An RFID-Based Visual Recognition System for the Retail Industry

by

Yongbin Sun

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Signature redacted

Author ....
Department of Mechanical Engineering
May 18, 2016

Signature redacted

Certified by..
Sanjay E. Sarma
Professor
Thesis Supervisor

Signature redacted

Accepted by..
Rohan Abeyaratne
Chairman, Department Committee on Graduate Theses

An RFID-Based Visual Recognition System for the Retail Industry

by

Yongbin Sun

Submitted to the Department of Mechanical Engineering
on May 18, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

In this thesis, I aim to build an accurate fine-grained retail product recognition system for improving customer in-store shopping experience. To achieve high accuracy, I developed a two-phase visual recognition scheme to identify the viewed retail product by verifying different types of visual features. The proposed scheme is robust enough to distinguish visually similar products in the tests. However, the computation cost of this scheme increases as the database scale becomes larger since it needs to verify all the products in the database. To improve the computation efficiency, my system integrates RFID as a second data source. By attaching an RFID tag to each product, the RFID reader is able to capture the identity information of surrounding products. The detection results can help reduce the verification scope from the whole database to the detected products only. Hence computation cost is saved. In the experiments, I first tested the recognition accuracy of my visual recognition scheme on a database containing visually similar products for different viewing angles, and my scheme achieved over 97.92% recognition accuracy for horizontal viewpoint variations of less than 30 degree. I then experimentally measured the computation cost of both the original system and the RFID-enhanced system. The computation cost is the processing time to recognize a target product. The RFID-enhanced system speeds up system performance dramatically when the scale of detected surrounding products is small.

Thesis Supervisor: Sanjay E. Sarma
Title: Professor

Acknowledgments

I am grateful for the help and support from my advisor, friends and family. They gave me confidence when I was at the bottom, and gave me advices when I was confused. Because of them, I had a very happy, fulfilling, and meaningful life at MIT. I will give my sincere thanks to all of them, and especially to the following people:

First, I would like to thank my advisor, Professor Sanjay Sarma, for his expert advices, constant encouragement throughout difficult problems, his patience, motivation, enthusiasm, immense knowledge and his guidance in all the time of my research. Without his support, I would have no way to finish this thesis and have a good time at MIT.

Second, I would like to thank my lab mates for their brilliance, enthusiasm, help and share. They gave me advices for my research, spent time discussing difficult problems with me, and helped me to understand the culture here. Thanks to Pranay Jain, Rahul Bhattacharyya, Isaac Mayer Ehrenberg, Dylan Charles Erb, Joshua E Siegel, Partha Sarathi Bhattacharjee, Eric William Fuentes and Jason S Ku.

I would also like to thank my mom and wife, who always support me, stand at the back of me no matter what happens. Without them, life would be totally different for me, they brought happiness and warm to me.

Next, I would like to thank my friends in MIT CSSA basketball team. We played games together, hanged out together and helped each other. Their personal quality and passion for success guide me to be a better myself. Thank you!

I also want to thank my best friends from primary school, middle school and college. You brought the best memory to me. The things that we experienced together would always stayed in my mind. Thank you all!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Online shopping has revolutionized today's retail industry, and become more and more prevalent. In the meantime, brick-and-mortar shopping has developed relatively slowly. In many situations, in-store customers cannot confidently decide which item to purchase like online shopping, which provides customers with sufficiently detailed information and comparison analysis. Imagine traditional brick-and-mortar shopping could take advantage of online shopping: it would be much easier for customers to find the most suitable products and make buying decisions satisfactorily. The improved shopping experience could increase sales revenues of retailers as well.

Inspired by this, I came up with a visual recognition solution for intelligent in-store shopping systems. When shopping in a brick-and-mortar store, customers need to point a phone camera towards the aisle. Then the system can automatically recognize the product being viewed, and display all its related information, such as its daily sales, customer reviews, coupon availability and comparison analysis with other similar products. Such product recognition system and the improved in-store shopping experience can drive the development of brick-and-mortar shopping.

1.2 Difficulties of retail product recognition

The aforementioned scenario shows the potential of intelligent in-store shopping systems for retail industry, but, to our knowledge, no such system has been developed successfully. To build such a system, different types of technologies for database management and retail product recognition are required. For database management, researchers and scientists have developed leading-edge technologies for online shopping systems, and we can take advantage of these techniques for in-store shopping systems. For product recognition, even though object recognition is a well-established problem in computer vision community, many factors make it still a difficult task for various applications. I will summarize three major difficulties below.

First, the fine-grained product recognition identifies each specific item. Many retail products under the same brand usually have similar visual appearance except for some detailed features, such as characters and digits. However, general-purpose object recognition algorithms cannot reliably distinguish products with similar visual appearance. The visual similarity increases the difficulty of fine-grained product recognition.

Second, robust object recognition algorithms are usually computationally expensive due to their complexity. These algorithms compute and compare different types of features between the product being viewed and the products in the database. Detection and extraction of different types of features involves significant computation, which aggravates difficulties for real-time applications.

Third, computation cost varies along with the database scale. To recognize the product being viewed, a system needs to verify all the products in the database. Nowadays, a typical supermarket carries more than 40,000 different products on average [9]. Working with a huge database requires a large amount of computation resources.

1.3 My solutions to retail product recognition

This thesis explores the engineering behind a prototype system that can recognize retail products in an accurate and quick fashion. The ability to accurately and quickly recognize products will facilitate the development of intelligent in-store shopping systems. In particular, I focus on the following aspects of our system and make contributions.

First, I propose a method to distinguish similar looking products by focusing on the detailed features that are significantly different among them. Such detailed features serve as the identification marks of products, so I call them signature patterns. My system locates and verifies the signature patterns by augmented normalized cross-correlation and Support Vector Machine (SVM), respectively. The visual recognition scheme that I developed makes normalized cross-correlation robust to geometric transformation variation, and SVM robust to illumination variation.

Second, to increase computation efficiency, I designed a two-phase scheme to recognize the viewed product. In the first phase, I compare the general features between the captured image and the candidates. Only the candidates with similar visual appearance as the product being viewed can enter the second phase. Not all the candidates can enter the second phase, so that computation efficiency is improved. In the second phase, I verify the signature patterns among similar looking products for accurate recognition.

Third, I integrated RFID technology to reduce the verification scope in the database. In our application scenario, I attached one passive RFID tag to each registered product and mounted a camera onto the patch antenna of an RFID reader. The RFID reader detects surrounding tagged products and passes the detection results to the server. The server only verifies the detected products instead of the whole database to identify the viewed product, which saves computation cost. One thing worth noting is that we cannot rely only on RFID for product recognition, because RFID is not able to precisely locate the tagged products.

1.4 Other applications of product recognition

A well-performed retail product recognition scheme can be beneficial to other applications as well. For example, a product recognition system can benefit retailers. To maximize sales, retailers place products in a way that matches the buying patterns of customers. This layout is called planogram. However, verifying that the actual products on shelves match planogram is time-consuming and laborious, so it is not done frequently. A product recognition system can automate the planogram verification process. Further, such a system can also help retailers detect out-of-stock and misplaced products, so that retailers can restock and rearrange products properly to avoid customer dissatisfaction and potential sales loss. Furthermore, a product recognition system can improve the shopping experience of visually impaired individuals. Today, 285 million people worldwide are visually impaired, and 90% of them live in developing countries [31], which do not regularly provide accommodations. The visually impaired have to rely on sighted people to help them with their daily activities like shopping. A visual recognition system could provide a low-cost way to help them collect items on their shopping lists independently.

1.5 Summary

I first described an application scenario of intelligent in-store shopping systems, and then discussed the main technical difficulties to develop such intelligent shopping systems. After that, I summarized the main contributions of my work by dealing with these difficulties, followed by other application scenarios of a product recognition system.

To improve computation efficiency, I chose to use RFID to prune the search, but this is not the only option. Some other techniques are also applicable for this purpose. For example, a system can first use indoor location, and only verify the viewing planogram instead of the whole database. The reasons why I choose RFID are that it is a low-cost solution, and easy to implement.

The remainder of this thesis will cover more details about our RFID-based visual object recognition system. Chapter 2 will discuss related work on object recognition. Chapter 3 will describe the structure of our system. Chapter 4 will show a series of tests and their corresponding results. Chapter 5 will present the final conclusions and future work.

Chapter 2

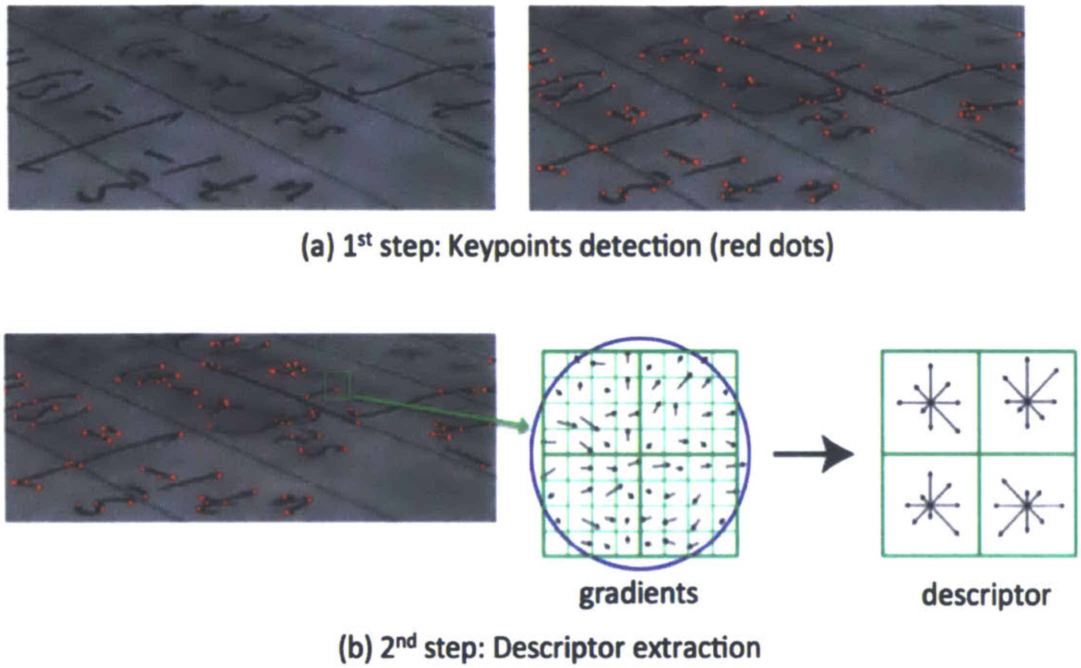
Related work

2.1 General-purpose object recognition techniques

Object recognition is a well-established problem, and computer vision researchers have developed a number of algorithms for recognition tasks. Feature-based algorithms are the most fundamental algorithms. They describe the appearance of an image by a set of feature vectors. Implementation of feature-based algorithms generally includes two steps. The first step is to detect the points that are scale-invariant within an image, called keypoints. The second step is to compute a vector to represent the local visual appearance around each keypoint by using the gradients of its neighboring region. The keypoints and their corresponding feature vectors form the feature sets. The basic steps of calculating image feature sets are shown in Figure 2-1.

Different feature-based algorithms have different approaches to extracting keypoints and computing feature vectors. Popular feature-based algorithms are Scale-Invariant Feature Transform (SIFT) [39], Speeded-Up Robust Features (SURF) [12], Oriented FAST and Rotated BRIEF (ORB) [41], Binary Robust Invariant Scalable Keypoints (BRISK) [37], and Fast REtinA Keypoint (FREAK) [11]. A summary of the key techniques of popular algorithms and their performance are shown in Table 2.1 and Table 2.2, respectively.

Based on the feature-based algorithms above, researchers have developed the bag-of-visual- words model [19, 23] to improve performance speed. The bag-of-visual-



(a) 1st step: Keypoints detection (red dots)

(b) 2nd step: Descriptor extraction

Figure 2-1: Steps to detect keypoints and extract feature vectors [39]

words models can generate a long feature vector to represent the visual appearance of an image [20, 47, 1]. While bag-of-visual-words models save computation cost, they come at the cost of recognition accuracy. One solution is to calculate a ranked list of object candidates by using bag-of-visual- words model, and then to perform feature-based algorithms within this ranked list [18].

Besides the bag-of-visual-words models, deep learning is another powerful tool for object recognition and image classification tasks. A deep learning model takes as input an image patch and outputs a probability distribution among different categories [10]. Krizhevsky *et al.* [36] developed a remarkable model by training a convolutional neural network model to classify 1.2 million high-resolution images into 1000 different categories in ImageNet LSVRC- 2010 contest. Their model contains 60 million parameters and 650,000 neurons. Later, to achieve better image classification accuracy, Girshick *et al.* [29] proposed a region-based convolutional neural network (R-CNN), which is further improved by Zhang *et al.* [48] for fine-grained object recognition purpose. However, a well-performed deep learning model usually requires a huge amount

Table 2.1: A summary of feature-based algorithms

Name	**Detector**	**Descriptor**	**Value in descriptor**
SIFT	LoG pyramid	Gradient values	Real value
SURF	Box filter pyramid	Haar wavelets	Real value
ORB	Oriented FAST	Gradient	Binary value
BRISK	AGAST	Moments	Binary value

Table 2.2: Computation time and precision for different algorithms for 1000 SURF keypoints [15]

Descriptor	SURF	SIFT	BRISK	ORB
Run time (ms)	117.1	448.6	10.6	4.2
Precision (%)	51.3	53.3	52.7	49.5

of training samples.

2.2 Retail product recognition

Many techniques for general-purpose object recognition have been applied for retail product recognition, and we will summarize related work below. In [38], Lin *et al.* built a product image search engine by implementing a multi-stage search scheme. They developed a dynamic weighting method to improve the recognition accuracy for low-quality input images. But this system requires testing images to have similar working conditions as training images, such as light conditions and viewing angles. This requirement cannot always be satisfied for most real situations. In [27, 28], George designed a system for identifying multiple retail products simultaneously in a hierarchical manner. His system first filters some possible labels for the imaged objects, and then obtains the final decision by minimizing an energy cost function. However, this system is not robust enough to distinguish objects with similar appearance. The same problem also exists for some commercial product search engines, such as Flow [3] powered by Amazon. Flow tries to identify products and display related information from Amzon.com. Flow achieves good performance for most textured products, and one working example is shown in Figure 2-2. In [43], Varol and Kuzu presented an approach to recognize retail products with high similarity in term of shape and design. They first segment the products from the background by developing a cascade of boosted classifiers by computing Histogram of Oriented Gradients (HOG) features, and then classify product logos by implementing a bag-of-visual-words model. However, their system is not robust enough to deal with scaling and rotation variation. Preprocessing, such as geometric distortion rectification, would improve their system. Zhang *et al.* [49] proposed a weighting scheme to handle scaling variations by adaptively assigning weights to different visual feature sets. They experimentally showed that their scheme outperforms many existing feature-based image retrieval approaches. But their scheme is computationally complex, since it needs to extract visual features of the same image under multiple scales.

Retail product recognition can provide an automatic approach for retail management as well. In [2, 6, 5], Carnegie Mellon University and Intel present a research project, aiming to improve in-store retail operation efficiency. They built a robot, called AndyVision, to detect out-of-stock, low-in-stock and misplaced products by using computer vision techniques, and to perform the shelf compliance task automatically. However, they only tested AndyVision in one campus store, and did not publish any technical paper.

Vision-based retail product recognition also provides a low-cost approach for improving the shopping experiences of visually impaired customers [45, 32, 13, 46]. In [13], Bigham *et al.* present a system, called VizWiz::LocateIt, to enable visually impaired individuals to use handheld devices with a built-in camera to find specific products. Their system is robust to different kinds of variations, but requires remote workers to outline the objects before the handheld device can localize them with computer vision techniques. Involving human-powered services increases the cost and decreases runtime efficiency. Another project toward real-time grocery product

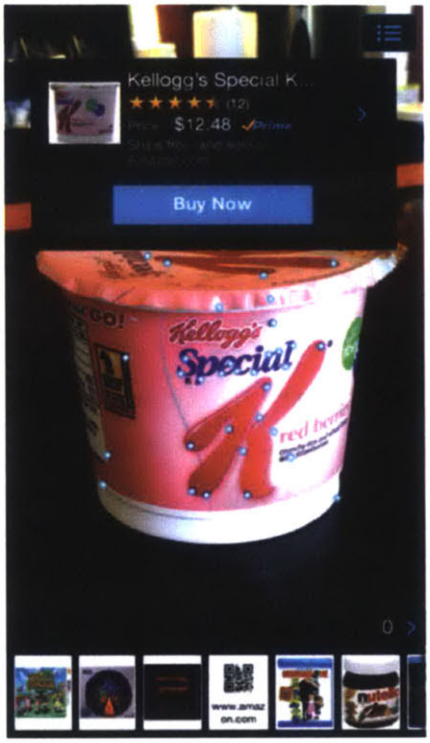


Figure 2-2: A working example of Amazon Flow

detection for blind shoppers is presented in [46]. They developed *ShelfScanner*, a mobile system that can detect predefined items on a shopping list from video streams. They implement an optical flow algorithm to deal with the scale variance problem, and develop a multiclass naïve-Bayes classifier to enhance recognition speed. However, their system is only useful when high-quality training data is available, and their system cannot handle products with similar visual appearances.

Most of the aforementioned vision-based systems are single-phase systems. In my approach, I implemented a two-phase scheme to verify signature patterns to distinguish visually similar products.

2.3 Senson fusion

One common disadvantage for all the vision-based retail product recognition systems is that they need to verify all the items in the database to recognize the viewed product, which is computationally expensive for a large database. To save computation cost, a second data source might be integrated. RFID, as a low-cost wireless communication technology, is a good candidate to serve this purpose. It has been extensively used in retail supply chain [26, 40, 34, 25], but I only focus on how to use RFID to reduce the verification scope in the database. Research following this idea can be found in [16, 14, 30, 35, 22]. In [14], Boukraa and Ando created an RFID-based 3D object recognition system, where RFID tags were attached to registered objects. They associated stereo-models with the ID information of RFID tags in the database. Their system downloads the 3D model for each detected RFID tagged object, and verifies the object according to the 3D model. A similar result can be found in [30], where Hontani *et al.* built a visual system to identify objects in the field of view. They combined RFID and CAD models for recognizing and tracking objects. Their system first estimates the initial pose and position of the detected object based on the RFID detection result, and then tracks the object according to the pre-downloaded CAD model. Both of above systems show how RFID benefits vision-based recognition tasks, but the authors only use geometric models in their work, which limits the recognition accuracy since many objects share common geometrical shapes.

2.4 Summary

This chapter discusses related work about general-purpose object recognition, retail product recognition, and the advantage of integrating RFID into a vision-based object recognition system. In my system, I follow the same idea in [14] and [30] to reduce the verification scope in the database by using RFID technology, but I used visual features instead of geometric shape to achieve high recognition accuracy. Further, I developed a robust scheme to recognize retail products, especially for visually similar products. The next chapter will cover all the details about the structure of my RFID-based retail product recognition system.

Chapter 3

The structure of the RFID-based retail product recognition system

This chapter describes the structure of my retail product system. The system consists of two different parts: the RFID detection and the visual recognition, and combines their results to accurately recognize the product being viewed. To control these two parts separately, I implemented two managers, the RFID manager and the computer vision manager, as shown in Figure 3-1. The RFID manager controls the RFID reader, fetches the detection results from it, extracts the data records of the detected products from the database, and pushes them to the computer vision manager. The computer vision manager verifies each of the detected candidates to identify the viewed product, and displays the recognition result to users. The reminder of this chapter will discuss RFID detection and visual recognition in detail.

3.1 RFID detection

An RFID framework includes RFID tags, an RFID reader and a host system, as shown in Figure 3-2. The host system controls the reader to detect surrounding RFID tags. The RFID manager in our system functions as the host system. In our application scenario, we attach an RFID tag to each product. Then the RFID manager learns about surrounding products by detecting the attached tags through the reader. This

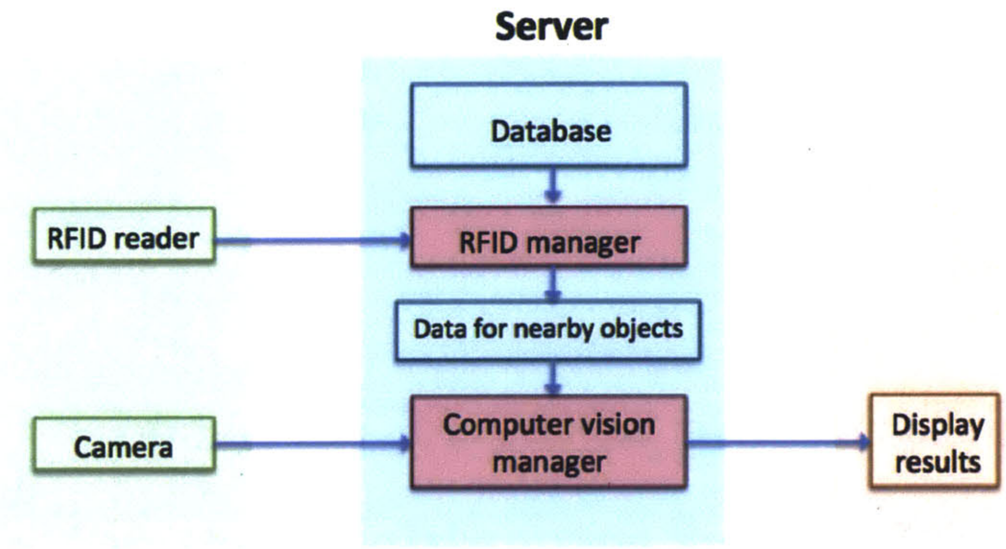


Figure 3-1: The structure of our system

section covers details about the RFID tag, the RFID reader and their communication protocol in our system.

3.1.1 RFID tag

I used UHF passive RFID tags for my system. A passive RFID tag does not contain any internal battery, and depends on RFID reader for operating power. The passive RFID tag consists of two parts: IC chip and antenna. The IC chip stores an ID

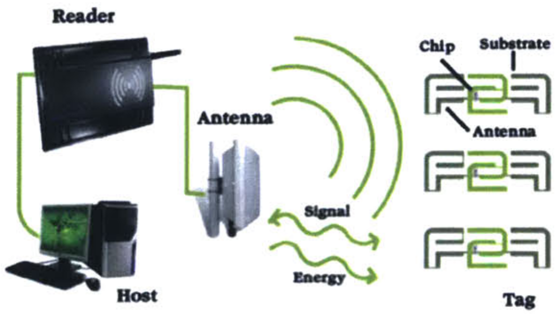
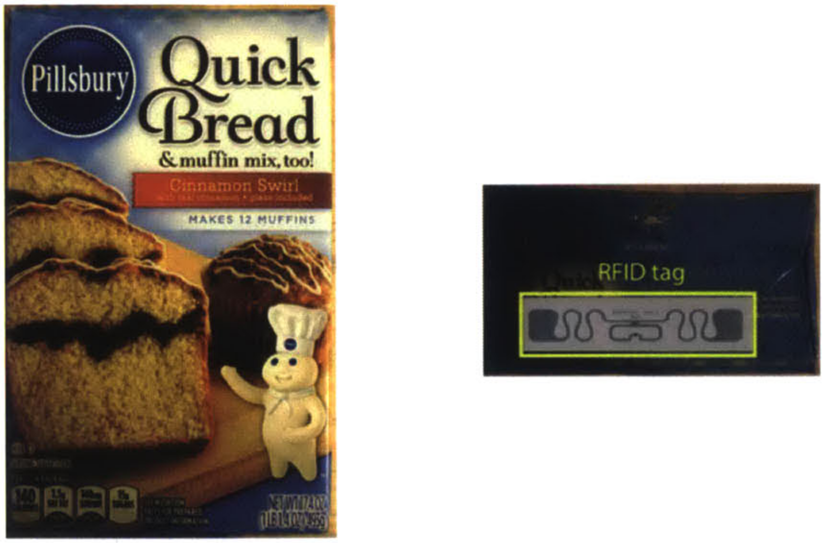


Figure 3-2: The framework of an RFID system [8]



(a) Front View

(b) Top View

Figure 3-3: A product with a passive RFID tag attached on the top

number. The antenna is attached to the IC chip, and transmits data using radio waves. In runtime, the tag is activated by radio waves of the RFID reader, and then transmits its ID number back to the reader.

In my system, I used "SMARTRAC 292_2 Belt" UHF passive RFID tags with the IC type of Impinj Monza 5, and attached one tag to every product as shown in Figure 3-3. Once the ID number is detected, we know that the attached product is nearby and it can potentially be the product being viewed. We cannot merely use RFID for recognition since that the RFID reader detects the tags in different directions but users can only view the products in one direction. The number of RFID detected products is usually more than the number of viewed products.

3.1.2 RFID reader

An RFID reader is a device used to capture the ID numbers of surrounding RFID tags. It is a radio transceiver, sending signals to the environment and receiving replied signals from responding tags. As the radio transmitter, an RFID reader modulates the carrier frequency with the desired baseband signal, and maintains the carrier

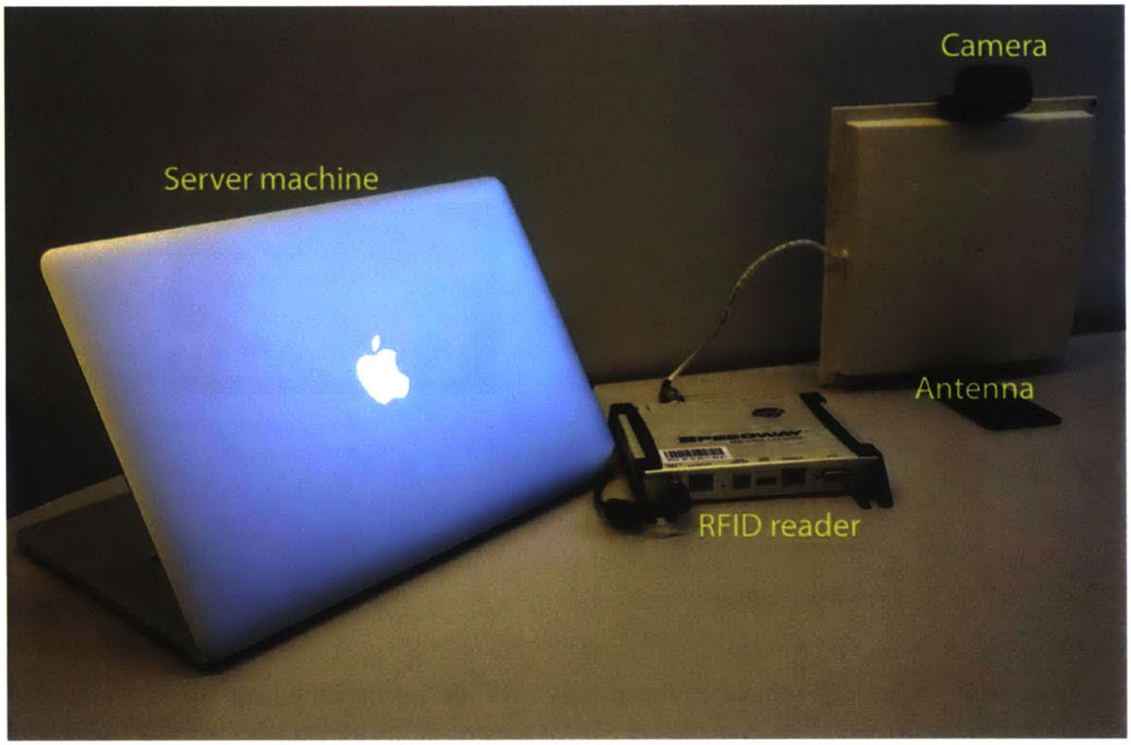


Figure 3-4: The prototype

signal at the desired frequency. As the radio receiver, an RFID reader can receive and interpret signals from responding RFID tags.

I used an Impinj Speedway Revolution RFID reader together with a patch antenna from Laird S9028PC series to detect tags. The Impinj Speedway Revolution RFID reader is a stationary UHF RFID reader, and is connected to a network port for data communication. The Laird S9028PC patch antenna provides directional pattern coverage. Figure 3-4 shows how the RFID reader and the patch antenna are configured in my prototype. I mounted a Logitech C525 webcam on top of the patch antenna for visual recognition.

3.1.3 RFID manager

The RFID manager is a program that connects my main system and the RFID reader. It communicates with the RFID reader by following the standard communication protocol, 'Low Level Reader Protocol' (LLRP). LLRP was ratified in 2007 to allow developers to have a common programmatic interface to RFID readers from different manufacturers. Besides following LLRP, my RFID manager also uses APIs from the SLLURP Library [7] for implementation. In runtime, the RFID manager fetches the RFID detection results from the reader by extracting the content of two fields, 'EPC' and 'Log_Time', in standard LLRP messages. The 'EPC' contains the ID numbers of all the detected tags, and the 'Log_Time' contains the latest time that the reader receives the respond from each corresponding tag. The RFID manager then keeps the detected ID numbers in a temporary list implemented by MySQL. This temporary list only maintains the ID numbers of the recent detected tags. Next, the RFID manager extracts the visual data of the products in the temporary list from the database, and pushes them to the computer vision manager for visual recognition.

3.2 Visual recognition

The computer vision manager receives a list of potential candidates from the RFID manager and recognizes the viewed product from them. It computes and verifies different types of visual features. The product being viewed is identified if it matches all the features of a candidate. To increase the computation efficiency and recognition accuracy, I developed a two-phase scheme for the visual recognition process.

3.2.1 Recognition pipeline

First of all, I introduce the pipeline of the visual recognition process. The computer vision manager recognizes the product being viewed in two phases. In the first phase, it compares the discriminative features between the viewed product and each candidate. The discriminative features can be any feature-based algorithm. Only the candidates with similar visual appearance as the viewed product can enter the second phase. In the second phase, I verify detailed patterns to distinguish similar looking candidates and make recognition decisions. The detailed patterns should be significantly different among similar looking products, and I refer them as signature patterns. One example of identifying the viewed product between two similar candidates is shown in Figure 3-5. In Figure 3-5, (a) and (e) are the first phase of each recognition process. In this phase, we match the SURF features between the captured image and the candidate. The number of the matched SURF features is large if the candidate has similar visual appearance as the viewed product. Otherwise this number is small, close to zero. By setting a proper threshold, I can filter out products with different visual appearance, and then only allows visually similar products to enter the second phase. In the second phase, I locate and verify the signature patterns of candidates for accurate recognition, as shown in Figure 3-5 (b) to (d) or (f) to (h). Each product in the database stores a template of its signature pattern, and a classification model for verification. The classification model is either an SVM model or a deep learning model in my system. For the SVM model, it gives a positive only to a pattern that matches its signature pattern, and a negative to any other image pattern. Hence a positive classification result indicates the identity of the viewed product, which is the case for Figure 3-5 (d). The following subsections will describe each step of the visual recognition process.

3.2.2 Match general features

In the first phase, the discriminative features are extracted and matched between the captured image and the potential candidates. The candidates with different appearance as the viewed product have very few matched features, and the candidates with similar visual appearance have many more matched discriminative features.

I choose SURF algorithm for this purpose, because it has a better tradeoff between speed and accuracy for describing visual appearance. The SURF algorithm first detects a set of keypoints, then creates a grid around each detected keypoint, and divides the grid cell into sub-grids. Within each sub-grid, SURF algorithm calculates its gradients and assigns them into a vector according to their orientations. This vector is the SURF feature around the corresponding keypoint. SURF feature vectors can be either 64 or 128 dimensional.

To find the matched SURF feature pairs between the viewed product and each candidate, I implemented a k-Nearest Neighbor (kNN) based algorithm. For each

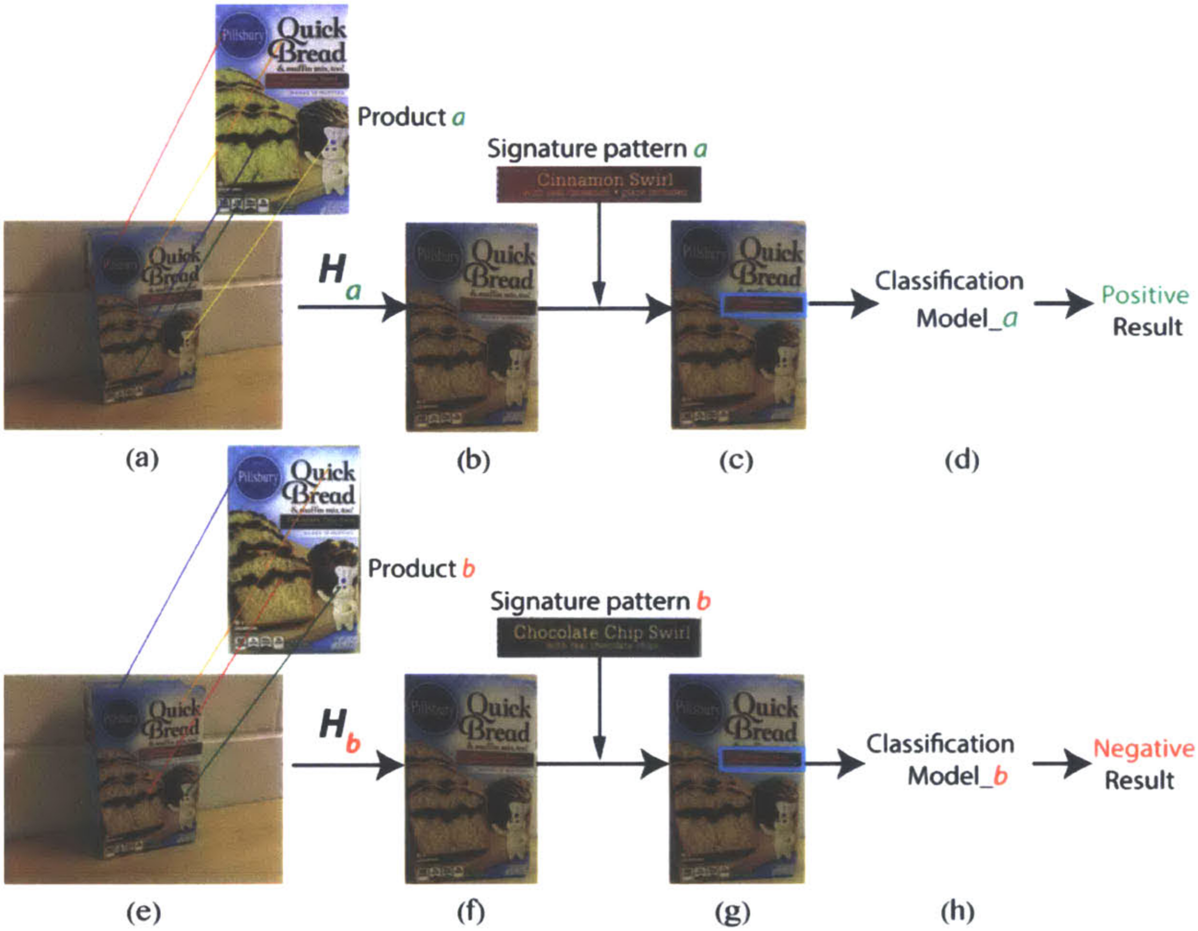


Figure 3-5: Identify the product being viewed between two visually similar candidates by SVM

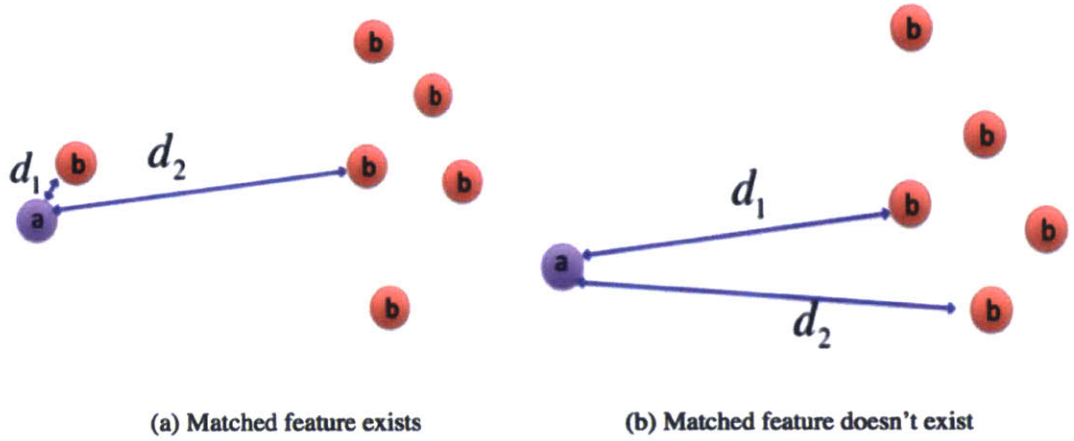
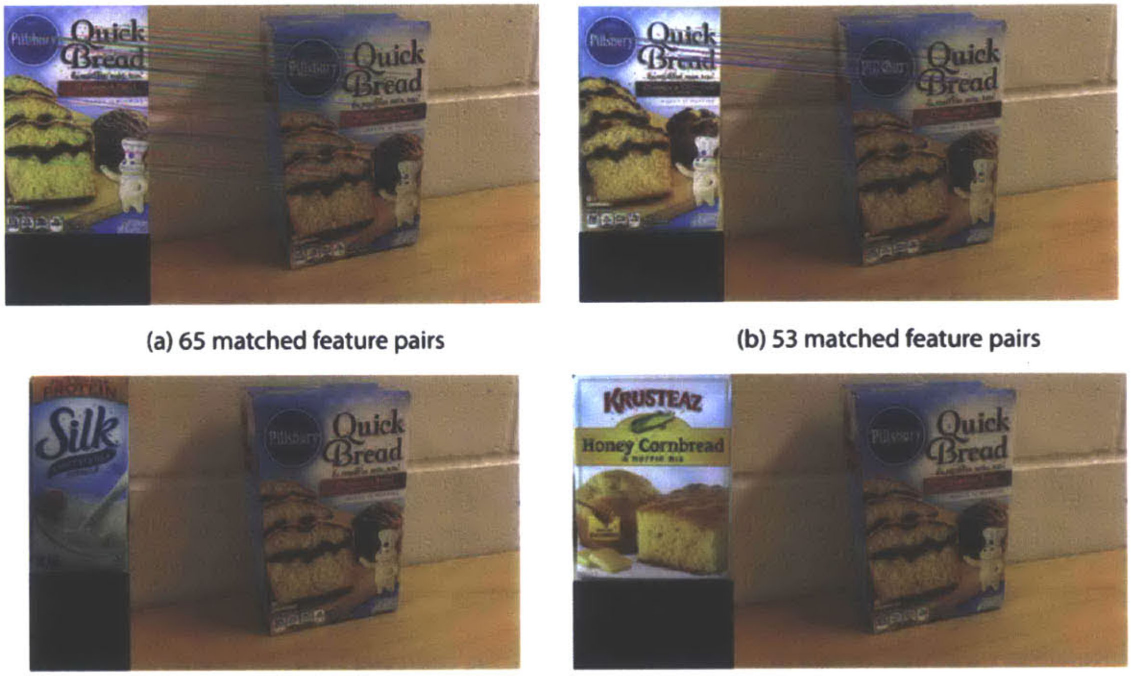


Figure 3-6: The illustration for feature matching

SURF feature of a candidate, I calculated the ratio between its Euclidian distances to its closest and second closest SURF feature in the captured image. A ratio smaller than the predefined threshold indicates the closest SURF feature as a match. This process is graphically illustrated in Figure 3-6. For a feature $\boldsymbol{a}$ in feature set $\boldsymbol{A}$, the computer vision manager detects its closest two features (their distances are expressed as $\boldsymbol{d_1}$ and $\boldsymbol{d_2}$, respectively) in feature set $\boldsymbol{B}$. It then calculates their ratio ($\boldsymbol{d_1}$ / $\boldsymbol{d_2}$), and keeps the $\boldsymbol{d_1}$ as a match to $\boldsymbol{a}$ if the calculated ratio is less than a predetermined threshold. This ratio ranges from 0 to 1. In Figure 3-6 (a), a matched descriptor b exists, and the ratio is small (close to 0). Contrarily, in Figure 3-6 (b), this ratio is relatively large (close to 1), and no matched feature exists. Increasing the predetermined ratio threshold introduces possible false positive feature pairs, and decreasing this ratio threshold leads to less matched feature pairs.

The candidates with similar visual appearance as the viewed product show much more matched feature pairs than the candidates with different visual appearance. Some examples are shown in Figure 3-7. The verification processes for visually different candidates are terminated early to save computation resource. In the second phase, I will verify signature patterns among visually similar candidates for accurate



(a) 65 matched feature pairs

(b) 53 matched feature pairs

(c) 0 matched feature pairs

(d) 0 matched feature pairs

Figure 3-7: Examples of different number of matched feature pairs

recognition.

3.2.3 Rectify geometric transformation

In the second phase, I verify the signature patterns of the remaining candidates. Signature patterns are distinguishable among visually similar products. For example, in Figure 3-7 (a) and (b), the signature patterns of both products are the image patches describing their ingredients on the middle-right part. If the viewed product has the same signature pattern as a candidate, then it is identified as this candidate. I use normalized cross-correlation to locate the most similar pattern of the signature pattern of a candidate from the viewed product. Normalized cross-correlation is robust to illumination variations, but vulnerable to geometric transformations, such as rotation, scaling, and affine transformation. Geometric transformations are introduced when we view products from different distances and angles. Hence we need to rectify all the possible geometric transformations before applying the normalized

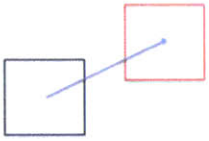
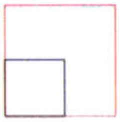
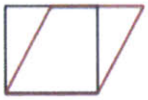
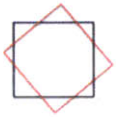
Translation		$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & y & 1 \end{bmatrix}$	**x is the displacement in horizontal direction, and y is the displacement in vertical direction**
Scale		$T = \begin{bmatrix} x & 0 & 0 \\ 0 & y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	**x is scale factor along the horizontal direction, and y is the scale factor along the vertical direction**
Shear		$T = \begin{bmatrix} 1 & y & 0 \\ x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	**x is the shear factor along the horizontal direction, and y is the shear factor along the vertical direction**
Rotation		$T = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$	**θ is the angle of rotation**

Figure 3-8: The basic homography transformations

cross-correlation.

In computer vision, the geometric relation between two images of the same planar surface is captured by a homography. When representing the homography as a 3 by 3 matrix ($\boldsymbol{T}$), and an image point ($\boldsymbol{x}$) as a three-dimensional vector in the form of $[\mathrm{x}, \mathrm{y}, 1]^{\mathrm{T}}$, we can relate two corresponding points ($\boldsymbol{x_1}$ and $\boldsymbol{x_2}$) in different images as $\boldsymbol{x_2} = \boldsymbol{T} \times \boldsymbol{x_1}$. Hence a new image under a specific geometric transformation can be obtained from the original image by applying the same homography matrix on each point. Basic homography transformations include translation, scaling, shearing and rotation (Figure 3-8), and complex transformations can be expressed by combing basic transformations.

When constructing our database, I extracted all the visual features from the front face of each product. If we set the front face image as the basic image, an image taken under different viewpoints and distances is considered as applying a corresponding homography matrix to the basic image. Therefore if we want to obtain the basic image from a random image, we only need to estimate the homography matrix, and apply it to the image inversely. To compute the homography matrix, we used the RANSAC algorithm [24] and the location information of the matched SURF features. The geometric transformation rectification process is illustrated in Figure 3-5 (b) and (c) separately.

3.2.4 Locate signature pattern

I use normalized cross-correlation to search for the most similar image pattern of the signature pattern of a candidate from the rectified product. Normalized cross-correlation is a standard way for pattern searching in computer vision. It slides a pattern template across all the positions within an image, and calculates a correlation score for each position. The correlation score is normalized to tolerate illumination variation. The normalized score ranges from -1 to 1, and the position with the highest score indicates the position of the most similar pattern of the template. However, normalized cross-correlation is not robust to any geometric distortion, and that is the reason we rectify geometric distortion before this step.

This process is graphically illustrated in Figure 3-9. A template pattern (a small rectangular pattern, $\boldsymbol{w}$) is moved within an image, $\boldsymbol{f}$, and a correlation score is computed for each position (i, j) according to Equation 3.1, where i = 0, 1, ..., M-1, j = 0, 1, ..., N-1; M and N are the height and width of the image; K and L are the height and width of template pattern; x goes from left to right, and y goes from top to bottom; $\bar{w}$ and $\bar{f}(i, j)$ are the average value of the template and the patch of the image within the sliding window, respectively. The size of template pattern should be smaller than the image.

$$C(i,j) = \frac{\sum_{x=0}\sum_{y=0}(w(x,y)-\bar{w})(f(x+i,y+j)-\bar{f}(i,j))}{\sqrt{\sum_{x=0}^{L-1}\sum_{y=0}^{K-1}(w(x,y)-\bar{w})^2}\sqrt{\sum_{x=0}^{L-1}\sum_{y=0}^{K-1}(f(x+i,y+j)-\bar{f}(i,j))^2}} \qquad (3.1)$$

Each product in the database stores a template of its signature pattern, and

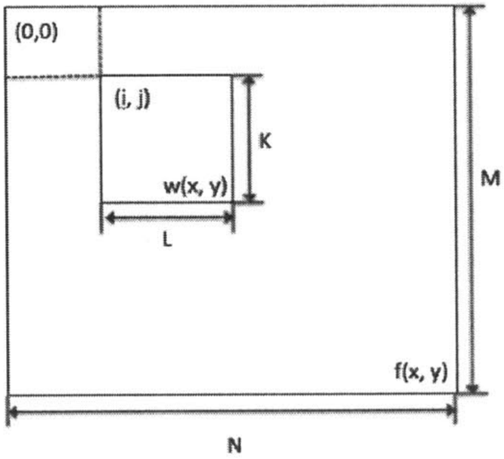
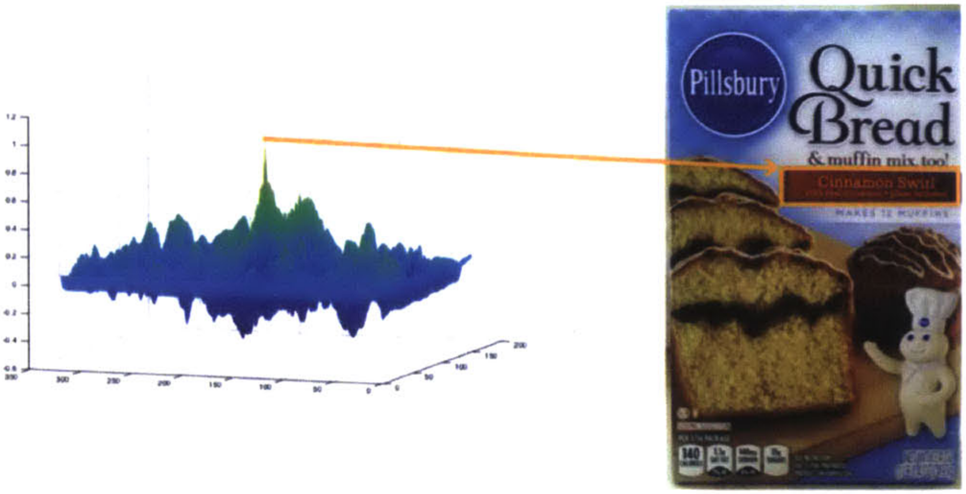


Figure 3-9: Normalized cross-correlation process

normalized cross-correlation uses this template to locate its most similar pattern in the rectified product. The most similar pattern is found at the position with the highest correlation score. One example is shown in Figure 3-10. Normalized cross-correlation is reliable for pattern searching once we rectify geometric distortion, but we cannot only use the correlation result for recognition since a false matched pattern may also have a high correlation score. For example, the correlation score for the true matched pattern is 0.732 in Figure 3-5 (c), and the correlation score for the false matched pattern is 0.686 in Figure 3-5 (g). Both correlation scores are high, and hard to distinguish. Further verification is required.

3.2.5 Verify signature pattern

In this subsection, I will discuss the techniques that I used for verifying the located signature patterns. I implemented two types of classifiers for this purpose: the Support Vector Machine (SVM) and the deep learning model. An SVM is a binary classifier. In my system, I trained one SVM model for each signature pattern, and the SVM model only gives a positive to the pattern that matches its corresponding signature pattern. But an SVM model needs a lot of space to store its support vectors. Later, I found that many signature patterns with different visual appearance belong to the



(a) Cross-correlation scores

(b) The rectified product

Figure 3-10: Locate the signature pattern based on normalized cross-correlation

same type, such as digital patterns, and it would be more spatially efficient if we could use one model to verify all the patterns of the same type. Therefore, I implemented a deep learning model to verify all the digital signature patterns. Training a well-performed deep learning model requires a huge number of training samples, and collecting them is a laborious task. So I use a publicly accessible handwritten digits dataset MNIST [4] to train my deep learning model. My current system only has one deep learning model for recognizing digital patterns due to the lack of training dataset for other pattern types. In this subsection, I will discuss the procedure to obtain well-performed SVM models and deep learning models, respectively.

SVM

An SVM model is formally defined as an optimized classifier that separates two categories of labeled training data (Figure 3-11). The SVM classifier is found by maximizing the margin between two data classes. An SVM model takes vector data as input, and all the input data must have the same dimension. In our case, the image pattern that matches the corresponding signature pattern is a positive sample, and other different image patterns are negative samples.

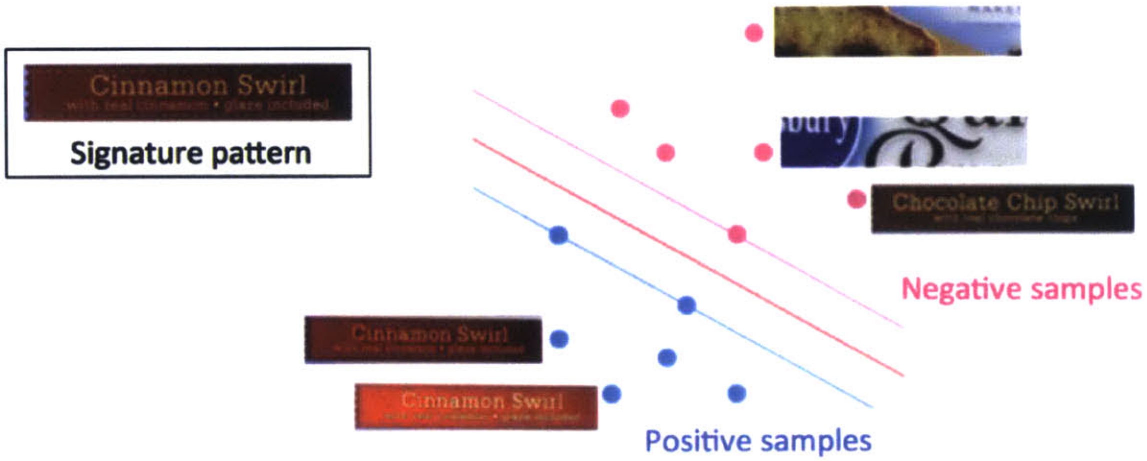


Figure 3-11: An SVM model

To fit the SVM model, I choose HOG feature [21] to describe the visual appearance of an image pattern, since the HOG feature is invariant to geometric and photometric transformations. The HOG feature of an image patch is determined by three factors: cell, block and stride, as shown in Figure 3-12. A cell is a small-connected square region. HOG algorithm generates a histogram for each cell by assigning the directions of its gradients into discrete angular bins. Adjacent cells form a block. The angular bins of each cell within a block are concatenated and normalized to generate a feature vector for the block. Finally, all the block features are concatenated to generate a complete feature vector of the image patch. The stride defines the pixel interval between two adjacent blocks.

Training a well-performed SVM model requires the HOG features from sufficient positive and negative training samples. In my case, the signature pattern of each product is the positive sample, and other different image patterns with the same dimension are the negative samples. The well-trained SVM model should only generate positive results to the patterns that match its corresponding signature pattern. In the training process, I could collect negative training samples from any portion of the product by cropping the patterns with the same dimension as the correspond-

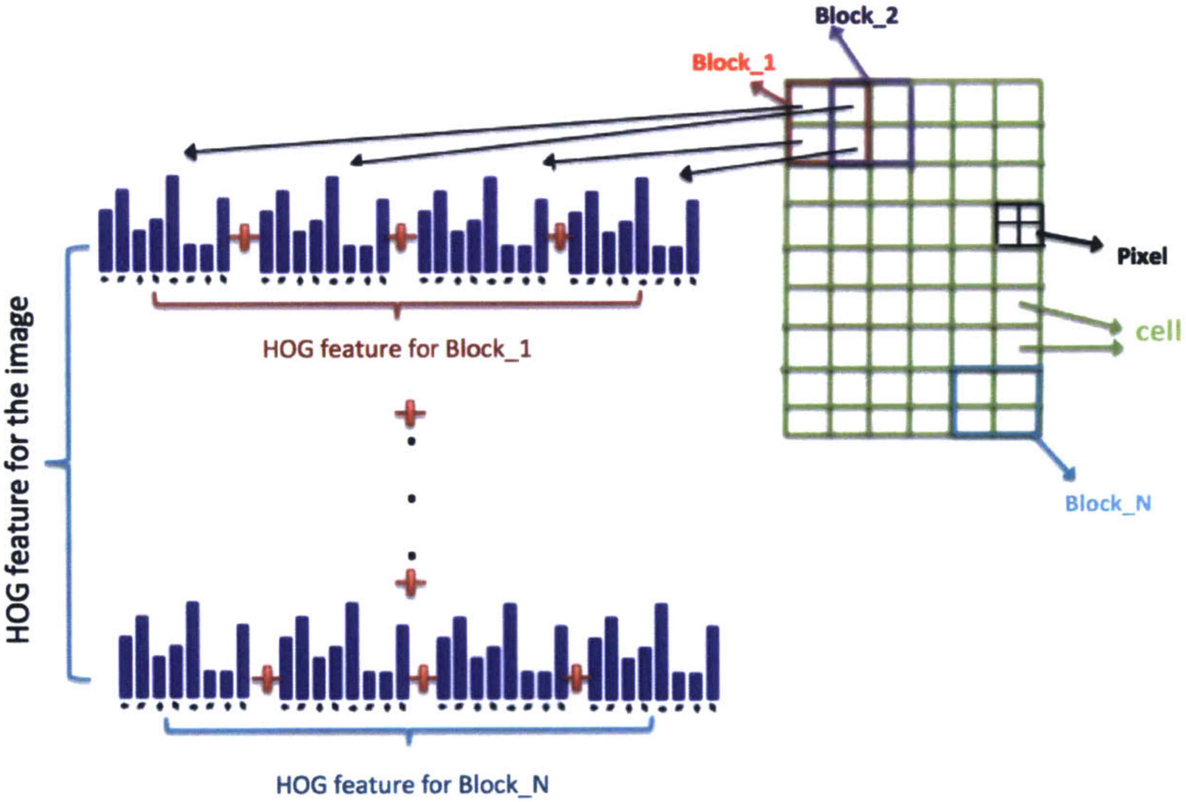


Figure 3-12: The HOG feature

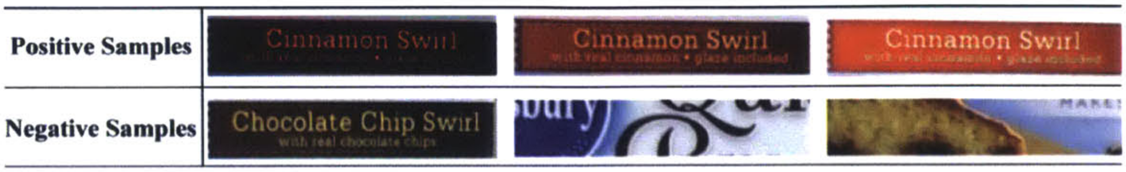


Figure 3-13: Training samples of Product ***a*** in Figure 3-5

ing signature pattern. To make the trained SVM model more robust to distinguish visually similar products, I also included the signature patterns from other similar looking products as the negative samples. However, the options for positive samples are very limited, since only the corresponding signature pattern can be the positive samples. To obtain a robust classifier, I need to involve sufficient positive samples under various variations. A well- performed SVM model should be illumination invariant in my case, since I have rectified geometric transformation, and only illumination variation remains. Sufficient positive training samples can be obtained by capturing the signature patterns under different light conditions. But manually controlling the light conditions to capture positive samples is very time- consuming and laborious. I then proposed a way to artificially obtain a large enough positive training samples. I convert the signature patterns from the Red-Green-Blue (RGB) color space to its Hue-Saturation-Lightness (HSL) color space to decouple color and illumination. In HSL color space, I modify their *Lightness* values, and convert the modified signature patterns back to the RGB color space. Artificially creating training samples in this way saves a lot of time for data collection. Some training samples for Product ***a*** in Figure 3-5 are shown in Figure 3-13.

The SVM model associated with each product gives a positive for a true matched signature pattern (Figure 3-5 (d)), and a negative for a false matched signature pattern (Figure 3-5 (h)). The positive verification result indicates the identity of the viewed product. If the viewed product is not in the database, instead of giving an incorrect result as standard image classification models, my system gives no positive result.

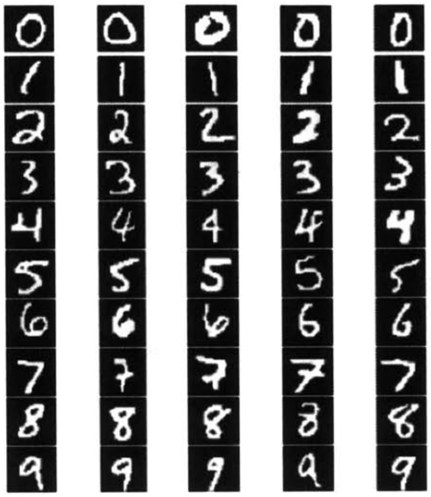


Figure 3-14: Samples in MNIST dataset

Deep learning model

The deep learning model is a powerful tool for classification tasks. A deep learning model has multiple layers to learn the deep features from an input data, and outputs a confidence distribution among different categories. In my system, I implemented a Convolutional Neural Network (CNN) to classify a digital pattern into one of ten digit classes from 0 to 9. A well- performed CNN model requires a huge amount of training samples, so I use the MNIST dataset to facilitate the process. The MNIST is a public dataset, and contains 70,000 handwritten digit images. Each image is 28 pixels by 28 pixels. Figure 3-14 shows samples in MNIST dataset.

I implemented an 8-layer CNN model (shown in Figure 3-15 with dimensions annotated) for the computer vision manager. The first five layers of my CNN model consist of convolutional layers and pooling layers to extract the deep features of an input image. The ReLU layer processes the deep features non-linearly by replacing negative values with zero, and leaving positive values unchanged. The latter part of this CNN is a fully connected neural network to classify the deep features into ten digit classes.

I implemented the Stochastic Gradient Descent (SGD) scheme to train this CNN model by shuffling the training dataset iteratively during the training process. Before

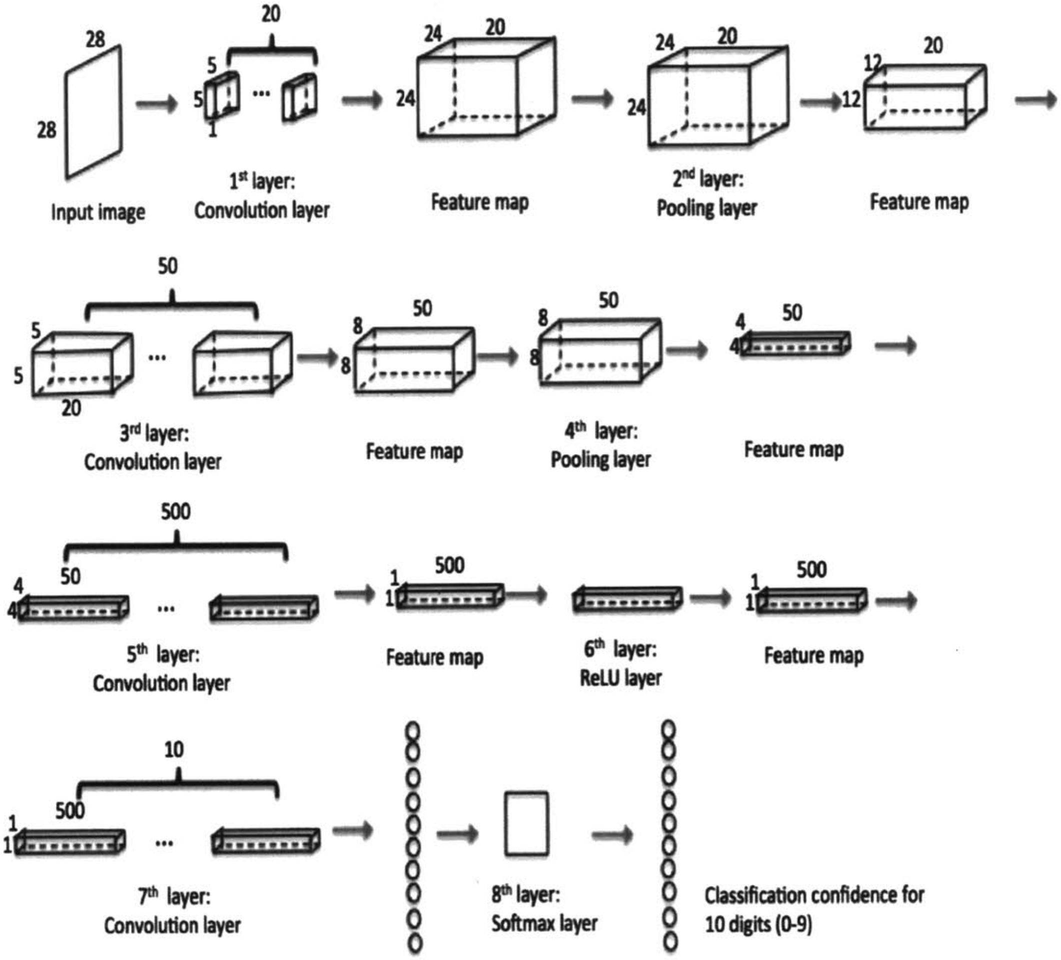


Figure 3-15: The structure of the CNN digit recognizer

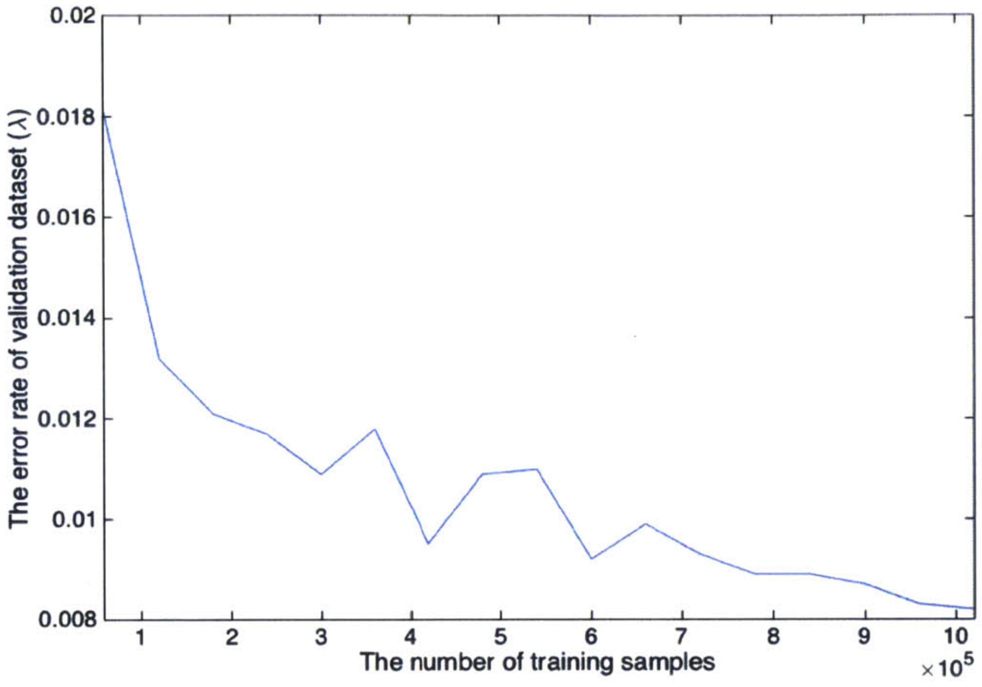


Figure 3-16: Error rate on validation dataset after each iteration

the training process, I initialized the model parameters according to the standard Gaussian distribution. During the training process, the model learns optimal parameters from an error back- propagation process. I calculated the error rate of the CNN model on validation dataset after each iteration process, and plot them in Figure 3-16. The training process stops when the error rate stops decreasing, and the validation error rate drops to 0.83% eventually. I used a GPU, CUDA toolkit and NVDIA cuDNN library to accelerate the training process.

Some preprocessing is required before we can send a digital pattern to the trained CNN model, since deep learning models are strict with the input data dimension. All the input data must have the same dimension as the training data, which are 28 pixels by 28 pixels. To further increase recognition accuracy, we need to centralize the digit because all the digits in the training samples are in the center. Lastly, we need to normalize the input data by subtracting it with the mean image of training samples. A complete process of recognizing a digit pattern is shown in Figure 3-17.

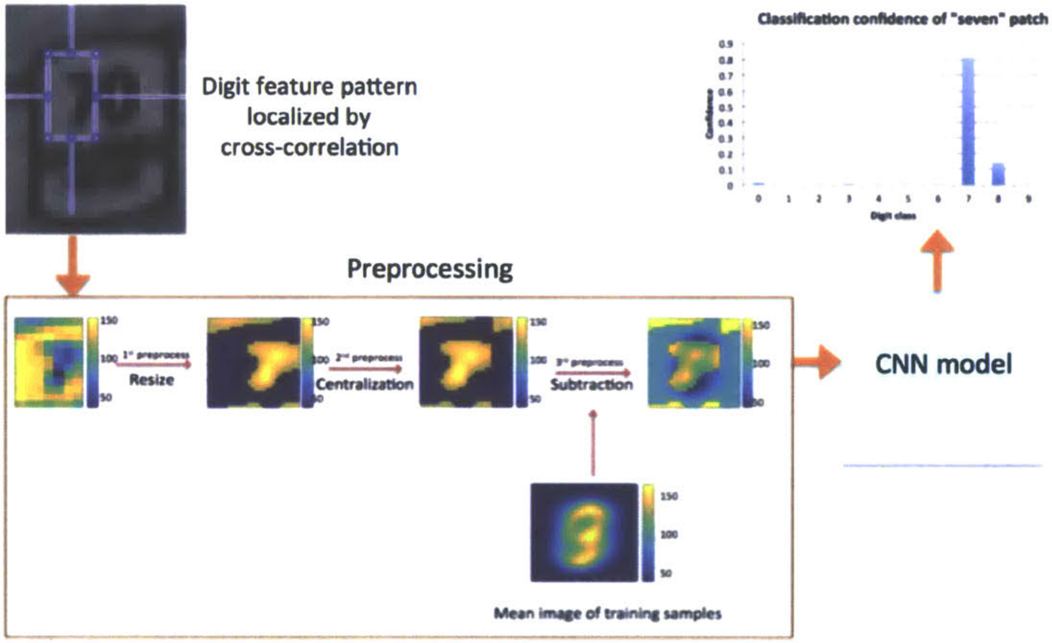


Figure 3-17: Preprocessing before a located digital pattern is sent to the trained CNN model

3.3 Summary

This chapter described the structure of my retail product recognition system. My system combines RFID detection and visual recognition to identify the product being viewed. By attaching one RFID tag to each product, RFID reduces the visual verification scope from the whole database to the detected products nearby. The computation cost is then saved, especially for large databases. I also developed a two-phase scheme to accurately and efficiently recognize the viewed product from all the potential candidates. In the first phase, I match the SURF features between the viewed product and candidates to check their visual similarity. Only the candidates with similar visual appearance as the viewed product can enter the second phase. In the second phase, I use normalized cross-correlation to locate the most similar pattern of the signature pattern of each candidate, and verifies it by either SVM model or deep learning model. The verification result indicates the identity of the viewed product.

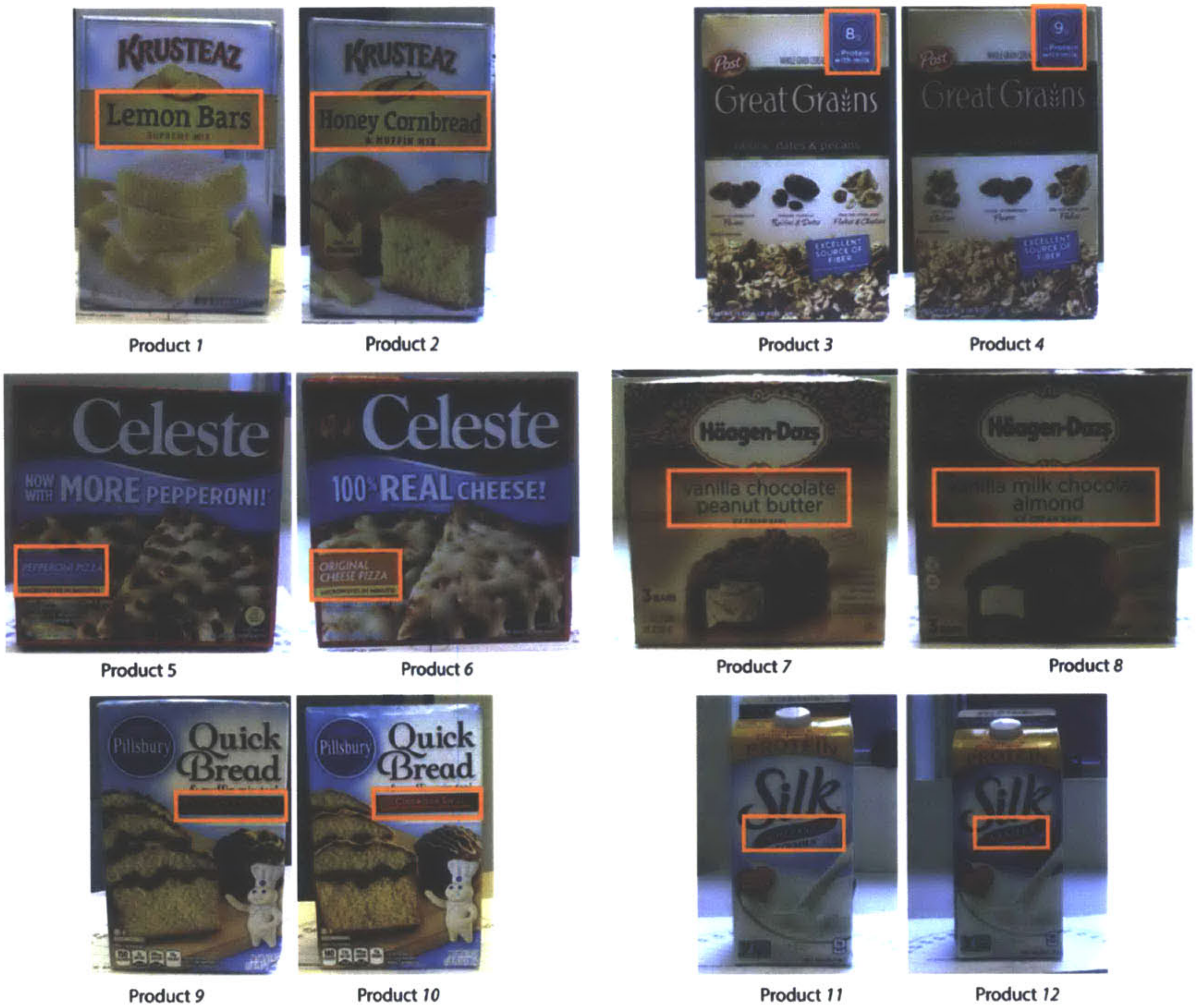
Chapter 4

Experiments

I conducted three experiments to test my retail product recognition system, and show their results in this chapter. In the first experiment, I tested the recognition accuracy of my system on visually similar products. I also compared the result between my system and other popular image recognition models, such as deep learning models and bag-of-visual-words model. In the second experiment, I introduced viewpoint variations to test my system. Different viewing angles cause different geometric transformations on the viewed product. My system can rectify the geometric transformation to some extent, and achieve high recognition accuracy when viewpoint variations are not severe. In the third experiment, I quantitatively describe the advantage of integrating RFID into my system. To better show the advantage, I created a system by following the same recognition algorithm as my system except that it does not use RFID. I recorded and compared the processing time of two systems for different number of surrounding products.

4.1 Experiment 1: Distinguishing visually similar products

This experiment tests the capability of my system to distinguish products with similar visual appearance. I collected 12 products for this experiment. The 12 products can



Product 1

Product 2

Product 3

Product 4

Product 5

Product 6

Product 7

Product 8

Product 9

Product 10

Product 11

Product 12

Figure 4-1: Samples of test products

be divided into 6 sets, and each set include 2 products that are visually similar. Samples of the 12 products are shown in Figure 4-1, and orange rectangles indicate corresponding signature patterns. The signature patterns of product 3 and product 4 are digital signature patterns, and are verified by the CNN model. SVM models verify the signature patterns of all the other products.

Currently, I pick the signature pattern of each product manually for each product in the database. But this process can be automated by detecting the patterns with maximum difference among similar looking products. The visual recognition process in our system verifies the signature pattern of each candidate to identify each viewed product. Verifying the signature patterns plays a significant role in distinguishing products with similar visual appearance. To show the robustness of our algorithm, I compare its recognition accuracy with other popular image classification models. These models include vgg-s, vgg-m, vgg-f [17], vgg-verydeep-16 [42], caffe-reference [33], AlexNet [36], and bag-of-visual-words models with different number of visual words. Except the bag-of-visual-words model, all the other models are convolutional neural network models with different structures. These convolutional neural network models are the top performing models on the ImageNet ILSVRC challenge dataset [44]. I used their pre-trained versions to learn the deep features of the collected samples of the 12 products, and trained a multiclass SVM classifier for each of them. This experiment tested one product at a time under ideal working conditions. I pointed the camera directly towards the test product, and kept this viewing angle during the whole process. I changed the background each time and involved slight scaling variations. I tested each product for 25 times for all the models, and calculated their average recognition accuracy on each product. Figure 4-2 plots the resultant average accuracy for each product for different models. My algorithm achieved 100% recognition accuracy for all the test products; both vgg-s and AlexNet achieved 97% accuracy on an average; all the other convolutional neural network models achieved over 92% accuracy averagely; bag-of-visual-words method with 400, 500 and 600 visual words performed the worst with the average recognition accuracy of 83%, 87% and 84%, respectively.

This test involves no variation, so that my algorithm could always locate the signature pattern of the truly matched candidate, and send it to the well-trained verification model. This stable performance benefits from the geometric transformation rectification process. The results of this experiment imply the feasibility of my proposed approach to train robust SVM models by artificially creating sufficient samples in HSL color space. However, the accuracy of my visual recognition approach decreases when the working condition involves severe variations, which will be discussed in the next subsection.

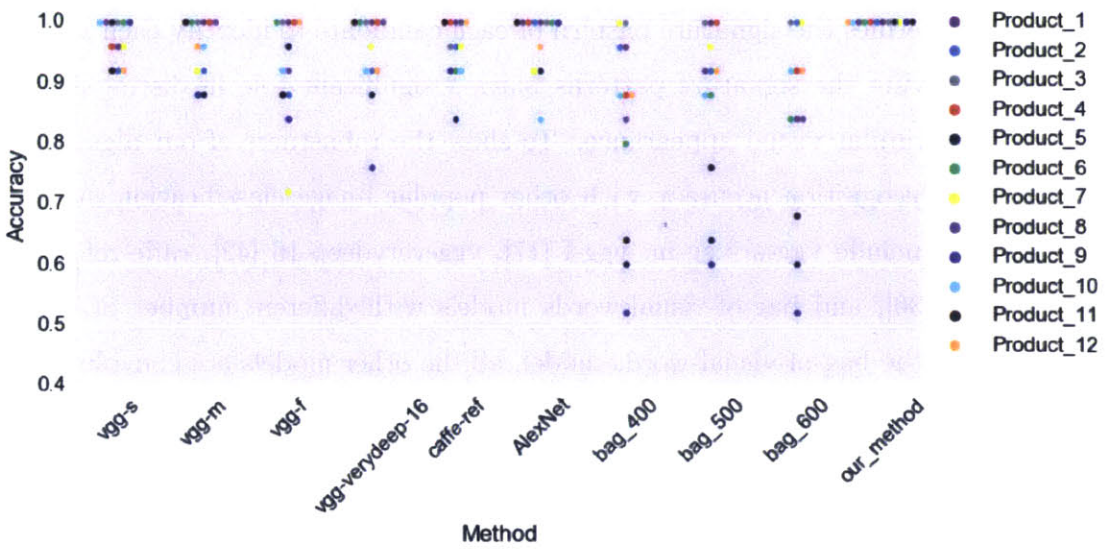


Figure 4-2: The recognition accuracy for different models

4.2 Experiment 2: Introducing viewpoint variations

This experiment involves horizontal viewpoint variations. The horizontal viewpoint variations can be expressed as different values of θ as shown in Figure 4-3. The viewing angle is an important factor in determining the recognition accuracy of a visual recognition system. When the viewing angle changes, the visual appearance and geometric shape of the viewed product change correspondingly. These changes influence our system from two aspects. First, when the viewing angle changes, the number of the matched SURF features between the viewed product and the true candidate decreases. If the number of the matched feature becomes less than the predefined threshold, my system would incorrectly consider that this candidate has different visual appearance as the viewed product, and terminate its verification process early. Second, when the geometric shape of the viewed changes severely, it is difficult to rectify the transformation completely. The remaining geometric transformation degrades the pattern searching performance of normalized cross-correlation.

In my system, I tried to rectify the geometric transformation by implementing the RANSAC algorithm to estimate the homography matrix, and warping the captured image inversely. To find the limit of my approach, I tested my system on 12

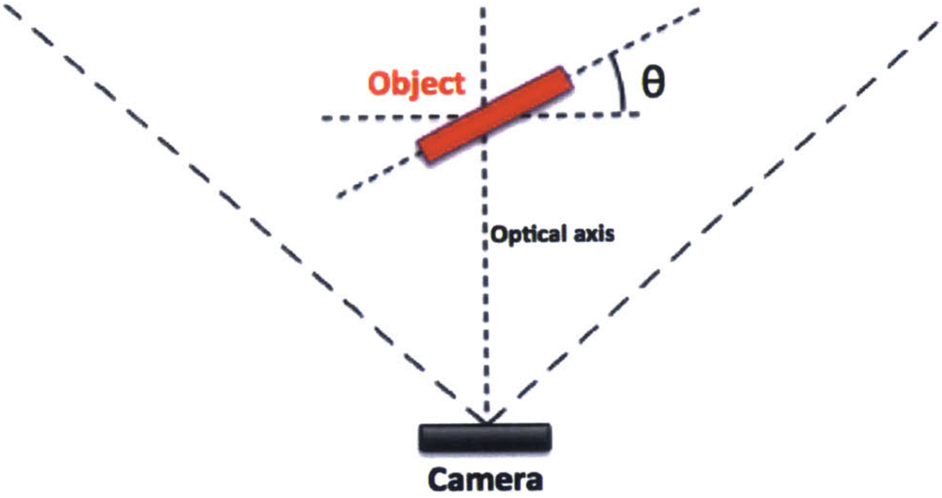


Figure 4-3: Measure the horizontal viewpoint variations

products under different viewing angles. I increased the viewpoint variation from 0 degree to 60 degree with a step of 10 degree. A protractor under the test product measures the angle change. For each viewing angle, I tested my system for 8 times for each product, and recorded their average recognition accuracies. Figure 4-4 shows the recognition results for different test products for different horizontal viewpoint variations. When the viewpoint variations are small, my system is able to rectify the geometric transformations, and maintain high recognition accuracy. The recognition accuracy drops rapidly for the viewpoint variations of greater than 40 degree, since we cannot rectify geometric transformations completely and missed some true matched signature patterns.

Further, I compared the average recognition accuracy for each viewpoint between my method and the aforementioned models. Figure 4-5 plots the comparison results of all the models. Deep learning models recognized products more accurately than bag-of-visual-words models, and my method outperformed all the other models for horizontal viewpoint variations of less than 40 degree. For viewpoint variations of less than 30 degree, the recognition accuracy of my method is over 97.92%. To regain high recognition accuracy for large viewpoint variations, one possible solution could

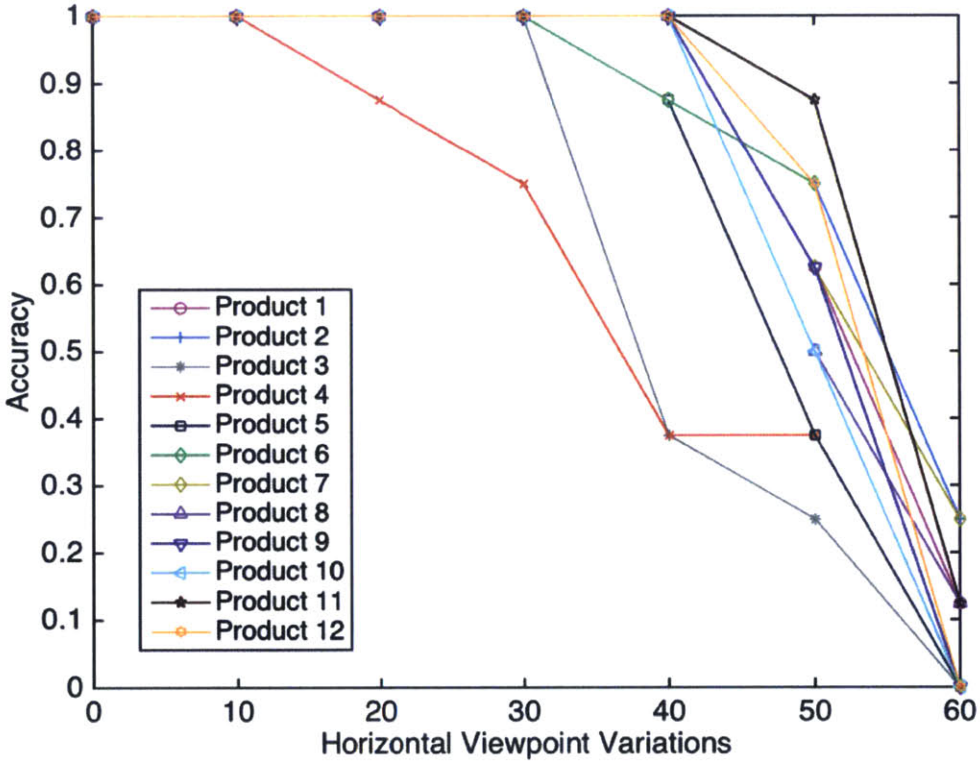


Figure 4-4: The recognition accuracy for different viewing angles

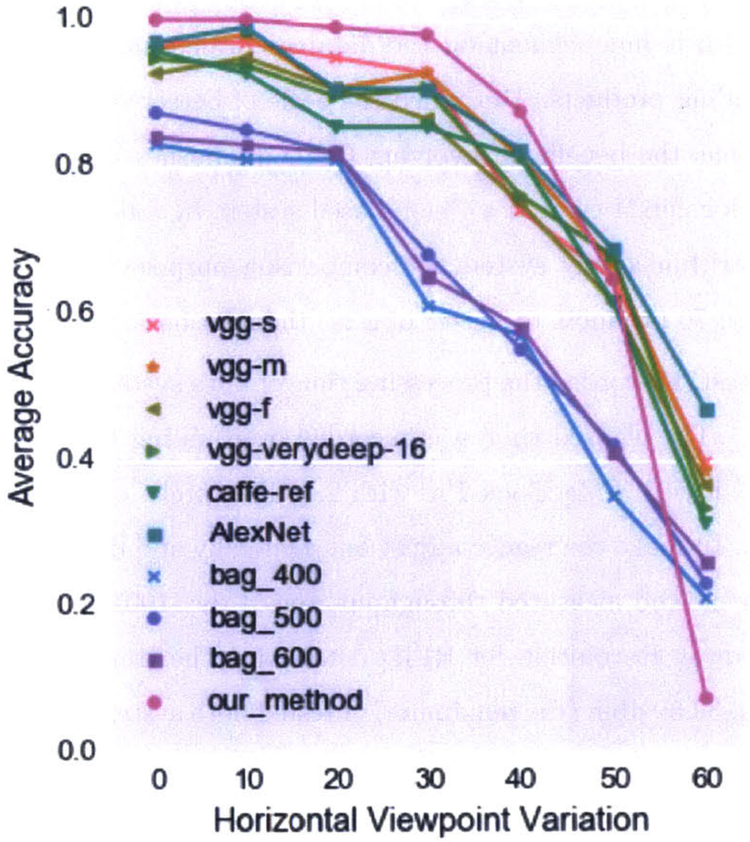


Figure 4-5: The average recognition accuracy for different models

be to implement my visual recognition scheme on multiple faces of each product, and to use other faces when one face suffers a severe viewpoint variation.

4.3 Experiment 3: The scale of surrounding detected tags matters

In the third experiment, I quantified the performance improvement of integrating RFID as a second data source for product recognition. Without RFID, a pure vision-based product recognition system has to verify the whole database to identify the viewed product. The computation cost increases as the database scale becomes larger. With RFID, my system only needs to verify the detected surrounding products since only surrounding products can potentially be the viewed product. The computation cost of my system is independent on the database scale, and only depends on the scale of surrounding products. But when the scale of detected products approaches the database scale, the benefit of involving RFID diminishes. To quantitatively describe this relationship, I created a vision-based system by following the same visual recognition algorithm as my system for comparison purpose. I also extended the database scale to 70 products to better obverse the relationship.

During the test, I recorded the processing time of both systems for recognizing the viewed product. The elapsed time was recorded by counting the CPU ticks on the server machine. I used a MacBook Pro with 2.6 GHz Intel Core i7 processor as the server machine. To make the reader adjust automatically and dynamically according to the environment and measured throughput, I used the '1001' LLRP mode offered by Impinj Speedway Revolution for RFID detection. The transmit power of RFID reader was set to 32.50 dbm (the maximum). I tested both systems on the 12 products that I used in previous experiments. For each product, I recorded the recognition time of both systems for different number of detected products. The computation time for the original visual recognition system is measured according to Equation 4.1, where T_i represents visual verification time for each candidate, and n is the total number of products in the database. Since a pure vision-based system needs to verify all the products in the database, its processing time should be nearly constant for each product being viewed.

$$T_{\mathrm{without\_RFID}} = \sum_{i=1}^{n} T_{\mathrm{i}} \tag{4.1}$$

$$T_{\mathrm{with\_RFID}} = T_{\mathrm{RFID\_detection}}(k) + \sum_{i=1}^{k} T_{\mathrm{i}} \tag{4.2}$$

The computation time for the RFID-enhanced recognition system is measured according to Equation 4.2, where k represents the number of detected surrounding products. $T_{\mathrm{RFID\_detection}}(k)$ represents the time that the RFID reader takes to detect the target product when k surrounding interfering tagged products exist. It changes

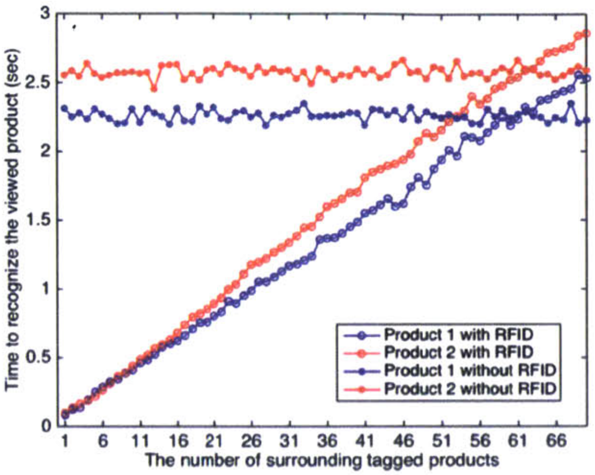


Figure 4-6: The processing time comparison between with and without RFID for product 1 and product 2

nonlinearly for different k due to tag collision problems. Figure 4-6 to Figure 4-11 show the processing time for different product sets.

The processing time of the vision-based recognition system is a summation of the elapsed time for verifying each product in the database, so we can see that it is independent on the scale of surrounding products and is nearly constant. On the other side, the processing time of the RFID-enhanced system basically consists of two parts: the time for RFID detection and the time for visual recognition. We can see that each product has a critical point that appears near the database scale. Before the critical point, the RFID-enhanced system spends less time recognizing the product being viewed. After that point, the RFID-enhanced system takes more time for recognition. The extra time is due to the time for RFID detection, which can be further increased by tag collision problems when the volume of surrounding tags becomes larger. More benefits of integrating RFID can be gained when less surrounding interfering RFID tags can be detected. One approach to achieve this goal is to match the detection area of the RFID reader to the field of view of the camera. For my prototype, I found that when setting the transmit power at 19.75 dbm, the gap region between these two areas is small, and the RFID detection range

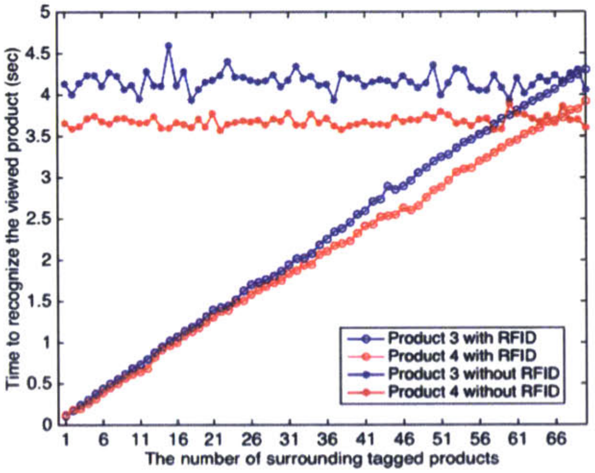


Figure 4-7: The processing time comparison between with and without RFID for product 3 and product 4

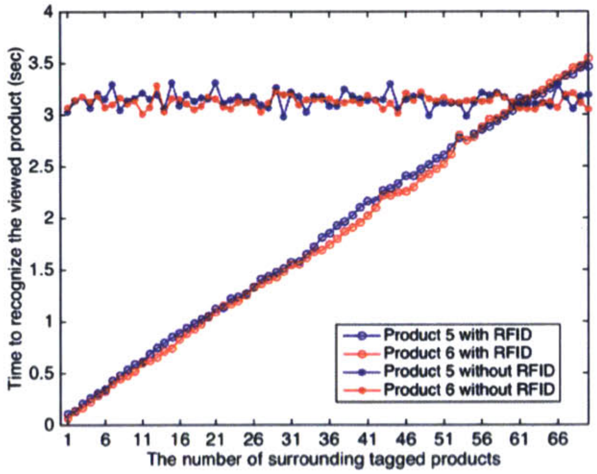


Figure 4-8: The processing time comparison between with and without RFID for product 5 and product 6

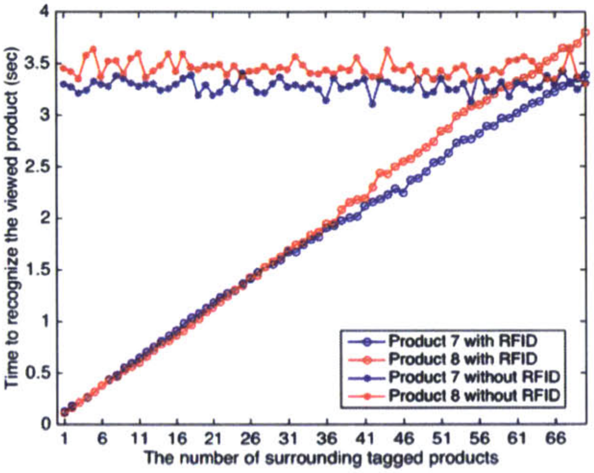


Figure 4-9: The processing time comparison between with and without RFID for product 7 and product 8

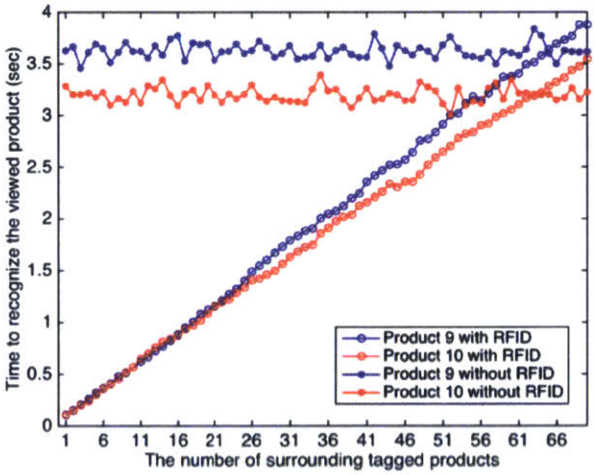


Figure 4-10: The processing time comparison between with and without RFID for product 9 and product 10

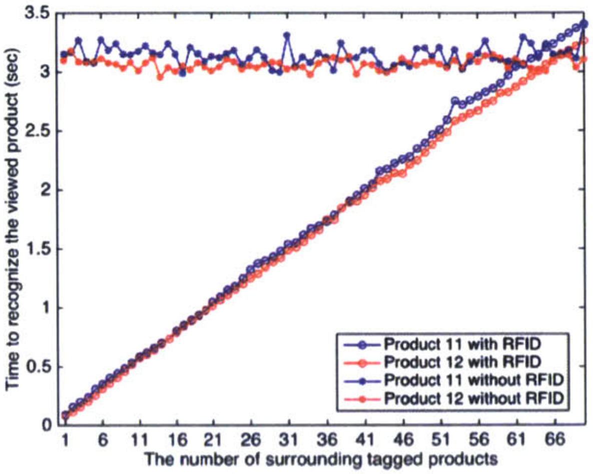


Figure 4-11: The processing time comparison between with and without RFID for product 11 and product 12

is about 1.5 meters in front of the system, which is also suitable for camera to capture details of the viewed product. Another thing worth noting is that the database in this experiment only contains 70 products, and the patterns in the plots could vary when the database includes a lot more products.

4.4 Summary

This chapter describes three experiments and their corresponding results. First, when no viewpoint variation is involved, my system outperforms many popular image classification models on the dataset containing visually similar products. Second, Our method achieved over 97.92% recognition accuracy for horizontal viewpoint variations of less than 30 degree. Both of the experiments are biased experiments, since I only tested on visually similar products. If I include more products with different visual appearance, other methods may perform better. Last, I quantitatively show the advantage of integrating RFID into our system. RFID helps increase computation efficiency dramatically when the scale of surrounding interfering tags is small.

Chapter 5

Conclusions and future work

5.1 Conclusions

In this thesis, I built a retail product recognition system aiming to improve customer in-store shopping experience. My system consists of two parts: the RFID detection and the visual recognition. The RFID detection is for increasing the computation efficiency of the system. By attaching an RFID tag to each product in the database, the RFID reader can detect surrounding registered products, and reduce the candidacy scope from the whole database to the detected products. To recognize the viewed product from the detected candidates, I implemented a two- phase scheme. In the first phase, I filter out visually different candidates, and only pass the candidates with similar visual appearance as the viewed product to the second phase. In the second phase, my system verifies the signature pattern of each remaining candidate to accurately identify the viewed product. Verifying the signature pattern makes my system robust to distinguish similar looking products.

I conducted a series of experiments to test the performance of my product recognition system. In the first experiment, I tested the recognition accuracy of my system under ideal working conditions. I then compared the result with other popular image classification models, including deep learning models and bag-of-visual-words models, on the same dataset under the same working conditions. My system outperformed all the other models, and achieved 100% recognition accuracy in this experiment. In the second experiment, I involved horizontal viewpoint variations. For the viewpoint variations of less than 30 degree, my system maintained over 97.92% recognition accuracy, and outperformed other models for the viewpoint variations of less than 40 degree. However, when the viewpoint variation becomes severe, the recognition accuracy of my system drops rapidly due to the difficulty of rectifying the geometric transformation completely. In the last experiment, I quantitatively described the advantage of integrating RFID into my system by recording the processing time of my system and a system without using RFID. This advantage is significant when the database scale is large and the volume of surrounding product is small.

5.2 Future work

In the future, I will go beyond RFID. Currently, I use RFID to capture surrounding information to reduce verification scope in the database. However, using RFID limits the application scope, since there are some situations that objects cannot be tagged, such as the products shown on websites, movies and magazines. These cases indicate that a pure vision-based recognition system has a wider scope of applications. We have discussed the computation efficiency problem of a vision-based fine-grained recognition system in previous chapters, but it is possible to resolve this problem by designing image classification models to obtain potential candidates, and then apply our fine-grained recognition scheme just among these candidates. Deep learning is a cutting-edge research topic and also a powerful tool for image classification tasks, so I plan to develop my model based on it. However, due to their complexity, well- performed deep learning models require a huge amount of labeled samples. So, designing well- performed "light-weight" deep learning models is a challenging task. Further, a deep learning model has fixed structure, and any structural change requires a retraining process. Training complex models is a very time-consuming process, and we do not hope this to happen frequently. However, for the retail industry, retailers may sell new products or stop selling old products frequently to maximize their sales revenues, which require the flexibility of reconstructing existing models. In current computer vision area, well-performed "flexible" deep learning models have not been developed yet. In my future work, I plan to design a hierarchical structure to optimize the output layers of conventional deep learning models. The hierarchical structure is supposed to store the learned "deep features" at different levels. When changing output classes, we only need to retrain a minor part of the model instead of the whole structure to make a trained model flexible enough for retail applications.

Bibliography

[1] https://en.wikipedia.org/wiki/Bag_of_words_model_in_computer_vision/. Accessed April 15, 2016.

[2] Cmu andyvision robot is in your store, doing your inventory. http://spectrum.ieee.org/automaton/robotics/industrial-robots/cmu-andyvision-inventory-robot/. Accessed April 15, 2016.

[3] Flow powered by amazon. http://flow.a9.com/. Accessed April 15, 2016.

[4] Mnist dataset. http://yann.lecun.com/exdb/ mnist/. Accessed April 15, 2016.

[5] A robot takes stock. https://www.technologyreview.com/s/428374/a-robot-takes-stock/. Accessed April 15, 2016.

[6] Robots in retail. http://www.cmu.edu/homepage/computing/2012/summer/robots-in-retail.shtml/. Accessed April 15, 2016.

[7] Sllurp library. https://github.com/ransford/sllurp. Accessed April 15, 2016.

[8] Strengthening smf competitive advantage through rfid implementation. http://www.rfid-f2f.eu/hardware.asp. Accessed April 15, 2016.

[9] Supermarket facts. http://www.fmi.org/research- resources/supermarket-facts/. Accessed April 15, 2016.

[10] Ufldl tutorial. http://ufldl.stanford.edu/tutorial/. Accessed April 15, 2016.

[11] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.

[12] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[13] Jeffrey P Bigham, Chandrika Jayan, Andrew Miller, Brandyn White, and Tom Yeh. Vizwiz:: Locateit-enabling blind people to locate objects in their environment. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 65–72. IEEE, 2010.

[14] Mustapha Boukraa and Shigeru Ando. Tag-based vision: assisting 3d scene analysis with radio-frequency tags. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–269. IEEE, 2002.

[15] Hilton Bristow and Simon Lucey. Why do linear svms trained on hog features perform so well? *arXiv preprint arXiv:1406.2419*, 2014.

[16] Heesung Chae and Kyuseo Han. Combination of rfid and vision for mobile robot localization. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2005. Proceedings of the 2005 International Conference on*, pages 75–80. IEEE, 2005.

[17] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[18] David M Chen and Bernd Girod. Memory-efficient image databases for mobile visual search. *MultiMedia, IEEE*, 21(1):14–23, 2014.

[19] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[20] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

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

[22] Guillermo Enriquez, Sunhong Park, and Shuji Hashimoto. Wireless sensor network and rfid fusion approach for mobile robot navigation. *ISRN Sensor Networks*, 2013, 2013.

[23] Li Fei-Fei and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 524–531. IEEE, 2005.

[24] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[25] Samuel Fosso Wamba and Harold Boeck. Enhancing information flow in a retail supply chain using rfid and the epc network: A proof-of-concept approach. 2008.

[26] Gary M Gaukler, Ralf W Seifert, and Warren H Hausman. Item-level rfid in the retail supply chain. *Production and Operations Management*, 16(1):65–76, 2007.

[27] Marian George and Christian Floerkemeier. Recognizing products: A per-exemplar multi-label image classification approach. In *Computer Vision–ECCV 2014*, pages 440–455. Springer, 2014.

[28] Marian George, Dejan Mircic, Gabor Soros, Christian Floerkemeier, and Friedemann Mattern. Fine-grained product class recognition for assisted shopping. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 154–162, 2015.

[29] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[30] Hidekata Hontani. A visual tracking system using an rfid tag. In *Electronic Imaging 2005*, pages 165–174. International Society for Optics and Photonics, 2005.

[31] Rabia Jafri, Syed Abid Ali, and Hamid R Arabnia. Computer vision-based object recognition for the visually impaired using visual tags. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013.

[32] Rabia Jafri, Syed Abid Ali, and Hamid R Arabnia. Computer vision-based object recognition for the visually impaired using visual tags. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013.

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

[34] Mikko Kärkkäinen. Increasing efficiency in the supply chain for short shelf life goods using rfid tagging. *International Journal of Retail & Distribution Management*, 31(10):529–536, 2003.

[35] Jin-Young Kim, Chang-Jun Im, Sang-Won Lee, and Ho-Gil Lee. Object recognition using smart tags and stereo vision system on pantilt mechanism. *Proceedings of ICCAS2005*, 2005.

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

[37] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[38] Xiaofan Lin, Burak Gokturk, Baris Sumengen, and Diem Vu. Visual search engine for product images. In *Electronic Imaging 2008*, pages 68200M–68200M. International Society for Optics and Photonics, 2008.

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

[40] Yacine Rekik, Evren Sahin, and Yves Dallery. Analysis of the impact of the rfid technology on reducing product misplacement errors at retail stores. *International Journal of Production Economics*, 112(1):264–278, 2008.

[41] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

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

[43] Gül Varol and Rıdvan S Kuzu. Toward retail product recognition on grocery shelves. In *Sixth International Conference on Graphic and Image Processing (ICGIP 2014)*, pages 944309–944309. International Society for Optics and Photonics, 2015.

[44] Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*, pages 689–692. ACM, 2015.

[45] Emily Wang. Project eye-helper an assistive technology for blind grocery shoppers. 2015.

[46] Tess Winlock, Eric Christiansen, and Serge Belongie. Toward real-time grocery detection for the visually impaired. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 49–56. IEEE, 2010.

[47] Jun Yang, Yu-Gang Jiang, Alexander G Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206. ACM, 2007.

[48] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.

[49] Yuhang Zhang, Lei Wang, Richard Hartley, and Hongdong Li. Handling significant scale difference for object retrieval in a supermarket. In *Digital Image Computing: Techniques and Applications, 2009. DICTA'09.*, pages 468–475. IEEE, 2009.