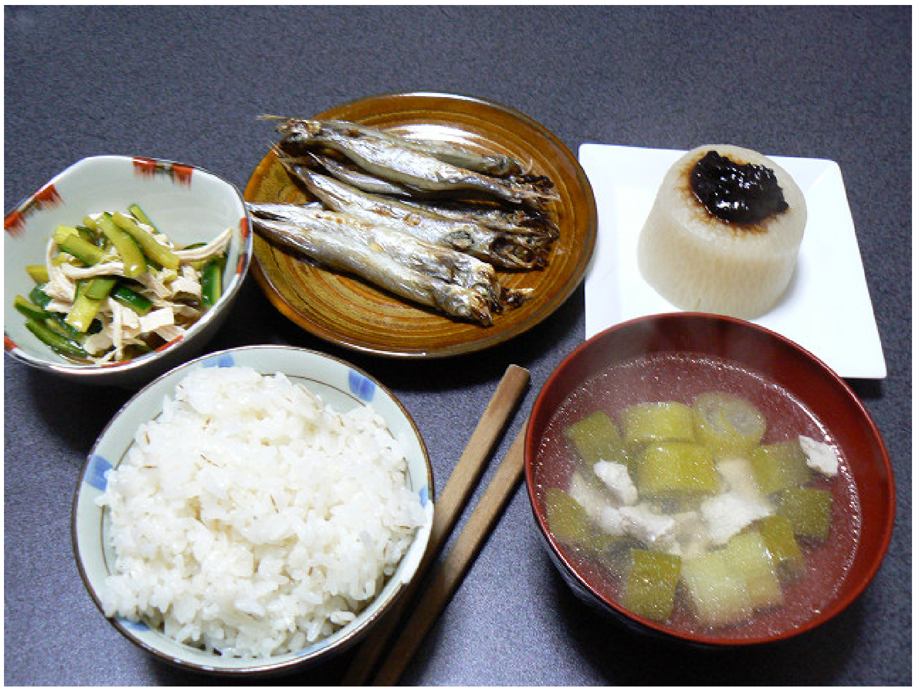


Figure 1.1 Example of the food image to analyze

Nowadays, most of the food analysis applications are using classification algorithm to classify foods from the image. Although classification method has a very high accuracy, it has one deficiency which make it unrealistic. That is it can only classify one object at a time. In order to analyze the calories and other nutrition contents in the food images, we first need to recognize multiple food items from a single image. As illustrated in the **Figure 1.1** (source: UEC-FOOD100 Dataset), the image may contain multiple food items from different categories. To recognize them all, we need an object detection system which can detect multiple foods from a single image at the same time, instead of a classification method. So the first challenge is to implement a food detection system instead of a food classification system, for multiple food images.

- **The duration of recognizing the food items.**

The second challenge is that the time consumed during the recognition process is important for an application designed for real time usage. It would be very annoying if the user has to wait one or two minute to know what is the amount of caloric intake he/she just had. So the second challenge is to reduce the processing time for the detection and analysis system to get the result. Most of the detection systems are time-consuming, because of the detection algorithms and the different numbers of object in one image. Usually, the more objects in one image, the more time it may take. We need to design a system to process all the images with different numbers of object at the same speed, and cost less time.

- **Insufficient information of nutrition content in the dietary assessment system.**

  Most existing food analysis systems, often implemented as smart phone applications (e.g. MyFitnessPal [15], SHealth [16]), help users to keep track of their food intake. These systems can assist users in achieving dietary goals such as weight loss, allergy management or maintaining a healthy diet. And some other approaches that use mobile phone cameras to automatically recognize the food. But they only implemented the recognition step, without carrying out the food attributes measurement experiments. And also the number of classes is limited for the food to be recognized. Crowd sourcing [17] is also employed for the nutritional analysis of food items which makes the algorithm costly and inhibits it from widespread application in daily life. Here comes the third challenge, which is how to expand the number of classes for the food and analyze the nutrition contents more accurately.

1.3 Contribution

In this thesis, we build an automatic food recognition and nutritional analysis system using deep neural network models. Our model is evaluated in several existing datasets as well as the dataset built on our own. The contribution of this thesis is summarized as follows.

- We explore a deep model based approach for food recognition and nutrition analysis task. Specifically, we develop an automatic food recognition and dietary assessment system that takes a photo of food items as input. And output the amount of nutritional ingredients of each food items from the image. A thorough dietary assessment report will be generated based on what you have during the meal. Besides, our system can also suggest with a healthy diet if you manually input some basic information about yourself.

- We design and implement an food detection model based on deep convolutional neural network models (e.g. Faster R-CNN) for food detection and identification. First we use region proposal network to generate thousands of region proposals from one image. Then we apply the state-of-the-art deep convolutional neural network to extract the feature maps from each proposals, and classify them as different food items. We also apply a regression module to locate each food item in the image. With our proposed model, the time consumed as detecting step is 10 times shorter than previous baseline models.

- We generate a new food related dataset, named FOOD20-with-bbx, based on the data in FOOD101 dataset [18]. The FOOD101 dataset contains only categories information. However, we choose 20 categories from the original dataset, and manually label each food items in the image with bounding box information. With this information, our dataset can be used in the evaluation of detection models.

- We conduct extensive experiments to test our detection model as well as food analysis system. We perform detection experiments in UEC-FOOD100 [19] and UEC-FOOD256 [20] datasets, which are existing datasets with bounding box information. We evaluate the performance of different models by comparing the mean Accuracy Precision (mAP) and the detection speed. We also train our model with our own dataset FOOD20-with-bbx, to test the robustness of our model with a larger range of food categories.

1.4 Organization

The remaining chapters of this thesis are organized as follows:

In chapter 2, we present a thorough literature review on different approaches towards object detection. It includes geometry-based approach, color-based approach and region-based approach. Also we introduce several state-of-the-art food recognition and diet analysis systems.

In chapter 3, we shows the illustration of state-of-the-art food analysis systems, classified separately by web-based systems, mobile-based systems and image-based systems. We then analyze the motivation of building an automatic system. After that, we focus on the detail of region-based object detection approach. By discussing the detail of modern detection techniques, we expose the drawbacks of some traditional technical schemes such as selective search, and illustrate the development of the detection technique we use in this thesis.

In chapter 4, the structure of food analysis system is presented and illustrated. We first show the whole scheme of our food analysis system based on deep learning models. And then we illustrate the detailed implementation of each modules, which are detection module, classification module and dietary assessment module. We illustrate how each part works and demonstrate the input and output of our system.

In chapter 5, we covers the details about our experiment implementation and evaluations. We first introduce the existing dataset we use in this thesis, as well as a new dataset that we build to experiment with. Then we describe the environment and the evaluation metrics of our experiments. Finally, we conclude the overall performance of our system.

In chapter 6, we conclude this thesis and discuss possible future works.

Chapter 2

Related Work

In order to perform food recognition, the first step is to detect the food items from the image. This is where object detection plays its role at. In this chapter, we first discuss various algorithms that can be used for object detection. With a comprehensive understanding of object detection algorithms, we then review the related works for food recognition and dietary analysis.

2.1 Object Detection

2.1.1 Geometry-Based Approach

Object detection and localization is one of the most important topics in computer vision. One of the popular strategies is the geometric solutions, such as the interpretation tree [21] and geometric invariance [22]. The approach has been dominated by the discovery of analytic representations (models) of objects that can be used to predict the appearance of an object under any viewpoint and under any conditions of illumination and partial occlusion. The expectation is that ultimately a representation will be discovered that can model the appearance of broad object categories and in accordance with the human conceptual framework so that the computer can "tell" what it is seeing.

However, object geometric approaches have the limitation of being less reliable at low spatial resolutions and having expensive computation. What's more, geometry also has the lack of robustness to object occlusion and clustering. As a result, color-based objection detection methods came into play.

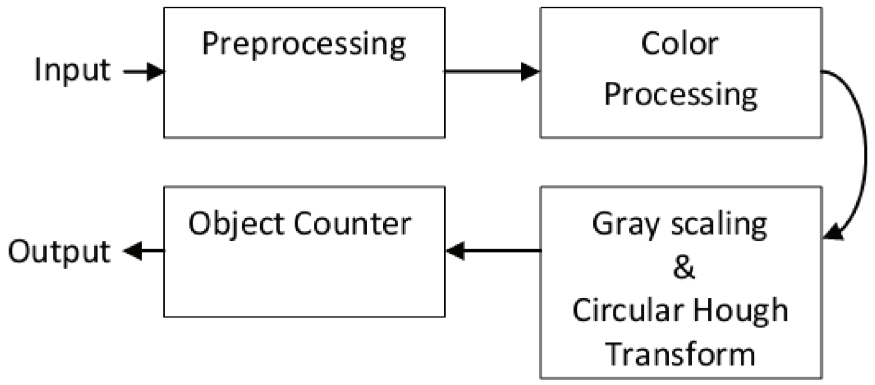


Figure 2.1 Mango recognition block diagram

2.1.2 Color-Based Approach

There are several methods being developed as color-based approaches. A simple and effective recognition scheme is to represent and match images on the basis of color histograms proposed by Swain and Ballard [23]. This method is called histogram backprojection which determines a confidence space after smoothing correspond to the object candidate. This work made a significant contribution in introducing color for object recognition. However, color proportion ratios are not robust to object occlusion. Vinod et al. [24] used histogram and discrete cosine transform (DCT) coefficients to represent object. The color histogram intersection is used to verify object presence. Matas et al. [25] used a color adjacency graph (whose nodes represent model colors and edges encode information about the adjacency of colors and their reflectance ratios) to identify object hypotheses and the color region adjacency for object match verification, and it can represent 3-dimensional deformable objects with perspective distortions. And later in 2010, Fan et al. [26] proposed an improved histogram backprojection approach to find the candidate target regions, and uses the weighted histogram intersection to verify the object presence and to give a cur to determine the scale and number of objects.

In 2012, [27] introduced the digital image processing techniques for object detection from complex background image. They discussed the methods on how to detect the mango from a mango tree. To achieve the objective, they implemented a four steps detection system based on color processing method shown in **Figure 2.1** (Source: [27]).

The images first are sent into the preprocessing block and perform image resize where set to 320×240 pixels. Then the resized image will directly go to the RGB adjustment where it will readjust the object color as lighter and darker all the other colors which are recognized as background. Next, color processing took place and the elimination of the unrelated color will take place and left the color of the object (mango). The elimination of the related object in here are by comparing the current pixel RGB value with the default RGB value of mango. If detected color is not related then changed the current pixel RGB value to 0 which is in black color.

However, the color-based approaches also suffer from the drawbacks since it is detecting the color histogram, the detector may not distinguish two objects with the same or similar color. Besides, if the image has many objects to be recognized, each with different color, the detector will only be capable of detecting one kind of the objects with the specific color. Because of these inefficiency, region-based object detection approaches has been proposed.

2.1.3 Region-Based Approach

The most common modern approaches in region-based approach is to scan the image for candidate objects and score each one. This is typified by the sliding-window object detection approach [28, 29], but is also true of most other detection schemes (such as centroid-based methods [30] or boundary edge methods [31]).

A typical scheme of using sliding-window object detection is to slide windows from left and right, and from up to down to identify objects using classification. To detect different object types at various viewing distances, it uses windows of varied sizes and aspect ratio, and then cuts out patches from the image according to the sliding windows. The patches are warped since many classifiers take fixed size image only. The warped image patch is then fed into a classifier or neural network to identify the class of the object in that specific region.

In 2010, Region-based segmentation and object detection was perform jointly from the study of [32], which proposed a new hierarchical approach facilitates both bottom-up and top-down reasoning about the image. Their hierarchical model enjoys three advantages: (1) Like multi-class image segmentation, their model uniquely explains every pixel in the image and group these into semantically coherent regions. (2) Like object detection, their model uses sophisticated shape and appearance features computed over candidate object locations with precise boundaries. (3) Their joint model over regions and objects allows context to be encoded through direct semantic relationships (e.g., "car" is usually found on "road").

2.2 Food Recognition and Dietary Analysis

On account of the comprehensive potential and value of the food detection and recognition, there are more and more researchers conducting experiments and researches toward the fields of food classification, detection and analysis. In this section, we introduce the evolution directions of food recognition and analysis.

2.2.1 Food recognition based on geometry features

Food category recognition has been a popular research area in nutrition study and computer vision for many years. However, food recognition is relative difficult because food items are deformable objects that may display significant variations in appearance. There are foods having a high intra-class variance (similar foods such as beef and steak look very different based on how to cook them), or low inter-class variance (different foods like fish and pork look very similar).

Approaches based on local features such as the SIFT [33] descriptor, global features such color histograms or GIST [34] features, and shape context [35] method have been proposed to recognize food items using geometry features. For example, Belongie [35] picks n pixels from the contours of a shape, obtaining $n-1$ vectors by connecting a pixel to the others, and uses these vectors as a description of shape at the pixel.

Besides, Felzenszwalb et al. [36] proposes techniquees to represent a deformable shape using triangulated polygons. Jiang et al. [37] proposes learning a mean shape of the object class based on the thin plate spline parametrization.

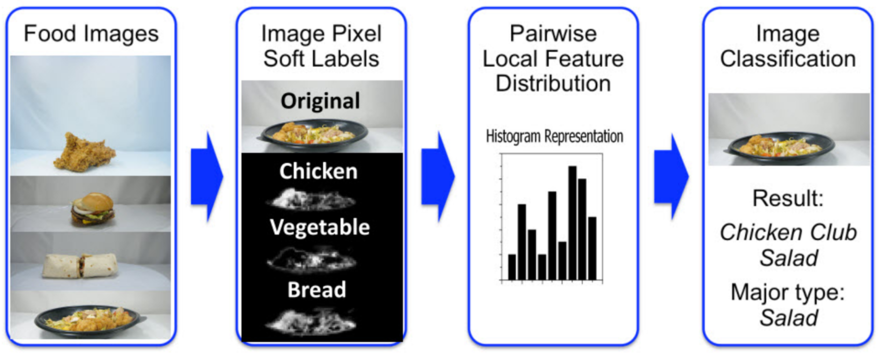


Figure 2.2 Framework of statistical-based approach

The geometry feature-based approaches work well for many items in object detection field. But there are two main problems makes it hard to detect some of the foods. One of the major reasons is that these approaches need to detect meaningful feature points like edges, counters and key points or landmarks, which is not available in food images because it is too precise. The other reason is that shape similarity in food images is hard to exploit since the shape of real foods is often quite without a clearly defined shape of form.

2.2.2 Food recognition based on statistical features methods

To overcome the problems described previously in geometry feature-based approaches, methods based on statistical features are proposed. These approaches does not only rely on the detection of features like edge or keypoints, but also detect local, statistical features like several pairs of pixels. The statistical distribution of pairwise local features effectively captures important shape characteristics and spatial relationships between food ingredients, thereby facilitating more accurate recognition.

Yang et al. [38] believed the key to recognizing food is to exploit the spatial relationships between different ingredients (such as meat and bread in a sandwich). So they propose a new multi-steps discriminative classifier working as the **Figure 2.2** (Source: [38]) shown. Each pixel in the food image is assigned a vector representing the probability with which the pixel belongs to each of nine food ingredient categories, using semantic texton forest [39]. The image pixels and their soft labels are used to extract statistics of pairwise local features, to form a multi-dimensional histogram. This histogram is passed into a multi-class SVM to classify the given image.

2.2.3 Food recognition based on machine learning methods

Later with the rapid development of traditional machine learning methodologies and neural network, more and more approaches based on machine learning algorithms appear in the field of food recognition.

A Bayesian framework approach was used in [40] to facilitates incremental learning for both food detection and food-balance estimation. Bossard et al. [18] proposed a novel approach that classification accuracy on the Food-101 test set of 50.67% by mining discriminative components using Random Forest [41]. Basically, the random forest is applied to cluster the superpixels of the training dataset. The discriminative clusters of superpixels are to train the components models. Other advanced classification techniques were also applied to this task, such as Improved Fisher Vectors (IFV) [42], Bag-of-Words Histogram (BOW) [43], Randomized Clustering Forests (RCF) [44] and Mid-Level Discriminative Superpixels (MLDS) [45].

As the ability of computing hardware is getting more and more stronger, convolutional neural network (CNN) is also widely used in food recognition and provides better performance than the convolutional methods. Deeper models has become the mainstream of food image detection and recognition. Kagaya et al. [46] applied CNN in food/non-food classification and achieved significant results with a high accuracy of 93.8%. They also trained the CNN for food recognition and the experimental results showed that the CNN outperformed all the other baseline classifical approaches by achieveing an average accuracy of 73.7% for 10 classes. Yanai et al. [47] fine-tuned the AlexNet model and achieved the best results on public food datasets so far, with top-1 accuracy of 67.7% for UEC-FOOD-256. [48] evaluates the effectiveness in classifying food images of a deep-learning approach based on the specifications of Google's image recognition architecture - Inception. Their architecture is a 54 layers CNN. Myers et al. [49] presented the Im2Calories system for food recognition which extensively used CNN-based approaches. The architecture of GoogLeNet [50] was used in their work and a pre-trained model was fine-tuned on Food-101.

Recently several authors [51–54] focused on analyzing which features and models are more suitable for the food recognition, and comply them into food analysis system to calculate the calories. [55] automatically estimate the food calories from a food image via simultaneous learning of food calories, categories, ingredients and cooking methods using multi-task convolutional neural networks. What's more, a food portion estimation method to estimate the energy from food images using generative adversarial networks was proposed by [56].

Although food recognition and nutrition contents analysis have been a popular field in recent year, there are still a few challenges to be solved. The first is that most of the works are dealing with image with only one food inside. They try to use classification method to recognize the food. The second challenge is the time consumed to detect the food items. Usually with detection models, it takes about 2 seconds to detect food items from the image.

In this thesis, we would like to address these drawbacks and propose an automatic food recognition and nutrition content analysis system to detect the food from images and generate dietary assessment reports for each meals.

Chapter 3

Design Consideration

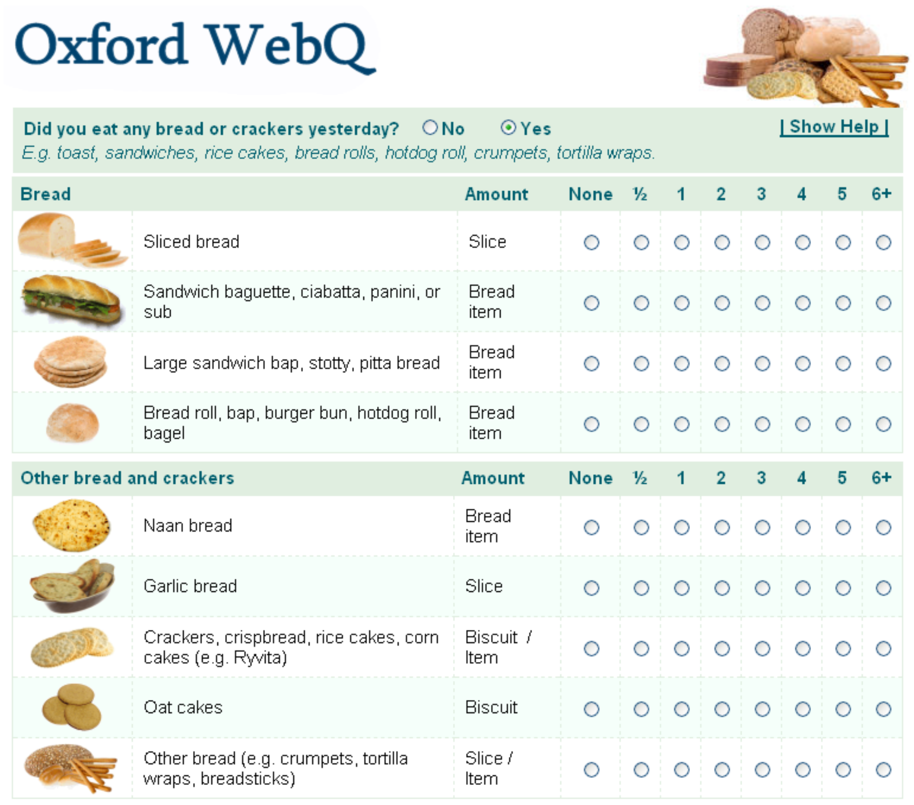
In this chapter, we show our design consideration of the food recognition and nutrition contents analysis system. We first introduce several commonly used food content analysis systems, discussing their efficiency and talk about the advantages and disadvantages for each of them. And give our motivation on why we want to build this system, and how we build it. Then we explore several techniques to build the system.

3.1 Current Food Analysis System

Based on a project [57] launched by The International Life Sciences Institute [1] (ILSI) Europe Dietary Intake and Exposure Task Force, they evaluated 43 new technology-based dietary assessment tools in total from 2011 to 2017, including web-based programs, mobile applications and wearable devices, in order to recommend general quality standards for future applications. And the results showing that most of the tools (79%) relied on self-reported dietary intakes. The main part (91%) used text entry and 33% used digital images to help identify foods. Only 65% had integrated databases for estimating energy or nutrients. Fewer than 50% contained any features of customization and about half generated automatic reports. Most tools reported on usability or reported validity compared with another assessment method (77%).

We introduction four types of technology categories and the representatives of each of them.

[1]http://ilsi.org/



Oxford WebQ

Did you eat any bread or crackers yesterday? No [✓ Yes] | Show Help |

E.g. toast, sandwiches, rice cakes, bread rolls, hotdog roll, crumpets, tortilla wraps.

Bread		Amount	None	½	1	2	3	4	5	6+
	Sliced bread	Slice	○	○	○	○	○	○	○	○
	Sandwich baguette, ciabatta, panini, or sub	Bread item	○	○	○	○	○	○	○	○
	Large sandwich bap, stotty, pitta bread	Bread item	○	○	○	○	○	○	○	○
	Bread roll, bap, burger bun, hotdog roll, bagel	Bread item	○	○	○	○	○	○	○	○

Other bread and crackers		Amount	None	½	1	2	3	4	5	6+
	Naan bread	Bread item	○	○	○	○	○	○	○	○
	Garlic bread	Slice	○	○	○	○	○	○	○	○
	Crackers, crispbread, rice cakes, corn cakes (e.g. Ryvita)	Biscuit / Item	○	○	○	○	○	○	○	○
	Oat cakes	Biscuit	○	○	○	○	○	○	○	○
	Other bread (e.g. crumpets, tortilla wraps, breadsticks)	Slice / Item	○	○	○	○	○	○	○	○

Figure 3.1 Example of web-based food analysis system

3.1.1 Web-based approaches

The web-based assessment tools have used similar approaches to standard golden methods such as FFQ, 24 hours recalls and dietary records. Some of the tools are hybrid methods like the Oxford WebQ [58], shown in **Figure 3.1** (Source: https://www.nutritools.org/tools/57) which is presented as a 24 hour recall but relies on reporting of 21 food categories over a 24 hours period.

Most of the web-based systems consist of three main areas: an administrator area, a researcher area and a participant area. The administrator area consists administrative information such as website declaration, website owner and website controller. The researcher area is often used to customize the project-specific contents like project name and logos, tailored invitation and reminder emails, set recall/diary/interview options, export food and nutrient analysis. The participant area includes the search function, portion size selection area, recipe builder, help, review and submit screens.

One of the most important aspect for the web-based system is to ensure the user is able to find foods they consumed in the food database. Some systems use food categories to aid searching. When the food list is limited, the closest matched food needs to be selected. This method enables the user to have a more accurate analysis.

Also, compared to the standard methods, web-based technologies can support portion size assessment, which is often hard in food analysis and dietary assessment. Most of the FQQ methods just assume standard portions while 24 hours recall methods with interviews may use household measures or books of photos. However, web-based methods can have different options for portion selection to support the user. These can include unit sizes, standard pack, food photographs, average portion sizes or entry of actual weights. For example, the Oxford WebQ uses standard categories of amounts of foods, with portion sizes described as a serving. For more detailed information of sizes of foods without a natural portion are given in the help section. Participants are expected to adjust their reporting of amounts in relation to the standard serving. Some of the web-based systems, which have had energy intake from a comparison method for relative or actual validity assessed are presented in **Table 3.1**

However, there are several drawbacks in web-based food analysis systems. One of the most annoying issues is that it gets really complicated when you have to open the computer, type in the URL of the system's website and fill up the questions every time finishing your meal. As a result, although once the users have used a web-based tool to record their diet they find the approach highly acceptable, the response rate in the research is pretty low. According to [59], in a test of repeated invitations to complete the Oxford WebQ, 53% of those invited completed the online version at least once; 66% completed it more than once but only 16% completed it on the requested four occasions. We need to make the system easy to use in order to help the user cultivate the habit of recording their meals. So it is important to figure out some attractive ways to keep the user active.

Table 3.1 Web-based tools for food analysis

Web-based systems	Feature	Comparison	Results
myfood24	Multipass recall designed for adolescents, adults, elderly. 45000 foods including brand names. 5669 food images.	Interviewer led MPR 75× 11-18 year olds	Mean intake 8368kJ / 2000$kcal$
Oxford WebQ	Web-based questionnaire. Twenty-one food groups with detail screens. Standard portion categories.	Interviewer led MPR for same day as Oxford WebQ 116 adults	Mean intake 8711kJ / 2082$kcal$
Food4Me	Web-based FFQ, based on EPIC Norfolk FFQ. 157 foods, for seven EU countries. Frequencies & portion pictures	EPIC Norfolk FFQ 113 adults	Mean intake 9857kJ / 2356$kcal$
NANA	Touch screen system aimed at elderly. Does not automatically code selected items. This is done later by a nutritionist	4 day food diary forty adults, mean age 72 years	Mean intake 8238kJ / 1696$kcal$

3.1.2 Mobile-based approaches

In order to overcome the drawback of complex operations in the web-based food analysis tools, there is growing interest in using smartphones to deliver nutrition interventions and to collect data on intakes, since mobile devices is much more convenient to use during day time. Some of the web-based tools may be optimized for used on mobile phones. Alternatively, there are also several mobile applications can be installed on the phone and require no internet access to operate and analysis. With such ubiquitous access to the internet and smartphones, self-monitoring of diet using a mobile phone has become a feasible option for many people.

There are many mobile applications like the My Meal Mate (MMM) [60] and My-FitnessPal [15] that monitors daily diets. For example, the MMM was developed using an evidence-based behavioural approach, and incorporates goal setting, self-monitoring of diet and activity and feedback was shown to lead to greater weight loss in a randomized controlled trial than self-monitoring of diet by website or paper-based tools. The MMM application allows the users to set a weight loss goal and self-monitor daily calorie intake toward achieving that goal. Users select the food and drink consumed from a database and log items in an electronic food diary.

The MMM application has many usability features, such as the ability to take photographs of food to serve as a memory aid, and store favorite meal combinations and recently used items. The app has an associated Web interface to upload the recorded data. A unique feature of MMM is the large UK-specific branded food database which was provided by Weight Loss Resources, a commercial company. The database contains 23000 food and drink records that reflect both generic and branded item. The diet measures captured on MMM have been found to correlate well with a reference measure [61] of diet.

In addition to dietary self-monitoring apps, other diet related interventions are being delivered by app [59]. These include the FoodSwitch and SaltSwitch apps, which enable users to scan the barcode of packaged foods and receive an immediate interpretive, traffic light nutrition label on screen, along with a list of alternative healthier lower-salt alternatives.

3.1.3 Image-based approaches

Due to the limitations of the self-report schema in web-based and mobile-based tools, there is a clear need for improved methodology of dietary assessment. People tend to think about some more objective ways to record the daily meals and analyze the components. The researchers found using images is a good way since taking photos for food is much simpler and more accurate than self-reporting or fill up tables from the website. Various methods of image-based dietary assessment have been developed, pilot tested, or validated. Image-based food analysis method refers to any method that uses images/video of eating episodes to enhance self-report of traditional methods, or uses images/video as the primary record of dietary intake.

The images of foods can be captured using any device, but two distinct approaches for capturing the image have been explored: active or passive.

- **Active methods** typically require individuals to capture images of foods with hand-held devices, such as digital cameras or smartphones. Generally images are captured before and after eating episodes and a reference marker is placed near the foods to assist image analysis techniques. Often the images of foods are supported by supplementary text or voice recordings describing the foods, or require user input to confirm details extracted form the image, such as food type or portion size. The active approach helps ensure the images obtained are relatively consistent for image analysis, but relies on users to remember to use the camera at every eating episode.

- **The passive approach** uses wearable cameras to automatically capture point-of-view images of daily events, including eating episodes, with virtually no user input. Thus, passive image capture does not rely on users to capture images of foods, however, the image captured are not directed specifically at foods, nor do they contain a reference marker to assist analysis. A novel aspect of passive image capture, in comparison to active methods, is the ability to aid memory recall during retrospective assessment without the need for the user to manually record dietary intake during the assessment period.

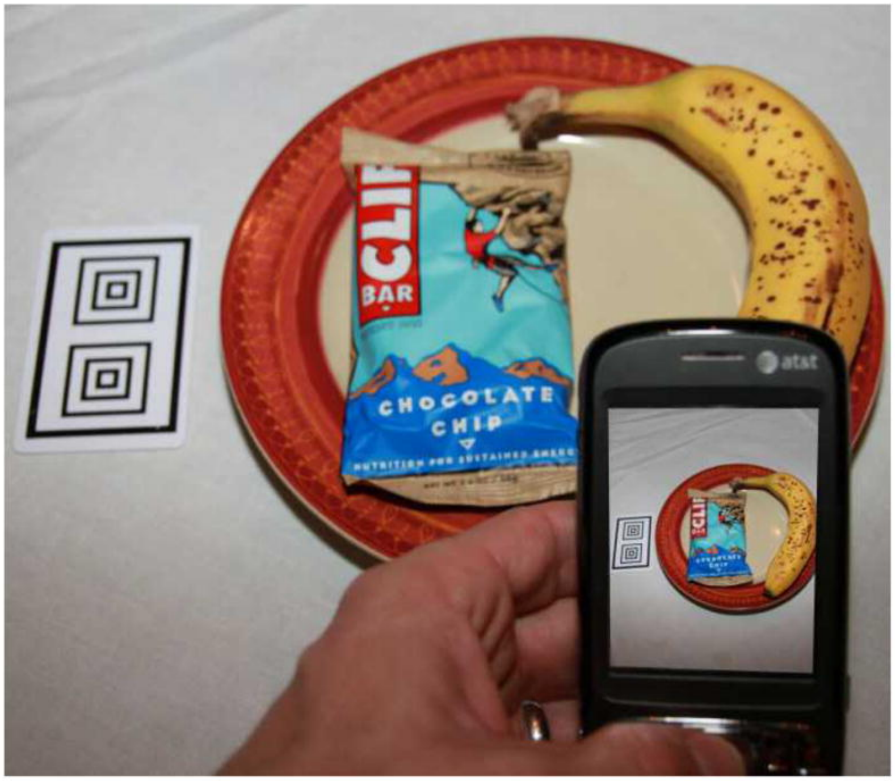


Figure 3.2 When using RFPM, participants use a smartphone to capture images of their food selection, plate waste and a reference card. These images are then immediately sent to a server for analysis

The first attempt to validate an image-based food record used camera-enabled PDAs in the Wellnavi method [62]. It required users to capture images of food before and right after eating episodes, at an angle of 45 degree. Foods were placed on a table, and the PDAs stylus was placed beside foods as a visual reference for portion size estimation. After the images were captured, users were required to describe the foods and provide ingredients with written text on the screen, especially when the foods were considered difficult to judge using images alone. The images and description of the foods were transmitted wirelessly to a server for manual image analysis by registered dietitian nutritionists. To aid analysis, a brief questionnaire was used to obtain additional information on dietary behaviors such as added sugar added to beverages, and typical condiment use.

The Remote Food Photography Method (RFPM) [63], and the mobile telephone food record (mpFR) [64] are more sophisticated methods that incorporate automated image analysis techniques into comprehensive dietary assessment systems.

Like shown in **Figure 3.2** (Source: [63]), users are required to capture images of foods on a table at 45 degree, before and after eating episodes. A reference card with a printed pattern is placed next to foods to correct for color and assist in the estimation of the food's area. The captured images are transmitted wirelessly to a server in near real time for analysis using a custom program. Features from the images are extracted for each food identified for food classification, and the program compares the foods with a searchable image archive of foods and portion sizes matched to the Food and Nutrient Databse for Dietary Studies. The food area is converted to grams based on the association between food area and weight for each respective food. To remind participants to record dietary intake, the RFPM incorporates the use of ecologic momentary assessment that is, automated prompts at meal times that require a user response.

The image-based tools make the food analysis progress more reliable and accurate. However, the issue now comes to the accuracy of recognizing the types of foods.

3.2 Motivation

3.2.1 Why we need an automatic system

The explosion of data enhances the rapid development of Internet of Things (IoT), social network and mobile networks. Now we have huge amount of data everyday, and we do not want to miss the chance of using them to improve our daily life. People keep uploading, sharing and recording food images, recipes and food diaries, leading to a large-scale of food related data. With this data, it will be helpful to build an automatic system that can learn the implication from the images, take the advantage of it. A food recognition and analysis system enables the user to record their daily meals and assess dietary habits, which can improve their health.

3.2.2 Food is Hard to Recognize

The image-based tools make the analysis process of foods more reliable and accurate. But there is still one issue needed to be overcome, that is how to recognize the types of food more correctly.

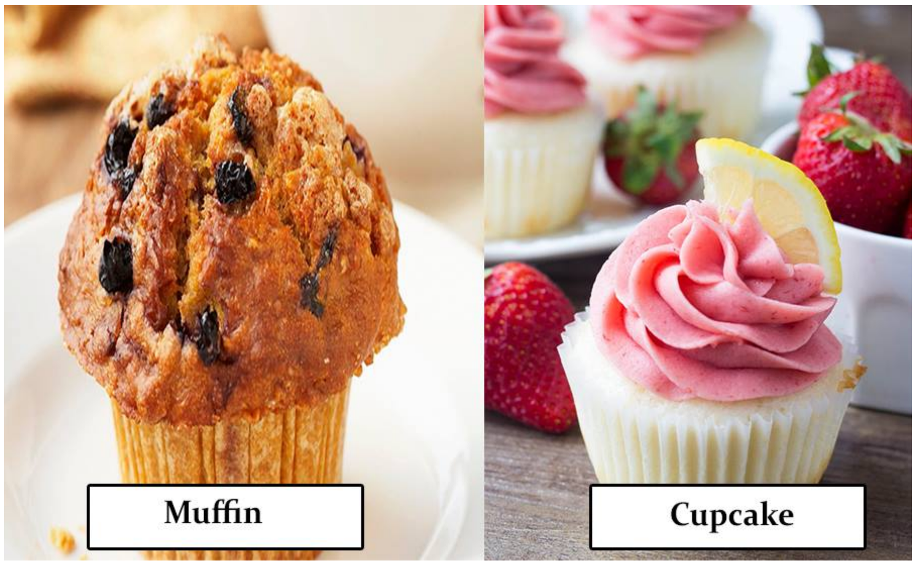


Figure 3.3 Example of low inter-class variance. Muffin and Cupcake looks similar from their appearance, but they are completely different foods.

From the statistical data we can see that even though image recognition task using modern techniques has been developed for a while, the majority of the methods in foods recognition still relied on self-reported dietary intakes. One reason for that is because comparing to other items, foods are typically deformable objects, which makes the process of defining their structure difficult. Besides, what makes the recognition more challenging is that some foods have a high intra-class (similar foods look very different) and low inter-class (different foods look very similar) variance, which may mislead the detectors to a wrong learning direction. Here is the examples of low inter-class variance (**Figure 3.3** (Source: https://www.cupcakeproject.com/cupcake-vs-muffin-update/).

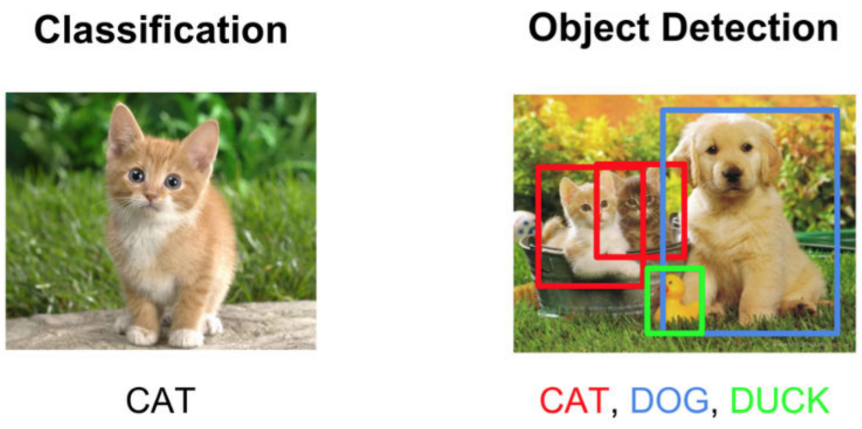


Figure 3.4 Difference between classification and object detection

3.2.3 Classification vs Detection

People may sometimes get confused when it comes to image classification and object detection scenarios. In general, if we want to classify an image into a certain category, image classification technique is what you need. In other words, we need to map the image to one category or several of them. However, on the other hand, if we aim to identify where the objects in an image, and, for example, count the number of instances for an object, we will need to use object detection. That is to say, we need to map the objects from different categories to the specific image. **Figure 3.4** (Source: https://blog.clarifai.com/classification-vs-detection-vs-segmentation-models-the-differences-between-them-and-how-each-impact-your-results) illustrates the difference between classification and object detection.

Obviously, an image can contain different categories of foods, and detection techniques enable us to recognize what kinds of foods we have in one meal. And then we can use the result to calculate the calories and analyze the foods.

3.3 Region Based Detection Approaches

Region based object detection approach is one of the most commonly used methods in detection field. In general, it proposes different region of proposals from the input image, and classify them into different categories. Compared with other methods, region based detection is more convenient to combine with deep neural network models.

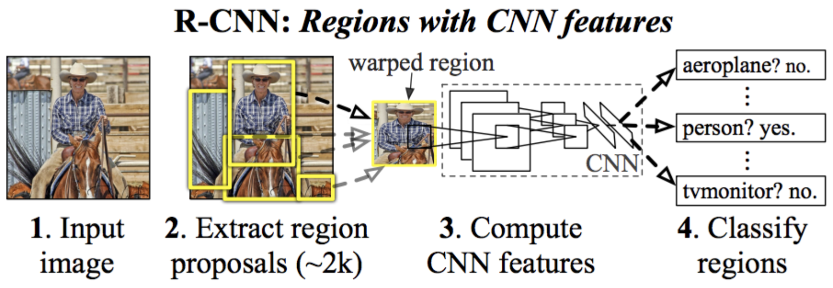


Figure 3.5 R-CNN object detection system overview

3.3.1 R-CNN

Region-CNN [65] is one of the state-of-the-art CNN-based deep learning models for object detection. Many other effective algorithms and systems such as Fast R-CNN, Faster R-CNN, YOLO and SSD are based on this approach.

Conventionally, for each image to be detected from, the traditional region-based object objection methods use a sliding window to search every position within the image. It is an easy and effective solution for detecting the objects with the similar size. However, different objects or even same kind of objects can have different aspect ratios and sizes depending on the object size and distance from the camera. And different image sizes also affect the effective window size. This process will be extremely slow if we use deep learning CNN for image classification at each location.

In order to speed up the process of detecting objects with different sizes from an image, R-CNN was invented. The goal of R-CNN is to take in an image, and correctly identify where the main objects (via a bounding box) in the image. So the input is an image, and the output is several bounding boxes and the labels for each object in the image. However, the commonly seen sliding window method is not used in R-CNN to propose different regions. Instead, it uses selective search [66] to propose a bunch of boxes from the image and see if any of them actually represent some objects. The bounding boxes is called region proposals in this thesis.

We first give a general working flow for R-CNN, as shown in **Figure 3.5** (Source: [65]). And then explore it step by step.

1. Takes an input image.

2. Extracts around 2000 bottom-up region proposals using selective search.
3. Computes features for each proposal using a large convolutional neural network.
4. Classifies each region using class-specific linear SVMs, and applies bounding-box regression to calculate the accurate position of the objects.

Selective Search

There are many papers offering methods for generating category-independent region proposals. R-CNN chooses selective search because they want to controlled comparison with prior detection work (e.g., [66]).

Selective search is one of the most efficient ways to generate possible object locations for use in object recognition. It takes the advantages of both segmentation and exhaustive search, aiming to capture all possible object locations guided by the image structure. The general procedure of selective search is that,

1. Generate initial sub-segmentation, which is used as the first generation of candidate regions.
2. Use greedy algorithm to recursively combine similar regions into larger ones, based on their colors, textures, size and shape similarity.
3. Use the generated regions to produce the final candidate region proposals.

It starts by over-segmenting the image based on intensity of the pixels using a graph-based segmentation method in computer vision. the different segments are the initial regions. And then it greedily group regions together by selecting the pair with highest similarity. The similarity measures are based on color similarity, texture similarity, size similarity and shape similarity. Loop the process until the whole image become a single region, and now it has generates a hierarchy of bounding boxes. After the selective search, about 2000 region proposals are generated from each image.

Feature Extraction and Object Detection

Once the region proposals are generated, R-CNN is ready to extract the feature vectors from the proposals and classify them. It uses AlexNet [67] to extract the feature vectors, and classify them into different objects.

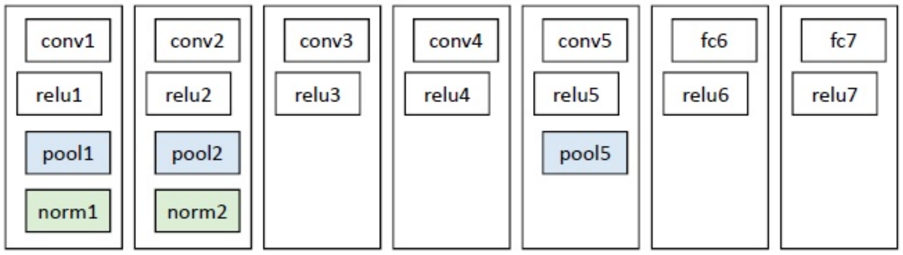


Figure 3.6 Struture of AlexNet

But there is a problem: the size of the region proposals from the first step are different, and CNN needs the input vector to have the same dimension. In order to compute features for a region proposal, the data in that region must be converted into a form that is compatible with the CNN (its architecture requires inputs of a fixed 227×227 pixel size). Regardless of the size or aspect ratio of the candidate region, all the pixels in the tight bounding box will be warped to the required size.

Then the CNN acts as a feature extractor. The structure of AlexNet is shown in **Figure 3.6** (Source: [67]), containing 5 convolutional layers and 2 fully connected layers. In the fully connected layers, the number of neurons is 4096, so R-CNN produces a 4096-dimensional feature vector as the output. Features are computed by forward propagating a mean-subtracted 227×227 RGB image through five convolutional layers and two fully connected layers. On the final layer of the CNN, R-CNN adds several SVMs to classify the region proposals to different classes. The output dense layer consists of the features extracted from the image and the extracted features are fed into an SVM to classify the presence of the object within that candidate region proposal.

Now, having found the object in the box, the final step of R-CNN runs a simple linear regression on the region proposal to generate tighter bounding box coordinates to get the final result.

How to train a R-CNN

Since R-CNN is a four steps object detection system, it can not be trained like other unified neural network using end-to-end training method.

Besides, the datasets with bounding box labelled are rare and usually people can not have enough training samples for the detector, so it is unable to train the R-CNN with random initialization. Instead, the authors use the pre-train model of AlexNet, and fine-tune with the region proposals generated using selective search, using PASCAL VOC dataset [68].

Supervised pre-training

The first step of training R-CNN is to pre-train the CNN on a large auxiliary dataset (ILSVRC2012 classification) using image-level annotations only (bounding-box labels are not available).

Domain-specific fine-tuning

After the pre-training step, we need to adapt the CNN to the detection task and the new domain (warped proposal windows). The original AlexNet has an output layer of 1000 classes, but the categories in PASCAL VOC is much less than that, about 20 classes. So we replace the CNN's ImageNet-specific 1000-way classification layer with a randomly initialized $(N + 1)$-way classification layer, where N is the number of object classes, which is 20 in PASCAL VOC, plus 1 for the background.

All region proposals with Intersection over Union (IoU) [2] larger than 0.5 are recognized as positive regions, the the rest are negatives. The R-CNN start SGD at a learning rate of 0.001, which allows fine-tuning to make progress while not clobbering the initialization. For each iteration, R-CNN uniformly sample 32 positive regions and 96 background regions to construct a mini-batch of size 128. That's because the number of background proposals is much larger than the number of foreground proposals. So if we randomly choose the training proposals, among 99% of the proposals will be background, which is bad for the training.

Once features are extracted and training labels are applied, the authors optimize one linear SVM per class.

In conclusion, R-CNN uses selective search to generate the region proposals as bounding box, and train the AlexNet to extract the feature vectors, and classify them using SVM. Finally, it applies bounding-box regression to locate a tighter bounding boxes for the object.

[2]https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Limitations of R-CNN

Although R-CNN surpasses the best performance of state-of-the-art object detection systems at that time, there are still limitations for this approach.

1. **Ad hoc training objectives** The R-CNN method has to train three different models seperately - the CNN to generate image features (log loss), the classifier that predicts the class (hinge loss), and the regression model to tighten the bounding boxes (squared loss). This makes the pipeline extremely hard to train.

2. **Training time is slow** The training is extremely slow (about 84 hours) and takes a lot of disk space, since for each image, R-CNN needs to generate 2000 region proposal and the CNN need to classify 2000 feature vectors.

3. **Prediction time is slow** The prediction (test) time is slow. The experimental result shows that the detection (test) time for one image is 47 second, which is not acceptable for a real time application.

4. **Fixed algorithm** Selective search is a fixed algorithm. Therefore, no learning is happening at region proposal stage. This could lead to the generation of bad candidate region proposals.

3.3.2 Fast R-CNN

In order to solve some of the problems in R-CNN model, Fast R-CNN [69] has been proposed by Ross Girshick. The whole approach is similar to the R-CNN, except that instead of feeding the region proposals to the CNN, Fast R-CNN feed the input image to the CNN to generate a convolutional feature map. As illustrated in **Figure 3.7** (Source: [69]), an input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers. The network has two output vectors per RoI: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.

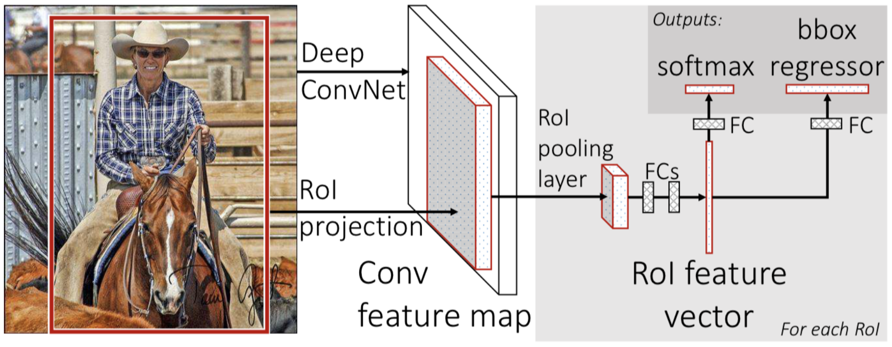


Figure 3.7 Fast R-CNN object detection system overview

RoI pooling layer

For the forward pass of the CNN, Girshick realized that for each image, a lot of proposed regions for the image invariably overlapped causing us to run the same CNN computation again and again (2000 times). And why not run the CNN just once per image and then find a way to share that computation accross the 2000 region proposals?

This is done by the technique of RoI pooling (Region of Interest Pooling). At its core, RoI pooling shares the forward pass of a CNN for an image across its subregions. As shown in **Figure 3.8**, the CNN features for each region are obtained by selecting a corresponding region form the CNN's feature map. Then the features in each region are pooled. So all it takes is one pass of the original image as opposed to 2000 in R-CNN.

The RoI pooling layer uses max pooling technique to convert the features inside each RoI into a a fixed spatial extent of $H \times W$ (e.g. 7×7) feature map (H and W are layer hyper-parameters). In Fast R-CNN, an region of interest is a rectangular window into a convolutional feature map. Each RoI is defined by a four-tuple (r, c, h, w) that specifies it top-left corner (r, c) and its height and width (h, w).

RoI max pooling works by dividing the $h \times w$ RoI window into an $H \times W$ grid of sub-windows of approximate size $h/H \times w/W$ and then max-pooling the values in each sub-window into the corresponding output grid cell. Pooling is applied independently to each feature map channel, as in standard max pooling.

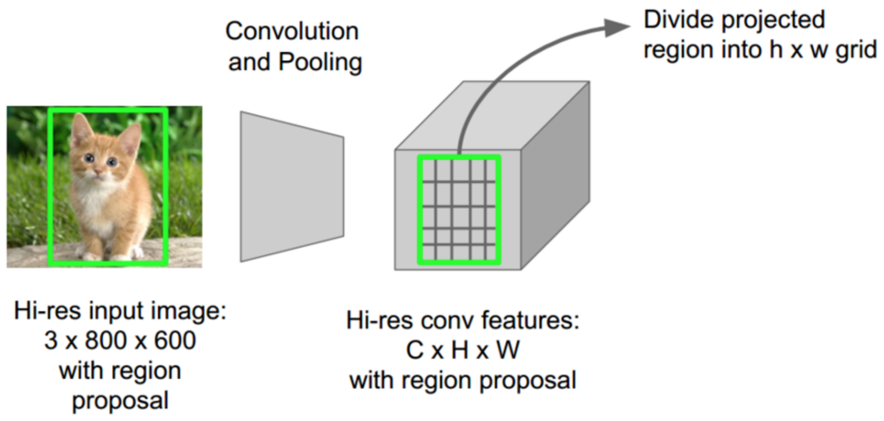

Convolution and Pooling
Divide projected region into h x w grid
Hi-res input image: 3 x 800 x 600 with region proposal
Hi-res conv features: C x H x W with region proposal

Figure 3.8 In RoI pooling, a full forward pass of the image is created and the conv features for each region of interest are extracted from the resulting forward pass.

Unified as a single network

The second improvement for Fast R-CNN comparing with R-CNN is that it can jointly train the CNN, classifier and bounding box regressor in s single model. Where earlier R-CNN had different models to extract image features (CNN), classify (SVM), and tighten bounding boxes (regressor), Fast R-CNN instead used a single network to compute all there.

Fast R-CNN implement this by replacing the SVM classifier with a softmax layer on top of the CNN to output a classification. It also added a linear regression layer parallel to the softmax layer to output bounding box coordinates. In this way, all the outputs needed came from one single network.

Bottlenecks of Fast R-CNN

From **Figure 3.9** we can infer that Fast R-CNN is significantly faster in training and testing sessions over R-CNN. However, during the testing time, including region proposals slows down the algorithm significantly (2 second) when compared to not using region proposals. That is because even though it feeds the CNN with the whole image each time, Fast R-CNN still needs to use selective search to generate the region proposals for each image. During the testing time, selective search algorithm takes about 2 second per image. Therefore, region proposals become bottlenecks in Fast R-CNN algorithm affecting its performance.

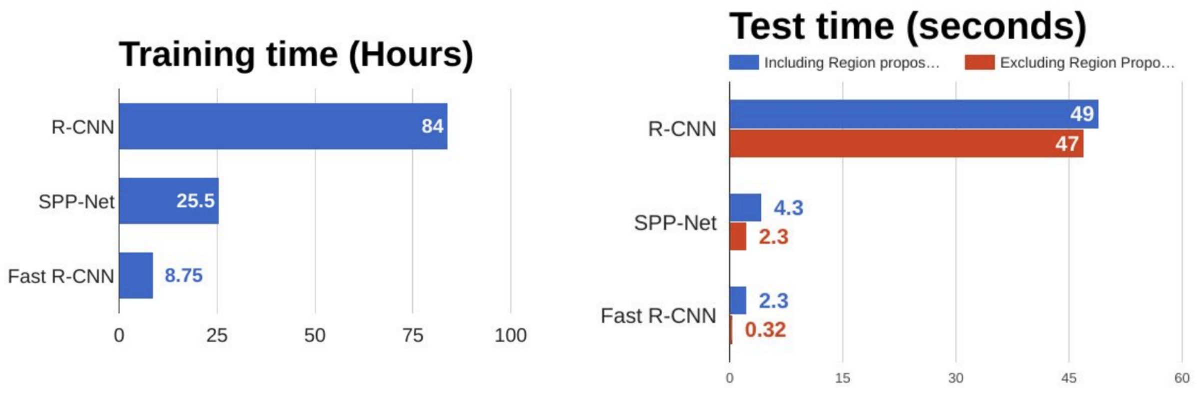


Figure 3.9 Training and testing time for object detection algorithms

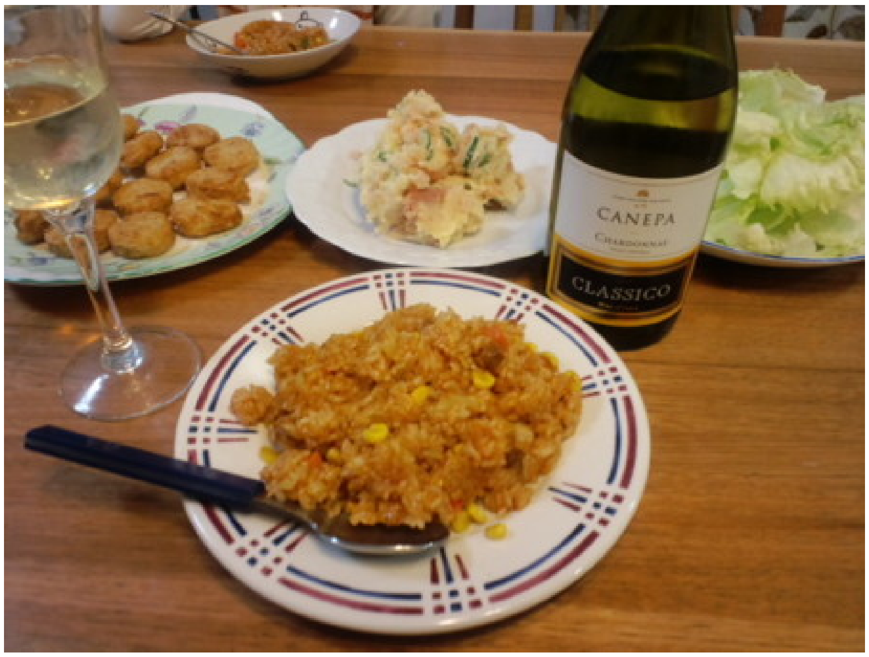


Figure 4.1 Example of food image with non-food background

Chapter 4

System Design

In this section, we first give an overview of our proposed food recognition and nutrition analysis system based on deep convolutional neural network. Then we discuss the detailed implementation of each parts of the system.

4.1 System overview

In order to improve the accuracy of detecting the foods, we propose a novel deep learning based object detection model to the database of food to analyze the component of each image. Some of the deep learning based classification models have been applied to the food recognition tasks, and have had great success. However, most of them are extracting the visual features of different foods from the entire image, which would inevitably include the background information as well. Obviously, the background features will influence the accuracy of detection, since the background may distract the detectors to classify a non-food object with the same shape or color with a specific food to that kind of food object. Here is an example of the food with a background **Figure 4.1** (Source: UEC-FOOD100 Dataset).

Region based deep learning based models have had a great success in object detection task, using different kinds of region proposal methods to generate the regions of interests (RoI). We assume that these RoIs will dissociate the food from the image or background, making it more convenient to extract the features of food and easier to recognize the objects. However, the state-of-the-art models from competitions like ImageNet, Large Scale Visual Recognition Challenge (ILSVRC) and MS-COCO have not yet been widely applied to databases with foods.

In this thesis, we are interested in using region based object detection methods to extract the regions of interests from the database of foods and recognize them in each image. After that, we apply modern technology-based dietary assessment tools to analyze the nutrition of each meals such as carbohydrates, calories, fat and protein.

As shown in **Figure 4.2**, our model consists of three steps.

- We apply the Region Proposal Network from the Faster R-CNN detection model to generate thousands of region of proposals. These region of proposals help to separate the food items from the background, and make it more convenient to extract features from it.

- The second step is to apply state-of-the-art Convolutional Neural Network (CNN) to extract the feature maps for each region of proposals, and classify them into different food items. Meanwhile, we apply an regression module from Faster R-CNN, to locate the food coordinates in the image.

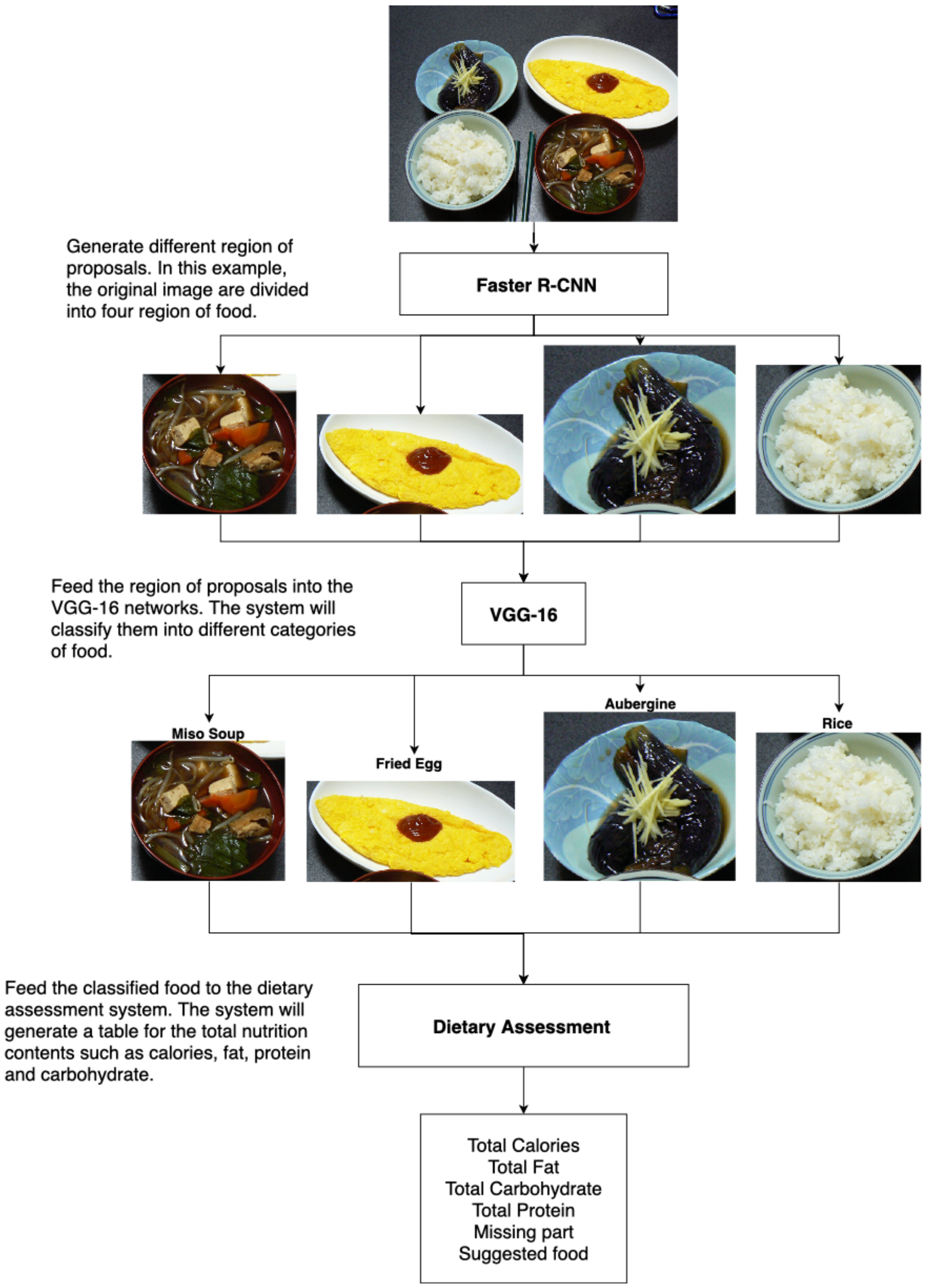

Generate different region of proposals. In this example, the original image are divided into four region of food.
Faster R-CNN
Feed the region of proposals into the VGG-16 networks. The system will classify them into different categories of food.
VGG-16
Miso Soup
Fried Egg
Aubergine
Rice
Feed the classified food to the dietary assessment system. The system will generate a table for the total nutrition contents such as calories, fat, protein and carbohydrate.
Dietary Assessment
Total Calories
Total Fat
Total Carbohydrate
Total Protein
Missing part
Suggested food

Figure 4.2 The automatic three steps system of food recognition and nutrition analysis system

- The last step is to use modern technology-based dietary assessment tools to analyze the nutrition of each food item, and come up with a health report for the dish.

4.2 Faster R-CNN

Both R-CNN and Fast R-CNN uses selective search algorithm to find out the region proposals, which is a slow and time-consuming process affecting the performance of the detection network. To overcome this issue, Shaoqing et al. found a way to make the region proposal step almost cost free through an architecture they invented, called Faster R-CNN [70]. The insight of Faster R-CNN is that region proposals depend on features of the image that were already calculated with the forward pass of the CNN. So why not reuse those same CNN results for region proposals instead of runing a separate selective search algorithm?

Faster R-CNN combines the feature extraction, region proposal, bounding box regression and classification models to a unified network, making it perform (especially during testing time) more efficiently. In general, Faster R-CNN works as **Figure 4.3** (Source: [70]) illustrated.

1. **Conv Layer** As an object detection algorithm, Faster R-CNN uses a basic convolutional module with convolution layer, relu activation function pooling layer to extract the feature map from the image. The feature map is utilized by the region proposal network (RPN) and classifier later.

2. **Region Proposal Networks** Region proposal network (RPN) is used to generate region proposals. In this layer, anchors are classified as foreground region or background region, together with a bounding box regression to fix the anchors location to gain more accurate region proposals.

3. **RoI Pooling Layer** RoI pooling layer are the same as in Fast R-CNN, combines the information from feature maps and region proposals, max pooling and transfer them to the fully connected layer to classify.

4. **Classification** The classification module uses the proposal feature maps to classify the category of each proposal, and apply bounding box regression again to gain the final accurate location for each object.

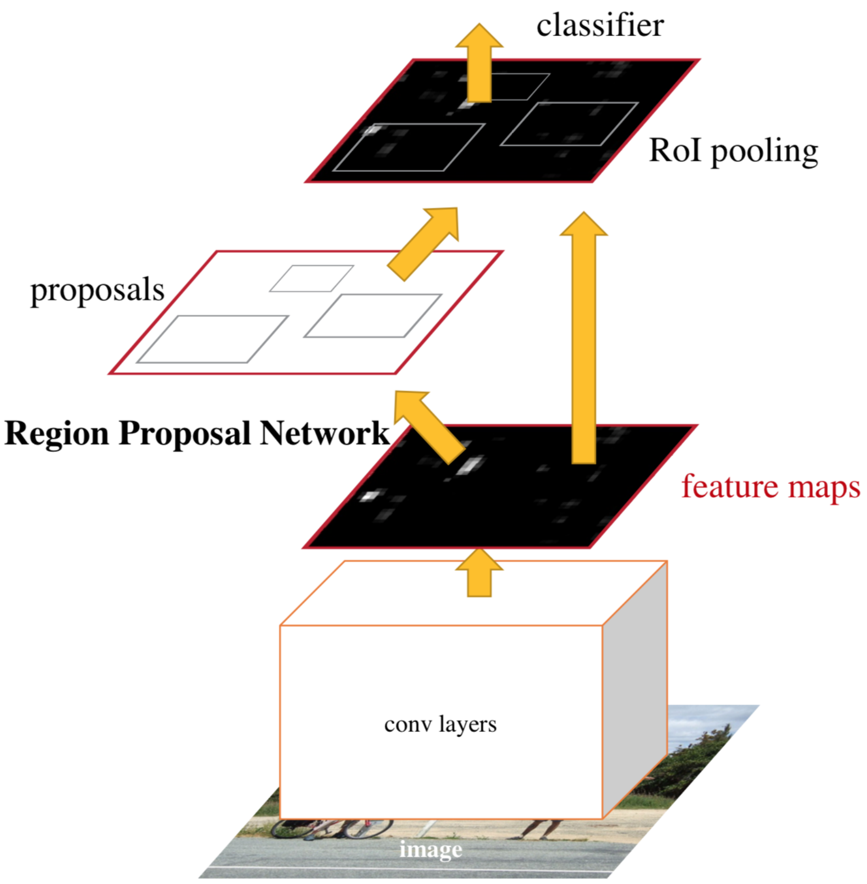


Figure 4.3 Faster R-CNN is a single, unified network for object detection. The RPN module serves as the region proposal network of this unified network.

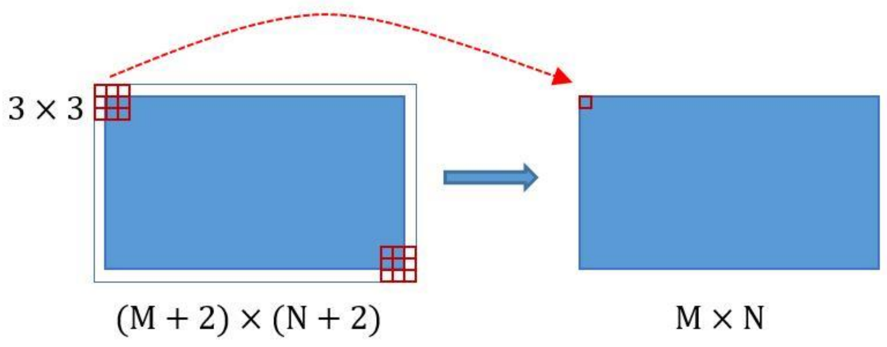


Figure 4.4 How convolutional layer works

4.2.1 Conv Layer

Conv layers contain 3 kinds of layers: convolutional layer, pooling layer and relu layer. For example, there are 13 convolutional layers, 13 relu layers and 4 pooling layers in VGG16 CNN.

- For all convolutional layers: kernel size is 3, padding is 1, stride is 1.
- For all pooling layers: kernel size is 2, padding is 0 and string is 2.

This setting is important since it applies zero padding to all convolutional layers. If the original image is $M \times N$, after the zero padding, it will be added to size $(M + 2) \times (N + 2)$. And then the conv kernel will convert it to $M \times N$ again. Only with this setting, the convolutional layers will not change the size of the input and output matrices. It is illustrated in **Figure 4.4**

Similarly, both the kernel size and the stride in pooling layers are 2, making the matrix from $M \times N$ to $M/2 \times N/2$ every time it passes a pooling layer. As a result, the $M \times N$ matrix will become $M/16 \times N/16$ feature map after the conv Layer, which will match the original image.

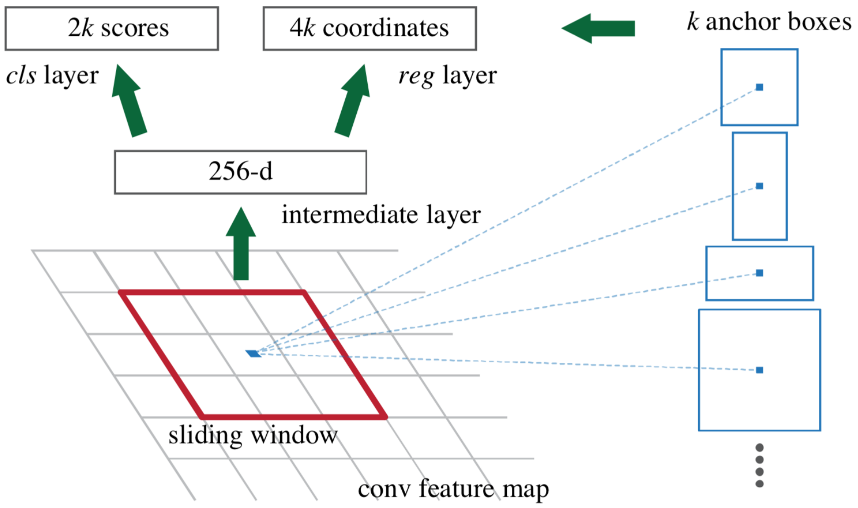


Figure 4.5 Region Proposal Network with k anchors

4.2.2 Region Proposal Networks

Some of the classical method of generating region proposals are very time-consuming, like the sliding methods and selective search in R-CNN and Fast R-CNN. So in Faster R-CNN, they use region proposal network (RPN) to accelerate the speed of generating region proposals. RPN can be divided into two part, one is a classification module that classify the anchors to be either foreground or background through softmax layer. The other part is a bounding box regression module that can calculate the offset between the anchor and the ground truth region.

Anchors

At each location in the feature map, RPN simultaneously predict multiple region proposals, where the number of maximum possible proposals for each location is denoted as k. The k proposals are parameterized relative to k reference boxes, which is the anchors. An anchor is associated with a scale and aspect ratio. By default there are 3 scales and 3 aspect ratios, yielding $k = 9$ anchors at each location. For the feature map of a size $W \times H$, there are $W \times H \times k$ anchors in total.

As illustrated in **Figure 4.5** (Source: [70]), the final layer in the Conv Layer has output dimension of 256, so there are 256 dimension for every location (point) in the feature map. With k anchors, we want to classify each anchor to class foreground or background, so there are $2k$ scores for the classification. Also, there are 4 offset values for each anchor, so in total is $4k$ coordinates for the bounding boxes.

Softmax layer to classify foreground and background

After extracting the anchors from the feature map, RPN uses a softmax layer to classify the anchor to be foreground or background. After the classification, RPN chooses suitable anchors to become the initial region proposals.

Bounding box regression

Proposal Layer

Proposal layer is responsible for compositing the initial foreground anchors and bounding box regression to generate the final region of interests (RoIs). It has three inputs: (1) the classification result of foreground and background anchors, (2) corresponding bounding box regression coordinates and (3) information from the original image. The information from the original image includes the resizing scale that used to resize the original image as $P \times Q$ to a fixed size $H \times W$ in order to go through the convolutional neural network.

Here is the general steps generating the region of interests.

1. **Generate anchors** Use the regression information to calculate the accurate position for each anchor.

2. **Sort the anchors** Sort all the anchors with foreground class based on their softmax scores, and choose the top 2000 anchors and ignore others.

3. **Non-max Suppression (NMS** Apply non-max suppression to eliminate points that do not lie in important edges.

4. **Choose the final RoIs** Sort the foreground anchors again based on their scores. Choose the top 300 anchors to be the region of interests.

In conclusion, Region Proposal Network generates the anchor first, and then classifies them and calculates the regression for each anchor. At the end, it chooses the top 300 of them to become the region of interests.

4.2.3 Region of interests pooling layer

The goal of the RoI pooling layer is to take in different size of region proposals and convert them into fixed size feature map.

Basically, RoI pooling layer is the same as in the Fast R-CNN object detection system. It has two input, original feature maps and the regions of interests. For example, the size of the proposals is $M \times N$, and the pooling size is 7×7.

- It first converts the proposals to fix the size of feature map using spatial scale computed earlier, that is $M \times N$ to $(M/16) \times (N/16)$.
- Then it maps the resized proposal to the feature map, and divides the $M/16 \times N/16$ to 7×7 blocks.
- It uses max pooling to get the largest grid among each block, and generate a 7×7 feature vector.

After the processing above, regions of interests with different sizes are all converted to fixed size with 7×7.

4.2.4 How to train a Faster R-CNN

The training of Faster R-CNN is based on the pre-trained network, in our thesis, VGG. It is called four step alternating training.

1. The first step is to train the region proposal network using the pre-trained model.
2. Then we use the RPN trained in the first step, we trained the separate detection network by feeding the proposals generated by the RPN. The detection network is also initialized by the ImageNet-pre-trained model.
3. After that, we use the detection network to initialize RPN training, but fixing the shared convolutional layers and only fine-tune the layers unique to RPN.
4. Finally, we keep the convolutional layers fixed, and fine-tune the unique layers of Fast R-CNN.

Based on this training process, RPN and the detection network can share the same convolutional layers and form a unified network.

Training RPN

Region proposal network is trained end-to-end by back-propaggation and stochastic gradient descent (SGD). Each mini-batch arises from a single image that contains many positive and negative example anchors. In this thesis, the sampled positive and negative anchors in an image to compute the loss function is 1:1.

The loss function for the whole RPN network is

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

where i is the index of anchors, p_i is the foreground softmax probability, p_i^* is the predicted probability of ground truth. t is the predicted bounding-box, t^* is the grounding truth box for the foreground anchors.

From the above equation, we can see the total loss function is combined by classification loss and regression loss, and we use λ to find the balance between these two losses.

Training detection network

After getting the regions of interests from the RPN, we need to use them to train the detection network. The training process is the same as training a detection network in Fast R-CNN.

In conclusion, Faster R-CNN replaces the region proposals part from Fast R-CNN, which uses selective search to generate the region proposals. Instead, Faster R-CNN uses a region proposal network to create the regions of interests, and speed up the process of detection.

4.3 Detection module

For the detection module, we apply Faster R-CNN model to detect the food items from the images.

According to the analysis above if we only extract the features from the food regions, the feature maps will be more clear and will decrease the noise caused by the background images. In order to separate the foods from the image and background, our model uses Faster R-CNN to detect the regions of foods. Faster R-CNN is a deep learning model for object detection that takes an image and outputs bounding boxes around objects of interest with class labels. It contains a Region Proposal Network that acts as an attention network that identifies the regions the classifier should consider.

However, existing Faster R-CNN is trained with VOC2007 database, which only contains 20 commonly seen objects, and none of them are food-related. So we should at first choose a food-related database, in this thesis we use FOOD100 [19], to fine-tune the pre-trained Faster R-CNN. Then we use the detection model to extract the regions of proposals for each food in the image.

The total loss function is shown in equation 4.1.

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{4.1}$$

Here i is the index of the anchor in one batch, p_i is the predicted probability of anchor i. If anchor is foreground, p_i^* is 1, otherwise p_i^* is 0. t_i represents the four coordinates of the predicted bounding box region. t_i^* is the matching ground-truth box for the foreground anchor.

In our thesis, we utilize several different deep neural network architectures in the convolutional neural network part in Faster R-CNN to compare the performance and find out the most suitable one. The first one we try is the AlexNet [67] architecture, with 7 layers to jointly detect objects and assign classes to the objects. Then we use VGG-16 [71] network, which is a very deep convolutional network (16 layers) for object recognition developed and trained by Oxford's renowned Visual Geometry Group. And we also try the Residual neural network (ResNet) [72], which is an artificial neural network of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. In this thesis, we compare with ResNet50 architecture with 50 layers in total.

4.3.1 Implementation Detail

Our model requires the size of images to be in an unique scale. So we resize the original different size with different scale so that all the shorter side of the image end up with $s = 600$ pixels, and longer side with $l = 1000$. For some images, s may be less than 600 as we cap the longer image side at 1000 pixels and maintain the image's aspect ratio. These values were selected following the rules of [69], which make VGG16 fits in GPU memory during fine-tuning. The smaller models are not memory bound and can benefit from larger values of s.

For the anchors of the Faster R-CNN model, we set 3 different scales with size of 128^2, 256^2 and 512^2 pixels, and also 3 aspect ratios of $1:1$, $1:2$ and $2:1$. This ends up with 9 different anchors for each of the point in the feature map. For example, an image of 1000×600 pixels image, the number of anchors could reach to 20000 in total. During the training period, we ignore all cross-boundary anchors so that they do not contribute to the training loss. If the boundary-crossing anchors are not ignored, it would introduce large, difficult to correct error terms in the objective function, and the training process would not converge. This implementation leaves us about 6000 anchors for one image for training. However, during testing time, our models still apply the fully convolutional RPN to the entire image.

Since we have 6000 anchors per image to train, some of the proposals (anchors) are highly overlapping with each other. We reduce the redundancy by adopting non-maximum suppression (NMS) [73] on the proposal regions based on their classification score. NMS is an algorithm that eliminate points that do not lie in important edges. The image is scanned along the image gradient direction, and if pixels are not part of the local maxima, they are set to zero. This has the effect of suppressing all image information that is not part of local maxima. We set the IoU threshold for NMS at 0.7, and this results in around 2000 proposal regions per image. After NMS, we use the top-N ranked proposal regions for detection. In the following, we train the detection network using 2000 RPN proposals, but evaluate different numbers of proposals at test-period.

In order to share the convolutional layers between the RPN and the detection module, rather than learning two separate networks, we use 4-step Alternating Training strategy in [70] to train our unified models with feature shared.

1. First we train the RPN module by using back-propagation and stochastic gradient descent. The network is initialized by a pre-trained model for the ImageNet classification, and fine-tuned end-to-end with our dataset. For each image, a sample of 256 anchors are randomly chosen to compute the loss function of a mini-batch, where the sampled positive and negative anchors are evenly selected.

2. In the second step, We train the same detection network used in Fast R-CNN with the proposals generated by the step-1 RPN. This detection network is also initialized by the ImageNet-pre-trained model. At this point, the two networks do not share convolutional layers.

3. For the third step, we use the detector network to initialize RPN training, but we fix the shared convolutional layers and only fine-tune the layers unique to RPN. Now the networks share convolutional layers.

4. Lastly, we keep the shared convolutional layers fixed and fine-tune the unique layers of the detection network. As such, both networks share the same convolutional layers and form a unified network.

4.3.2 Data augmentation

To enrich the training samples, we use additional data augmentation techniques to generate more training data. This include flips and rotations.

Flip - The first technique is to flip images horizontally and vertically. In our implementation, we only apply it horizontally. The reason is that many categories of food with bowl or plate may get wired when we flip them vertically, such as soup and ramen.

Rotations - We apply three kinds of rotation techniques to our datasets, which are 90 degree, 180 degree and 270 degree. Most of the images are squares or rectangles. If we rotate them with other degrees, we will not able to preserve the size of it.

After detecting the image, we generate several regions of food and for each region, we give it a score as the likelihood of that kind of food.

4.4 Classification module

4.4.1 Classification

The classification module uses the proposal feature maps computed from each image to compute each proposal belongs to which category, and calculate the score for that class. At the same time, it apply bounding box regression again to gain the more accurate location for each proposal.

There are at most five foods in one image in UECFOOD100 database. So we choose at most five bounding boxes regions with the highest scores. If the number of regions detected in the image is less than five, we just use that number to classify the foods.

CNN models have been widely used in classification task. In this thesis, we use VGGNet, one of the state-of-the-art deep neural network, to extract the feature map of the proposed regions, and classify them into different kinds of food. In the classification module, the detection results from former part are fed into an VGGNet to obtain a 4096 dimensional feature vector used to classify them into more fine accurate categories.

In our structure, we use VGG-16 model, which contains 16 weight layers in the network, as well as several fully connected layer. We replace the last layer of VGG to a softmax layer with 100 units, representing the 100 categories of UEC-FOOD100 model. When we do the experiments using other datasets, we also change the number of units in the final layer to correspond to the number of categories in the datasets respectively.

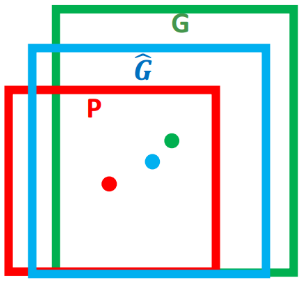


Figure 4.6 Bounding box regression

All deep learning models have been pre-trained with the natural image dataset ImageNet and fine-tuned with our training data. The images in the training set were randomly split into a training and validation set for tuning the learning and determine early stopping. Also in this thesis, additional augmentation was done to individual food images. This included rotations, flips, horizontal and vertical shifts and scale shifts.

4.4.2 Regression

As for the regression part, usually the initial region of proposals we get from the detection module can not cover the whole object. We apply the bounding box regression module from Faster R-CNN to fine tune the coordinates of the bounding box, so that it can match the ground truth bounding box better.

Usually the initial region proposal can not cover the whole object, so RPN uses a bounding box regression to fine tune the location of the bounding box, so that it can match the ground truth box entirely.

As shown in **Figure 4.6** (Source: https://www.cnblogs.com/makefile/p/bbox-regression.html), the original foreground anchors is the red box P, the green box G represents the ground truth location. The bounding box regression is to find a transformation that cast the P to G', which is closer to ground truth G. A simple way to do this is first do translation in equation 4.2 and 4.3

$$G'_x = A_w \cdot d_x(A) + A_x \tag{4.2}$$

$$G'_y = A_h \cdot d_y(A) + A_y \tag{4.3}$$

And then zoom in or out like in equation 4.4 and 4.5

$$G'_w = A_w \cdot \exp(d_w(A)) \tag{4.4}$$

$$G'_h = A_h \cdot \exp(d_h(A)) \tag{4.5}$$

In order to learn $d_x(A), d_y(A), d_w(A), d_h(A)$, we assume that all these operations are linear operation. So that we can use linear regression to fine tune the bounding box coordinates.

Linear regression is to learn weight W for the input feature vector A to match the ground truth t. That is $t = W \cdot A$. The loss is defined as equation 4.6

$$Loss = \sum_{i}^{N} (t^i - W^T \cdot A^i)^2 \tag{4.6}$$

4.5 Dietary assessment

The final step of our model is the dietary assessment module that can be used to analyze the nutrition of the meal. In a nutritional sense, all types of foods, whether they are fats, proteins, carbohydrates or sugars, are all important sources of calories, which people need to live and function. In this thesis, we mainly focus on calculating calories, fats, carbohydrates and proteins contents for each meal. Additionally, healthy food like vegetables, nuts and whole grain food are also expected in a healthy diet.

Our system will ask the user about the basic information of their bodies, such as age, gender, weight and activity level. Based on the given information, the system will automatically generate the suitable amount of calories the user need for a healthy diet. The system will also determine whether the user are having enough energy and specific kinds of food.

Table 4.1 Snippet of the reference nutrition facts table. Each row contains the amount of calories, fat, carbohydrate and protein for each food item. All food items are considered as 400 grams.

Food(400g)	Calories	Fat(g)	Carbohydrate(g)	Protein(g)
Steak	1365	63	0	187.3
Ramen	760	29	78	35
Miso Soup	81	3.3	9.8	6.5
Fried Rice	619	12.8	106.8	12.8
Sushi	536	7.7	103.3	13.4
French Fries	428	21.4	57.1	4.8
Takoyaki	1264	92.8	67.2	38.4
Pizza	690	20.4	103.9	26
Hamburger	1086	73.7	0	99.1
...				

Now the question comes to how to use the system? All you need to do is simply take out your mobile phone and take a picture of your meal before having it. The system will automatically recognize your meal, separate them from the background, and classify them in just a second. After the analysis, the system will tell you the total amount of the calories of your meal, plus the amount left for you to take after this meal. Besides, if you are missing some specific kinds of healthy food, the system will remind you of it.

The standard source for dietary assessment is the USDA National Nutrient Database (NNDB). It contains nutritional facts about 8618 basic foods. We utilize the information provided from this database.

First we build a reference table of nutrition facts tables including all the food items in our datasets. The standard source we used for building this reference table is the USDA National nutrient Database (NNDB) [1], which lists nutritional facts about 8618 basic food items. For the estimation of the amount of calories and other nutritional ingredients, our system takes one photo and the bounding boxes regarding each food item in that photo at each time. Then it automatically calculate the amount of nutritional ingredients for the food items by mapping the detected food to the reference table.

Table 4.1 is a snippet of our reference table. For each item, we assume each food item is 400 grams, which is a normal size for a single serve. For each row of the reference table, we can get the amount of calories, fat, carbohydrate and protein from it. There is in total, 100 rows in the reference table, representing the 100 categories of food items in UEC-FOOD100 dataset.

In order to provide a healthy diet, we also design an automatic diet calculator. The calculator requires the user to input their age, gender, weight, height and activity level information. And it will automatically generate a healthy diet specifying the suitable amount of nutritional content needed everyday for that user. For example, if a 24 year-old man, with weight of 60 kilogram and height of 170 centimeter, with an activity level of moderate, a suitable amount of nutritional ingredients for a healthy diet will be 2399 calories per day, 311 grams of carbohydrate, 109 grams of protein and 80 gram of fat.

[1] http://www.ars.usda.gov/ba/bhnrc/ndl

Chapter 5

Experiment and Evaluation

In this chapter, we set up experiments to evaluate our proposed models and methodologies. Our experiment framework consists of three parts: data preprocessing, model training and performance evaluation. First, we give an introduction of experiment databases and evaluation metrics. After that, we present the experiment environment and computing resource. Finally, we compare the performance of our proposed methodologies with baseline models and discuss the robustness of our methods.

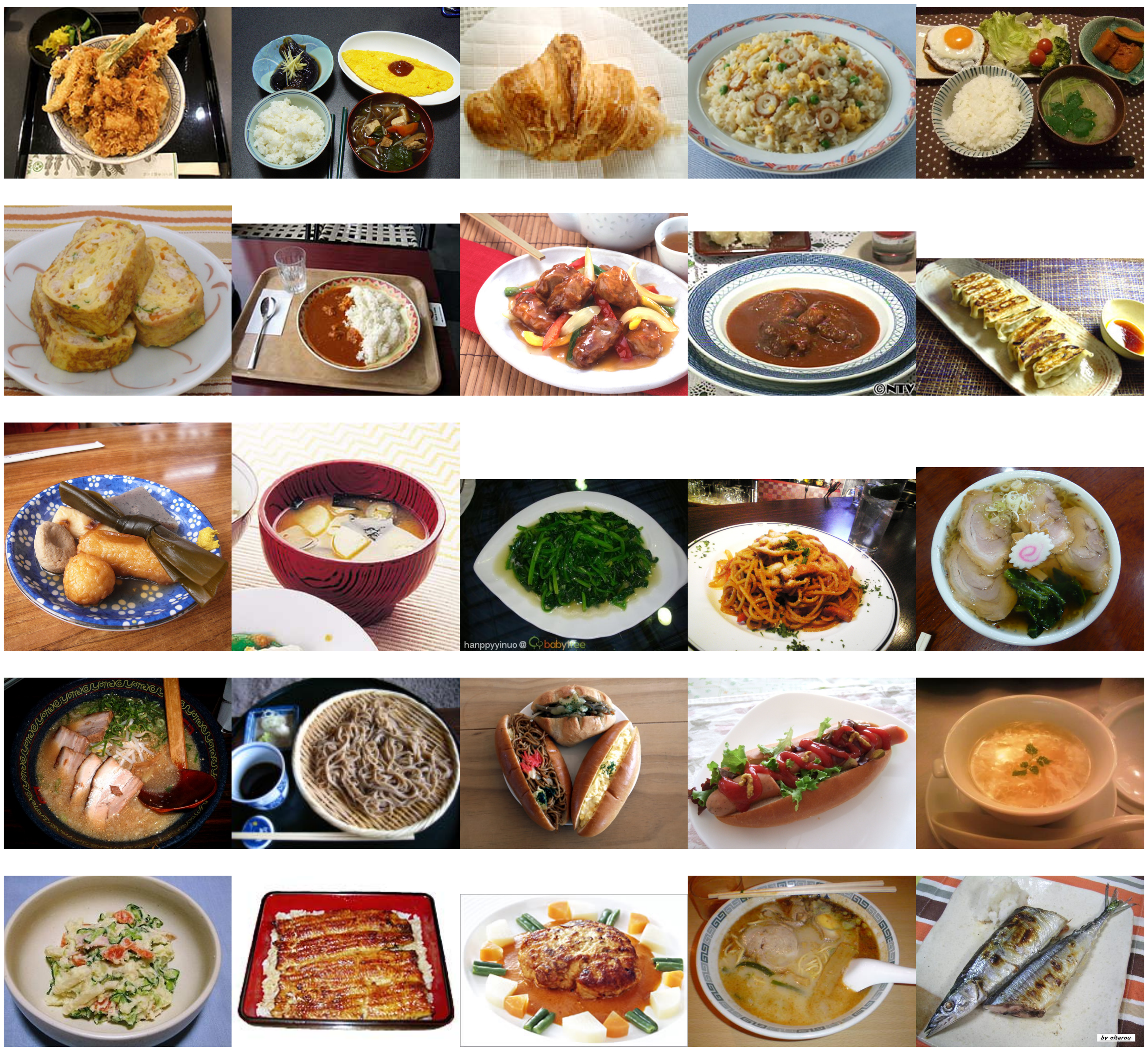


Figure 5.1 Some examples of food items in the UEC-FOOD100 Dataset. This dataset includes both single food item and multiple food items in one image.

5.1 Datasets

Two real-world datasets: UEC-FOOD100 [19], UEC-FOOD256 [20] are used to evaluate the proposed models in this thesis. More concretely, UEC-FOOD100 contains 100-class food images while UEC-FOOD256 contains 256-class food images. Some of the food categories in the UEC-FOOD100 dataset are shown in **Figure 5.1**.

Inside of the UEC-FOOD100 database, there are 12740 images in total of 100 categories of different food items. Most of them in this dataset are popular foods in Japan. The dataset has four *txt* files: *category.txt*, *category_ja_ utf8.txt*, *multiple_food.txt* and *bb_ info.txt*. *category.txt* matches the category with the index from 1 to 100. *category_ja_ utf8.txt* translates the categories' name to Japanese. *multiple_food.txt* contains the list representing food photos including two or more food items. *bb_ info.txt* includes the information of the bounding box coordinates for each food items in one image. The format of bounding box file is $image\_index, x1, y1, x2, y2$, which$image_index$ is the unique index of the image. $x1$ is smaller coordinate of the bounding box in x-axis, $y1$ is the smaller coordinate of the bounding box in y-axis. $x2$ and $y2$ are the larger coordinate for the bounding box.

The structure of UEC-FOOD256 dataset is the same as that of UEC-FOOD100, except that the number of images is 31395, and the number of categories is 256.

Table 5.1 UEC-FOOD100 Dataset statistics

Total number of categories	100
Total number of images	12740
Number of images with single food item	11566
Number of images with multiple food items	1174
Category with largest amount of images	Miso Soup (729)
Category with least amount of images	Chicken Rice (101)

Each of the categories in both datasets include more than 100 images, with the bounding box information indicating food location within each food photo. In order to show the overview of the UEC-FOOD100 dataset, we summarized the statistics in **Table 5.1**.

The bounding boxes information for each food items is crucial in food detection task. We can not train our model without the bounding boxes indicating the location of the food. Both of the UEC-FOOD100 and UEC-FOOD256 datasets contain the bounding boxes information for each food item in the photos. However, most of the categories of food in these two datasets are Japanese food, with similar shape and ingredients. We want to test the robustness of our model for a larger range of food categories. So we find another commonly used dataset named FOOD101 [18], which is a challenging data set of 101 food categories, with 101000 images. For each class, 1000 images are included including some uncleaned images provided on purpose. This dataset contains mostly western food with exactly one item in each photo.

The problem with FOOD101 dataset is that it does not provide the bounding box information for the food item. So in order to test our model's robustness for a larger range of categories of food, we generate a new dataset by lining out the bounding boxes manually for a portion of images from FOOD101 dataset. To be more specific, we choose 20 classes from FOOD101, and draw out the bounding boxes for 300 photos within each class. We call our dataset FOOD20-with-bbx. Here are some examples taken from FOOD20-with-bbx shown in **Figure 5.2**

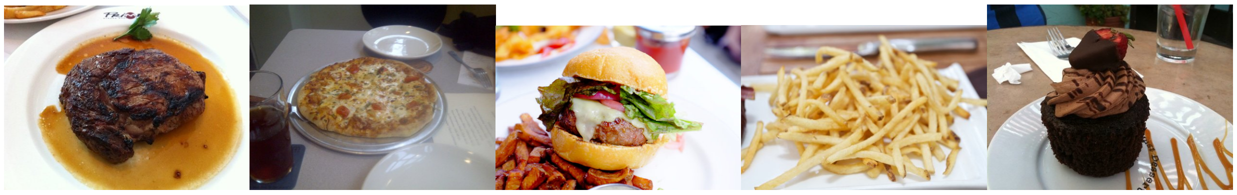


Figure 5.2 Some examples of food items in the FOOD20-with-bbx Dataset. This dataset includes only single food item.

5.2 Evaluation Metrics

In this thesis, we primarily evaluate detection mean Average Precision (mAP) to compare the performance.

In object detection model, evaluation is not as simple as other models, because there are two distinct tasks to measure at the same time.

- **Classification** - Determining whether an object exists in the image.

- **Regression** - Determining the location of the object.

The other reason is that there will be many classes and their distribution is non-uniform. In our cases, the number of images in the first category in UEC-FOOD100 and UEC-FOOD256, which is rice, is four to five times more than in the other categories. A simple accuracy-based metric will introduce biases. It is also important to assess the risk of misclassifications. Thus, there is the need to associate a "confidence score" with each bounding box detected and to assess the model at various level of confidence.

To overcome these difficulties, the mean Average Precision (mAP) is introduced. The mAP combines the classification evaluation and regression evaluation into one metric.

5.2.1 Classification Evaluation

Average precision (AP) is a popular metric in measuring the accuracy of object detectors. It computes the average precision value for recall value over 0 to 1.

Precision (5.1) measures the false positive rate or the ratio of true object detections to the total number of objects that the classifier predicted [1]. If the precision score is close to 1.0, it means that there is a high likelihood that whatever the classifier predicts as a positive detection is in fact a correct prediction. Recall (5.2), on the other hand, measures the false negative rate or the ratio of true object detections to the total number of objects in the data set. If the recall close to 1.0, then almost all objects that are in your data set will be positively detected by the model.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{5.1}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{5.2}$$

It is important to know that there is an inverse relationship between precision and recall and that these metrics are dependent on the confidence score. With a higher confidence score, the number of detections classified as positive class is less.

[1] https://en.wikipedia.org/wiki/Precision_and_recall

To calculate the AP, for each class, the precision-recall curve is computed from the model's detection output, by varying the model score threshold that determines what is counted as a model-predicted positive detection of the class. AP (5.3) is basically the area under the precision-recall curve. For simplicity, we divide the recall values equally into set of 11, and calculate the maximum precision measured at each set (5.4).

$$AP = \int_0^1 Precision(Recall)d(Recall) \tag{5.3}$$

$$AP = \frac{1}{11} \sum_{Recall_i} Precision(Recall_i) \tag{5.4}$$

Precision and recall are always between 0 and 1. Therefore AP falls within 0 and 1.

5.2.2 Regression Evaluation

The section above describe how to evaluate one of the two required tasks in detection, the classification. Now we move on to the evaluation of the object's location precision, which is the regression task. We must consider the amount of overlap between the part of the image segmented as true by the model vs. that part of the image where the object is actually located.

In order to evaluate the model on localization, we must determine how well the model predicted the location of the object. In this thesis, we use the Intersection over Union (IoU) metric to measure the accuracy of the localization. Computing IoU can be determined via 5.5. We compute the $areaofoverlap$ between the predicted bounding box and the ground truth bounding box. And the $areaofunion$ is simply the area encompassed by both the above bounding box.

$$IoU = \frac{AreaofOverlap}{AreaofUnion} \tag{5.5}$$

Model object detections are determined to be positive or negative depending upon the IoU threshold. The IoU thresholds may vary with different task, and in this thesis we use IoU threshold of 0.5.

Table 5.2 Experimental environment statistics

Deep learning API	Keras
Neural Network backend	Tensorflow
Computing Platform	Compute Canada
Computing Resource	NVIDIA P100-PCIE GPU

Now we have defined Average Precision (AP) and the IoU metric, the mean Average Precision (mAP) score is calculated by taking the mean AP over all object classes with the IoU threshold of 0.5

5.3 Experiment Setup

5.3.1 Experiment platform

We implement our Faster R-CNN in Keras [74], which is a high-level neural network API written in python. In the backend, we use TensorFlow platform [75], which is an end-to-end open source platform for machine learning designed by Google.

5.3.2 Computing Resources

For the computing resources, we deploy our proposed models in Compute Canada [2] and run all our experiments in Cedar Cluster. In our experiments, we allocate memory as 96000 megabyte per node, and use NVIDIA P100-PCIE-12GB GPU to train our models. Due to the restriction of Compute Canada, we can only submit two jobs at the same time for running.

5.3.3 Experiment statistics

In order to reproduce the same experiments and get the results in our thesis, we list the statistics of the experimental environment in **Table 5.2**.

[2]https://www.computecanada.ca/

5.4 Results

5.4.1 Baselines and Experiment Settings

For comparing with the existing works on food detection model, we choose two baseline models to perform the same task. The first one is a R-CNN [65] based detection model, which is a fundamental approach in object detection. The second model is a CNN-based food image segmentation model by Shimoda er al. [76] as the baseline model. We refer it as BP-based model in our experiments. The way in this baseline model to generate the region proposals is by using selective search. For each region proposal, the food area is estimated by saliency maps generated from a CNN. And we use the non-maximum suppression (NMS) to unified the overlapped areas. This work can be modified to a detection model since it is a segmentation model, which classify the the object pixel-by-pixel.

In order to make the comparison more accurate, we try to keep the environment setting in the same way as [76] did. We split the dataset UEC-FOOD100 in the same way, making 80% of them as training data and the rest 20% as testing samples. We also try to train our model as the same iteration as the baseline model did in their paper, with the momentum of 0.9 and weight decay rate of 0.0005. For the first 30000 iterations, we use the learning rate of 0.001, and for the last 10000 iterations, we use 0.0001.

5.4.2 Detection Results

UEC-FOOD100 Dataset

In the first place, we try to train our model in 40000 iterations as the baseline models [76] did. However, in reality, it takes 15 days in total to train 239 epochs in our experiment devices. Due to the restriction of our computing resource, we can not train them the same iterations as the baseline models did. We show the training loss for the first 239 epochs in the UEC-FOOD100 dataset in **Figure 5.3**.

In this experiment, we show the mAP of the total 100 food classes in the dataset. In order to compare with the baseline model in [76], we also calculate the mAP over 53 classes of food, which these classes contain more than 10 images in the test dataset, and over 11 classes which has more than 50 images in the test dataset.

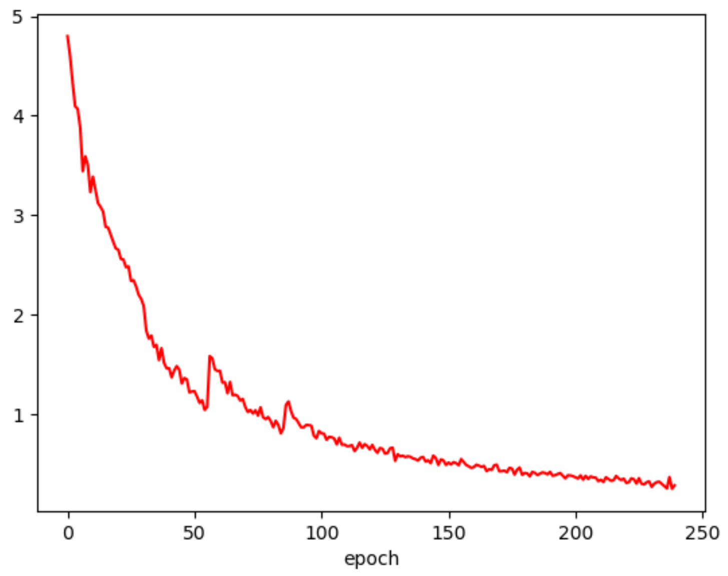


Figure 5.3 Training loss for the 239 epochs during training step in UEC-FOOD100 dataset

We show the result with UEC-FOOD100 dataset in **Table 5.3**. The baseline results are cited from their original paper [76]. From the table we can see, even though we train our model way less than the baseline models did, we still outperform the R-CNN model in the second category of experiment, which is the experiment that the test set contains more than 10 images in each of the classes of food. The reason why the performance gets better from category 1 to category 3 is that with more images in the test set indicating that there are more images, in other words, training data, in these classes than in the other classes. And convolutional neural network has the tendency to predict the class with more training data.

From our perspective, the reason for the low mAP in these experiments is the short training time and low power of computing resource. Usually it takes 40000 iterations to train a Faster R-CNN, while we can only train it as 250 iterations. Although the overall mAP is low, we keep positive attitude towards this direction. The reason is that we find out the classification accuracy is high if the region proposal network can propose the regions correctly. During the training process, the classification accuracy could reach to 98% as long as the region proposal network provide it with the correct bounding boxes. We believe that with the longer training time and more powerful computing resource, we can achieve a much better result than what we currently have.

Table 5.3 The results in UEC-FOOD100 dataset. Category 1 is the experiment that test with all the classes of food. Category 2 is the experiment that test with 53 classes (more than 10 items in test dataset). Category 3 is the experiment that test with 11 classes (more than 50 items in test dataset).

mAP(%)	Categories 1	Category 2	Category 3
RCNN-based model	26.0	21.8	25.7
BP-based model	49.9	55.3	55.4
Our model	17.6	22.7	25.3

Table 5.4 The time required to detect all the food items in one images. Both R-CNN based model and BP-based model using selective search method to generate region proposals, while our method based on Faster R-CNN, using region proposal network to generate the region proposals.

Model	Time(s)
RCNN-based model	45.6
BP-based model	2.3
Our model	0.3

Here is some examples of successfully detected samples in **Figure 5.4**. We leave some further experiments to the future work.

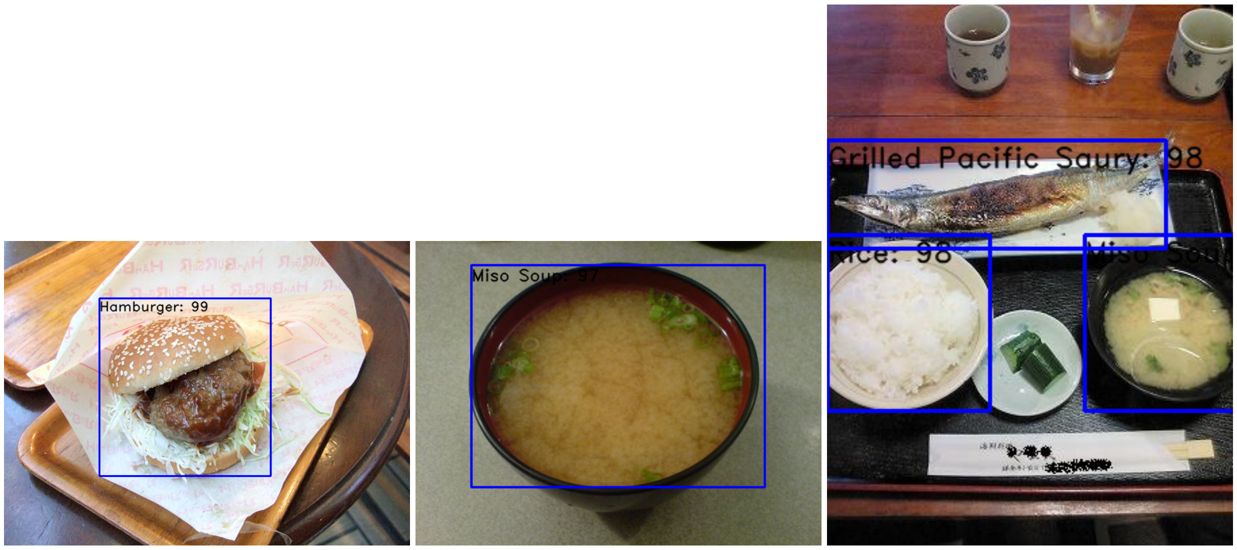


Figure 5.4 Examples of the detection result from UEC-FOOD100

While the measurement of mAP illustrates how well we detect each food items, the measurement of speed of detection is also a vital indicator for a detection system. A user-friendly food detection system requires immediate response for the input of food images, we can not ask them to wait for 1 minute or 30 second to see the nutrition estimation of their meal. So we also calculate how long it costs to recognize the food items from one single image, and compare them in **Table 5.4**.

From the result we can see. Our model outperforms the other two approaches in the aspect of speed. The time required to detect food items from one image is around 47 second for the R-CNN based model. The reason is that it generate the region proposals using selective search [66] algorithm, and classifies each of the proposals using separate SVMs model, which takes a huge amount of time. The BP-based model runs in around 2 second per image, which is tolerable compared with the former model. Most of the time spent in the detection process is also in the region proposals part, since it uses the same algorithm, the selective search, as the R-CNN based model. Our model however, detects the food items much faster than the other two models. The reason for that is we use region proposal network (RPN) [70] instead of selective search, to generate the region proposals, and simultaneously train it with our classification module.

UEC-FOOD256 Dataset

We also test our model in UEC-FOOD256 dataset, which contains 256 categories of food items. As for comparison, we train our models using the same steps from the previous experiments, and test them in the same categories. The total training iterations is 100 epochs. However, as the number of classes of food items increases, the number of categories with more than 10 items in the evaluation dataset also increases along with it. As well as the number of categories with more than 50 items. We calculate the mAP over 132 categories with more than 10 items and over 21 categories with more than 50 dishes in the evaluation dataset.

We show the result with UEC-FOOD256 dataset in **Table 5.5**. From the table we can see that the mAP for all categories have decreased compared to the results of previous experiments in UEC-FOOD100 dataset, the reason we think is because the number of food classes has increased, making our model harder to recognize the food items from the photos and detect each of them. One of the other reasons is that many food categories in this dataset share very similar looks, such as *miso soup* and *beef miso soup*, which make it even harder to distinguish.

Table 5.5 The results in UEC-FOOD256 dataset. Category 1 is the experiment that test with all the classes of food. Category 2 is the experiment that test with 132 classes (more than 10 items in test dataset). Category 3 is the experiment that test with 21 classes (more than 50 items in test dataset).

mAP(%)	Categories 1	Category 2	Category 3
Our model	10.3	13.6	17.8

Table 5.6 The results in FOOD20-with-bbx

# of iteration	Top-1 accuracy (%)	Top-5 accuracy (%)
100	28.9	60.4
250	45.6	80.3
500	73.4	95.5

FOOD20-with-bbx Dataset

In order to test the robustness of our model in a larger range of food items, we train our model using a dataset with a large number of western food items photos. Manually we generate a new dataset called FOOD20-with-bbx, which is a dataset containing 20 categories of food from FOOD101 [18], bounded by the bounding boxes coordinates. We use the model pretrained by the UEC-FOOD100 dataset, fine tuning it with 80% of images in FOOD20-with-bbx dataset, and test the mAP for the rest of it. The result are shown in 5.6. With the number of iteration being 500, the top-1 accuracy in FOOD20-with-bbx dataset is 73.4% and top-5 accuracy comes to 95.5%.

Although the training iteration is still long enough, the result is comparatively well. From my perspective, there are mainly three reasons for that. The first one is that this dataset is small compare to UEC-FOOD100 and UEC-FOOD256. In FOOD20-with-bbx, we have 20 classes in total, and each class contain around 200 images. So we have about 4000 images in total, divided into 20 categories. The second reason is that in our dataset, there is exactly one food item in each image, which make it a single food detection task. The third reason is that the categories in FOOD20-with-bbx are mostly western food, which make it easier to distinguish from their shape and texture.

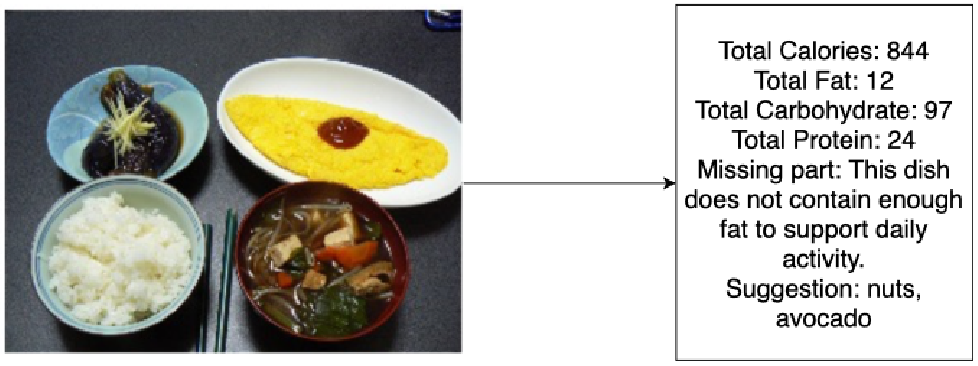

Total Calories: 844
Total Fat: 12
Total Carbohydrate: 97
Total Protein: 24
Missing part: This dish does not contain enough fat to support daily activity.
Suggestion: nuts, avocado

Figure 5.5 Example of result from dietary assessment system

5.5 Dietary Assessment

After detecting every food items within each photos, we are ready to analyze the nutrient contents for each food items. That is the final part of our entire food detection and analysis system.

The system takes a single image and the bounding boxes regarding each food items inside the image as input, calculating the amount of nutritional ingredients for the food items by mapping the detected food to the reference table. For example, if a 24 year-old man, with weight of 60 kilogram and height of 170 centimeter, with an activity level of moderate, a suitable amount of nutritional ingredients for a healthy diet will be 2399 calories per day, 311 grams of carbohydrate, 109 grams of protein and 80 gram of fat.

For each food items detected in the image, we assume it is 400 grams, which is a normal size for a single serve. **Figure 5.5** is an example of dietary assessment in our system.

In order to provide a healthy diet, we also design an automatic diet calculator. The calculator requires the user to input their age, gender, weight, height and activity level information. And it will automatically generate a healthy diet specifying the suitable amount of nutritional content needed everyday for that user.

5.6 Challenges & Opportunities

As can be seen in previous section (5.4), our proposed model does not perform well compared with some of the state-of-the-art models. There are still many unsolved problems and challenges remaining in our model. From our perspectives, the challenges can be concluded into the following three aspects.

5.6.1 Model Complexity

The first problem is model complexity. One of the major reasons why the performance is not good enough is because our model requires huge amount of time for training. During our experiments, we train our model in around 12 days, but still with only 250 epochs reached. One of the reasons why it is so slow is probably because we use VGG network in our proposed system. VGG-16 neural network is applied in the feature extraction and classification modules, which performs well in some of the state-of-the-art works. However, there is one major drawback for using VGG, that is it require a huge amount of time to train. Training a VGG net is extremely slow, since the architecture weights in VGG are quite large. Due to the number of fully-connected nodes, the weights of VGG with 16 layers is over 533MB, which make it slow to train the entire network.

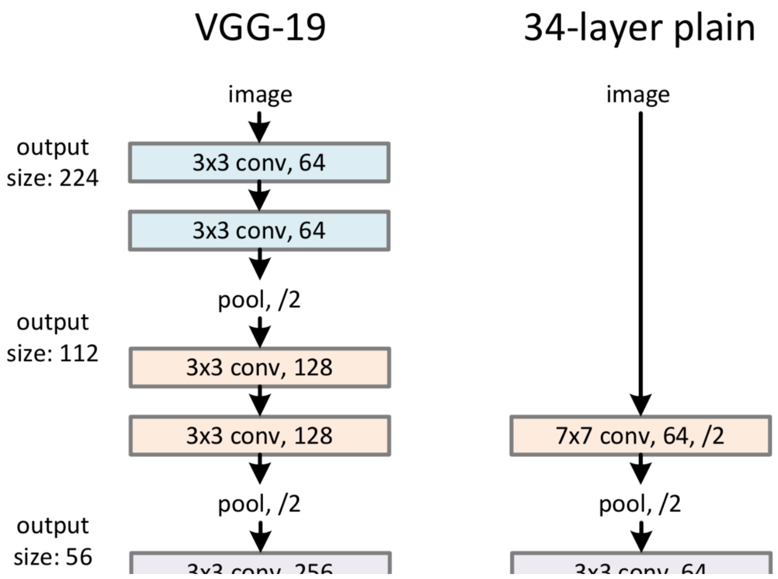


Figure 5.6 Part of the architecture from VGG-19 and ResNet-34 networks. Left is VGG-19 and right is ResNet-34.

In order to solve this problem, we will continue to work on modifying our system by replacing the VGG network with other state-of-the-art neural networks, for example deep residual networks (ResNet) [72]. In Kaiming He's presentation [77] about ResNet, he mentioned that "ResNet has lower time complexity than VGG-16/19". ResNet is constructed by several building blocks, and the model size is substantially smaller due to the usage of global average pooling layers rather than fully-connected layers. Usually the size of weights for ResNet with 50 layers is 102MB, which make it faster to train than the VGG network.

Besides the size, how these two neural networks apply convolution also affect the speed of training. From **Fig 5.6** [72] we can see, The ResNet reduces the number of rows and columns by a facter of 2 and it used aounrd 240M floating point operations per second (FLOPS). In VGG-19 neural network, the first two layers apply convolution on top of the full 244×244 image, which is relatively expensive. For the first layer, it does around 170M (FLOPs) and produces $62 \times 244 \times 244$ output from $3 \times 244 \times 244$ image. Since layer applies the same conv filter, with simple calculation, the number of FLOPs should be closed to $170M \times (64/3)$. As comparison, this layer alone has roughly as many FLOPs as whole ResNet-34 network.

5.6.2 Data Collection

The second challenge is the datasets we are using. During the evaluation process, the images in our training and testing datasets are mostly Japanese food items, which are hard to distinguish from the appearance. One of the major reasons is that some of the food from Japan have a high intra-class variance, while some of the others have low inter-class variance.

Foods with high intra-class variance means two foods may look very different from appearance, but they are actually belong to the same category. For example, some sushi is covered by salmon, while others are covered with beef or eggs. In our datasets, they are all contained by the class of sushi, but they look very different.

Foods with low inter-class variance means two foods substantially belonging to two different categories share the similar appearance. For example, both miso soup and tea are filled into cups, and look similar since they are all water with dark color. But actually, these two foods are totally irrelevant and should be classified into different categories.

Both foods with high intra-class variance and foods with low inter-class variance affect the performance of our proposed model. To solve this challenge, in this thesis we have generated a new dataset from FOOD101 [18], named FOOD20-with-bbx dataset. FOOD101 is a commonly used dataset for food classification, which contain 101 categories of food from western food. The problem of using this dataset in our experiments is that it does not have the information of bounding box coordinates for each food item. So that we manually labelled 20 classes from it with bounding box coordinates to generate a new dataset for food detection task. In the future work, we will continue labeling our FOOD20-with-bbx dataset, to expand this dataset to a larger range of food categories. Or combine our dataset with the Japanese food datasets such UEC-FOOD100 or UEC-FOOD256, to create a more diverse food dataset.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we delve into the food recognition and nutrition analysis problem. In particular, we explore several detection techniques in existing food related datasets, and generate a new dataset on our own. What's more, we build a automatic system that can generate a nutrition facts table for an image with food items.

To have a better understanding of why we need to have a food recognition and analysis system, we first introduce the background of how obesity affects the society, and elaborate three most common methods to assess our dietary intake. Each of these three approaches have obvious drawback, and it give us the motivation to build an automatic food recognition and dietary assessment system.

In order to build an food recognition system, object detection technique plays in important role in it. We first introduce three kinds of object detection streams, that is geometry-based approach, color-based approach and region-based approach, to have a brief taste of how object can be detected using computer vision methods. And we also summarize bunch of state-of-the-art food recognition and dietary assessment systems and models.

Following up, we take a close look at the detail of how most state-of-the-art object detection systems work. Started from the basis R-CNN approach, we illustrate the process of detecting object from one image. It first generates several region proposals using selective search algorithm. After that it uses a deep neural network to extract the feature maps from each of the region proposals, to classify them as different objects using class-specific linear SVMs. This model gives us a hint that given an image of food, we need to first propose several region proposals from it, and then classify these region proposals as different food items.

However, this method has its own limitation, that is the speed of generating the region proposals using selective search algorithm is so slow that it takes 47 second to detect one single image. This drawback brings us the model that we mainly rely on, the Faster R-CNN model. We go through the detail of Faster R-CNN model in the following sections. Taking the advantages of region proposal network and RoI pooling layer, Faster R-CNN can generate thousands of region proposals in a very short term.

Based on Faster R-CNN, we build an automatic food recognition and dietary assessment system. Our model consists of three steps. The first is to use the Faster R-CNN to generate many region of proposals, which separate the food items from the image. And then we use state-of-the-art deep convolutional neural network to extract the feature map for each region of proposals, to classify them as different food items. The final step is to apply modern technology-based dietary assessment tools to analyze the nutrition of the food, and come up with a health report for the meal.

Finally, we conduct several experiments to evaluate the efficiency and effectiveness of our system. First, we test our system in some existing datasets, such as UEC-FOOD100 and UEC-FOOD256, which are both food related datasets in Japanese style. We evaluate the performance of our model in the aspects of both accuracy and speed. Besides, we also build a new dataset, FOOD20-with-bbx, from an existing dataset FOOD101. We use our own dataset to evaluate the robustness of our model. According to our evaluation, our model can detect food items 10 times faster than the baseline models, and promise a validated future work direction.

6.2 Future Work

There are several ways we can keep exploring in the future. We divide the future into two main directions. The first is exploring the future of object detection algorithms, the second is to explore the future of food analysis system.

For the object detection algorithms, an interest and useful direction is to learn or detect objects in new or unseen classes, or incrementally learn to distinguish among subclasses after the "main" class has been learned. If the model is capable of learning new classifiers based on existing ones, it will greatly reduce the effort required to learn new object classes. A possible way to accomplish it is to use unsupervised learning methods. Some recent works have addressed these issues, mostly based on deep learning and transfer learning methods.

In order to improve the detection performance, scale adaption is also a promising direction. Objects usually exist in different sizes. To increase the robustness to scale changes, it is demanded to train scale-invariant, multi-scale or scale-adaptive detectors. In our thesis, the scale of the object is determined by the scale of the anchors. Although we have 9 anchors with different scales, we may still missing some giant objects. There are some implementation of multi-scale detectors which can produces multi-scale feature maps [78], or narrows representation differences between small objects and the large ones with a low-cost architecture provide insights into generating meaningful feature pyramid. [79].

For the food analysis system, one way of improving the performance is to introduce the weight estimation. After food items are detected, it is important to accurately estimate the weight of each food items in order to determine the nutrient context. Generally there re two ways to estimate the weight. One is to estimate the volume of the food, then use the density information for that particular food to estimate weight [80]. Another way is to directly estimate the weight of food using area information and training data.

Bibliography

[1] C. D. F. Craig M. Hales., Margaret D. Carroll and C. L. Ogden, *Prevalence of Obesity Among Adults and Youth: United States, 2015–2016*, 2016. [Online]. Available: https://www.cdc.gov/nchs/data/databriefs/db288.pdf

[2] E. A. Finkelstein, J. G. Trogdon, J. W. Cohen, and W. Dietz, "Annual medical spending attributable to obesity: payer-and service-specific estimates," *Health affairs*, vol. 28, no. 5, pp. w822–w831, 2009.

[3] S. Subramanian and A. Deaton, "The demand for food and calories," *Journal of political economy*, vol. 104, no. 1, pp. 133–162, 1996.

[4] D. Albanes, "Total calories, body weight, and tumor incidence in mice," *Cancer research*, vol. 47, no. 8, 1987.

[5] K. Ashton *et al.*, "That 'internet of things' thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[6] J. Stoppelman. (2004) Yelp. [Online]. Available: https://www.yelp.com/

[7] *Most Common Dietary Assessment Methods*, 2014. [Online]. Available: http://www.ucdenver.edu/research/CCTSI/programs-services/ctrc/Nutrition/Documents/Dietary-Assessment-Methods.pdf

[8] *Nutrition Data System for Research*, 2019. [Online]. Available: http://www.ncc.umn.edu/

[9] N. Slimani, P. Ferrari, M. Ocke, A. Welch, H. Boeing, M. Van Liere, V. Pala, P. Amiano, A. Lagiou, I. Mattisson *et al.*, “Standardization of the 24-hour diet recall calibration method used in the european prospective investigation into cancer and nutrition (epic): general concepts and preliminary results,” *European Journal of Clinical Nutrition*, vol. 54, no. 12, p. 900, 2000.

[10] W. C. Willett, L. Sampson, M. J. Stampfer, B. Rosner, C. Bain, J. Witschi, C. H. Hennekens, and F. E. Speizer, “Reproducibility and validity of a semiquantitative food frequency questionnaire,” *American journal of epidemiology*, vol. 122, no. 1, pp. 51–65, 1985.

[11] K. Poslusna, J. Ruprich, J. H. de Vries, M. Jakubikova, and P. van’t Veer, “Misreporting of energy and micronutrient intake estimated by food records and 24 hour recalls, control and adjustment methods in practice,” *British Journal of Nutrition*, vol. 101, no. S2, pp. S73–S85, 2009.

[12] F. Zhu, M. Bosch, T. Schap, N. Khanna, D. S. Ebert, C. J. Boushey, and E. J. Delp, “Segmentation assisted food classification for dietary assessment,” in *Computational Imaging IX*, vol. 7873. International Society for Optics and Photonics, 2011, p. 78730B.

[13] P. Pouladzadeh, P. Kuhad, S. V. B. Peddi, A. Yassine, and S. Shirmohammadi, “Mobile cloud based food calorie measurement,” in *2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*. IEEE, 2014, pp. 1–6.

[14] ——, “Food calorie measurement using deep learning neural network,” in *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*. IEEE, 2016, pp. 1–6.

[15] *MyFienessPal*, 2018. [Online]. Available: https://www.myfitnesspal.com/

[16] *Samsung Health.*, 2018. [Online]. Available: https://health.apps.samsung.com/

[17] J. Noronha, E. Hysen, H. Zhang, and K. Z. Gajos, “Platemate: crowdsourcing nutritional analysis from food photographs,” in *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 2011, pp. 1–12.

[18] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *European Conference on Computer Vision.* Springer, 2014, pp. 446–461.

[19] Y. Matsuda, H. Hoashi, and K. Yanai, "Recognition of multiple-food images by detecting candidate regions," in *Proc. of IEEE International Conference on Multimedia and Expo (ICME)*, 2012.

[20] Y. Kawano and K. Yanai, "Automatic expansion of a food image dataset leveraging existing categories with domain adaptation," in *Proc. of ECCV Workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV)*, 2014.

[21] W. E. L. Grimson, D. P. Huttenlocher *et al.*, *Object recognition by computer: the role of geometric constraints.* Mit Press, 1990.

[22] J. Mundy and A. Zisserman, "Geometric invariance in machine vision," 1992.

[23] M. J. Swain and D. H. Ballard, "Color indexing," *International journal of computer vision*, vol. 7, no. 1, pp. 11–32, 1991.

[24] V. V. Vinod and H. Murase, "Object location using complementary color features: histogram and dct," in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 1. IEEE, 1996, pp. 554–559.

[25] J. Matas, R. Marik, and J. Kittler, "On representation and matching of multi-coloured objects," in *Proceedings of IEEE International Conference on Computer Vision.* IEEE, 1995, pp. 726–732.

[26] B. Fan, L. Zhu, Y. Du, and Y. Tang, "A novel color based object detection and localization algorithm," in *2010 3rd International Congress on Image and Signal Processing*, vol. 3. IEEE, 2010, pp. 1101–1105.

[27] R. Hussin, M. R. Juhari, N. W. Kang, R. Ismail, and A. Kamarudin, "Digital image processing techniques for object detection from complex background image," *Procedia Engineering*, vol. 41, pp. 340–344, 2012.

[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *international Conference on computer vision & Pattern Recognition (CVPR'05)*, vol. 1. IEEE Computer Society, 2005, pp. 886–893.

[29] A. Torralba, K. P. Murphy, and W. T. Freeman, "Contextual models for object detection using boosted random fields," in *Advances in neural information processing systems*, 2005, pp. 1401–1408.

[30] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *Workshop on statistical learning in computer vision, ECCV*, vol. 2, no. 5, 2004, p. 7.

[31] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 1, pp. 36–51, 2008.

[32] S. Gould, T. Gao, and D. Koller, "Region-based segmentation and object detection," in *Advances in neural information processing systems*, 2009, pp. 655–663.

[33] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[34] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International journal of computer vision*, vol. 42, no. 3, pp. 145–175, 2001.

[35] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *Advances in neural information processing systems*, 2001, pp. 831–837.

[36] P. F. Felzenszwalb, "Representation and detection of deformable shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 208–220, 2005.

[37] T. Jiang, F. Jurie, and C. Schmid, "Learning shape prior models for object matching," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 848–855.

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

[39] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

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

[41] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[42] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[43] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2169–2178.

[44] F. Moosmann, E. Nowak, and F. Jurie, "Randomized clustering forests for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1632–1646, 2008.

[45] S. Singh, A. Gupta, and A. A. Efros, "Unsupervised discovery of mid-level discriminative patches," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 73–86.

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

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

[48] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Munari, M. Mordonini, and S. Cagnoni, "Food image recognition using very deep convolutional networks," in *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*. ACM, 2016, pp. 41–49.

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

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

[51] S. Ao and C. X. Ling, "Adapting new categories for food recognition with deep representation," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. IEEE, 2015, pp. 1196–1203.

[52] Z. Ge, C. McCool, C. Sanderson, and P. Corke, "Modelling local deep convolutional neural network features to improve fine-grained image classification," in *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015, pp. 4112–4116.

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

[54] ——, "Foodcam-256: a large-scale real-time mobile food recognitionsystem employing high-dimensional features and compression of classifier weights," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 761–762.

[55] T. Ege and K. Yanai, "Image-based food calorie estimation using knowledge on food categories, ingredients and cooking directions," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*. ACM, 2017, pp. 367–375.

[56] S. Fang, Z. Shao, R. Mao, C. Fu, E. J. Delp, F. Zhu, D. A. Kerr, and C. J. Boushey, "Single-view food portion estimation: Learning image-to-energy mappings using generative adversarial networks," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 251–255.

[57] A. Eldridge, C. Piernas, A.-K. Illner, M. Gibney, M. Gurinović, J. de Vries, and J. Cade, "Evaluation of new technology-based tools for dietary intake assessment—an ilsi europe dietary intake and exposure task force evaluation," *Nutrients*, vol. 11, no. 1, p. 55, 2019.

[58] B. Liu, H. Young, F. L. Crowe, V. S. Benson, E. A. Spencer, T. J. Key, P. N. Appleby, and V. Beral, "Development and evaluation of the oxford webq, a low-cost, web-based method for assessment of previous 24 h dietary intakes in large-scale prospective studies," *Public health nutrition*, vol. 14, no. 11, pp. 1998–2005, 2011.

[59] J. E. Cade, "Measuring diet in the 21st century: Use of new technologies," *Proceedings of the Nutrition Society*, vol. 76, no. 3, pp. 276–282, 2017.

[60] M. Carter, V. Burley, and J. Cade, "Development of 'my meal mate'–a smartphone intervention for weight loss," *Nutrition Bulletin*, vol. 38, no. 1, pp. 80–84, 2013.

[61] M. C. Carter, V. Burley, C. Nykjaer, and J. Cade, "'my meal mate'(mmm): validation of the diet measures captured on a smartphone application to facilitate weight loss," *British Journal of Nutrition*, vol. 109, no. 3, pp. 539–546, 2013.

[62] D.-H. Wang, M. Kogashiwa, S. Ohta, and S. Kira, "Validity and reliability of a dietary assessment method: the application of a digital camera with a mobile phone card attachment," *Journal of nutritional science and vitaminology*, vol. 48, no. 6, pp. 498–504, 2002.

[63] C. K. Martin, J. B. Correa, H. Han, H. R. Allen, J. C. Rood, C. M. Champagne, B. K. Gunturk, and G. A. Bray, "Validity of the remote food photography method (rfpm) for estimating energy and nutrient intake in near real-time," *Obesity*, vol. 20, no. 4, pp. 891–899, 2012.

[64] F. Zhu, M. Bosch, C. J. Boushey, and E. J. Delp, "An image analysis system for dietary assessment and evaluation," in *2010 Ieee International Conference on Image Processing.* IEEE, 2010, pp. 1853–1856.

[65] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[66] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

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

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

[69] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[70] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[71] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

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

[73] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3. IEEE, 2006, pp. 850–855.

[74] F. Chollet *et al.*, “Keras,” https://keras.io, 2015.

[75] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[76] W. Shimoda and K. Yanai, “Cnn-based food image segmentation without pixel-wise annotation,” in *International Conference on Image Analysis and Processing*. Springer, 2015, pp. 449–457.

[77] K. He, *Deep Residual Networks*, 2016.

[78] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[79] J. Liu, S. Zhang, S. Wang, and D. N. Metaxas, “Multispectral deep neural networks for pedestrian detection,” *arXiv preprint arXiv:1611.02644*, 2016.

[80] S. Kelkar, S. Stella, C. Boushey, and M. Okos, “Developing novel 3d measurement techniques and prediction method for food density determination,” *Procedia Food Science*, vol. 1, pp. 483–491, 2011.