A Study on Composite Data Mining Methods for Linking Real-World Information with Web Resources

Chenyi Liao

A Study on Composite Data Mining Methods for Linking Real-World Information with Web Resources

Chenyi Liao

Department of Computational Science and Engineering
Graduate School of Engineering

Nagoya University

A thesis submitted for the degree of

Doctor of Engineering

2019

Abstract

The real-world data are data collected by sensors from the real-world. Nowadays, the rapid growth of the Internet of Things (IoT) and the ubiquitous computing makes the large-scale real-world data (namely the sensor data) growth. Sensor data can be self-organizing data via spatial-temporal information. Smartphone as a prominent data collection device can take photos, collect GPS data in built-in camera and GPS sensor. It can produce a photo with shooting location and time. A panoramic camera combined with Light Detection and Ranging (LiDAR) can collect 3D point cloud data. Through calibration, point cloud data with street images can be automatically corresponded. The Web data are typically represented in HTML documents including Web text and Web images on the Internet. Search engines were invented to organize these mass volume of Web data. Users can search relevant Web images and Web text in keywords. A limitation of current search engines is that users must have enough prior knowledge to provide the appropriate keywords. If sensor data can be used as the seed for search engine, which makes it possible to search Web data even if the user is in an unfamiliar environment. Therefore, the challenge to link sensor data with relevant Web data is a crucial task.

In this thesis, the discussion starts from the point whether it is possible to mine relevant Web data through real-world data. Two types of real-world data are used to link relevant store Web data. The store information is selected as the research object because the store information typically includes both the real-world data (location, photos, etc.) and Web data (store's Web page etc.). Firstly, GPS location data is used to extract nearby store event records from Web pages.

Secondly, street images taken by users are used to identify store signages and link them with corresponding Web pages.

Event data, such as coupon, exhibition, or party notifications hidden in millions of Web pages, may help users spend their free time more efficiently. However, it is very time-consuming for a user to find those information from thousands of Web pages manually. An automatic extraction method of nearby event data from Web pages according to users' location information is proposed. This task involves two academic issues. First, the event data provided to users should be structured data, which is extracted from stores' Web pages. Since the Web page is massed as semi-structured data, a Web page segmentation algorithm that converts the HTML documents into processable structured data is proposed. This algorithm ensures each event record is an atomic data item as in a relational database. Second, not all data extracted from Web pages are event data. So, a Web event extraction algorithm that extracts only event records from Web records and filters out other irrelevant records is proposed. An evaluation experiment of event extraction using proposed methods achieves a F1-score of 83.96%, which means that 83.96% of event data (both of the precision and the recall) can be correctly extracted from Web data.

For linking a street image with a Web image, an image-matching based identification of store signage using Web-crawled information is proposed. In this study, automatic matching of street images with relevant Web pages to enable the identification of store signage in street images is addressed. This task involves two issues. First, identification methods for signage usually involve image matching, which attempts to match query images to other similar viewings using pre-labeled copies from a target dataset. This target dataset is automatically generated from Web-crawled information at low cost and as clean as possible. This Web image dataset construction is introduced, that self-generates a high-quality dataset through automated Web mining, including data filtering and pruning strategies, which effectively reduce the identification error caused by noise, imbalance, and insufficient data. Second, a Hybrid Image Matching (HIM) method for the match-

ing of street images with Web-crawled images is introduced. This can effectively avoid the impact of target dataset noise and imbalance from Web mining. An experimental result achieves 91% accuracy in a real-life application, which confirms its effectiveness.

All methods proposed in this thesis present a new concept of linking real-world information with Web resources. The experimental results verify the feasibility of this concept. The application implemented by proposed methods and source code of proposed methods are also provided in this thesis. It should be a contribution to various potential applications.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this Chapter, a general introduction of this thesis is presented. The problem establishment of linking real-world data and Web data collection is introduced in Section 1.1, including the background (in Section 1.1.1), the motivation and the necessity of linking real-world data and Web data (in Section 1.1.2), and the structure of this thesis (in Section 1.1.3). The backgrounds, the challenges and proposed methods related to the two tasks of this thesis, which are Event Data Extraction and Signage Matching, are presented in Sections 1.2 and 1.3, respectively.

1.1 Overview of This Thesis

1.1.1 Background

The Web data are data that comes from large or diverse number of sources on the World Wide Web (or the Web for short). Because Web data allow sharing of information through HTTP protocol [1], Web pages described in HTML [2] are the main carrier of Web data. As shown in Figure 1.1, the Web data typically include Web elements, Web texts, and Web images. Web elements are mainly represented by HTML elements, such as hyperlinks and other HTML elements in the HTML document. The Web texts are texts visible to users in an HTML document. The Web images include photos and computer graphics (CG) embedded in the Web page.

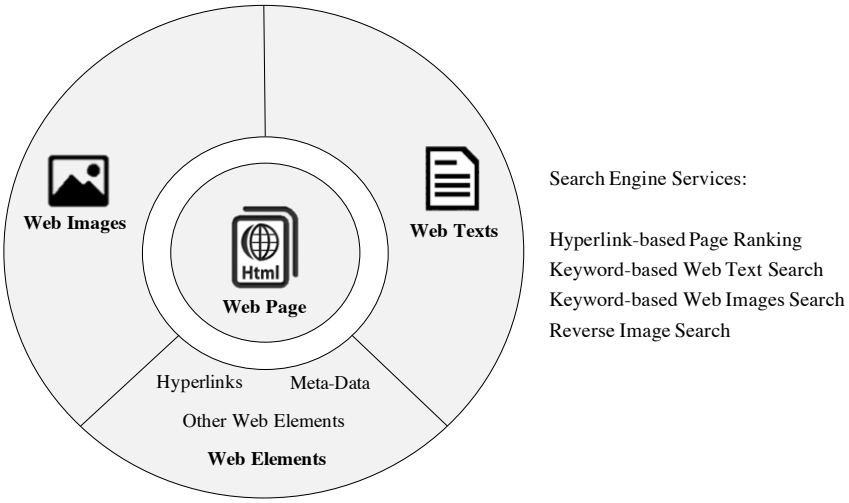


Figure 1.1: Constitution of Web data.

With the development of the World Wide Web, the volume of data resources on the Internet has shown an exponential growth. Up to 2016, a survey explains there are over 130 trillion (10^{12}) individual Web documents indexed by Google [3]. Search engines were invented to organize this mass volume of Web data. As shown in Figure 1.1, a search engine can index a Web page exploiting hyperlink. The Page Rank mechanism [4] is a kind of Web structure mining to discover knowledge from hyperlinks [5]. It measures the human interest and attention to a Web page according to the number of pages that are linked. Search engines typically provide a keyword search function, which retrieves Web text according to user's keywords. In addition to classic text retrieval, a search engine can also handle multimedia data, i.e. it can match Web images with relevant Web texts, so that relevant Web images can be searched by text queries. The reverse image search [6] is possible to retrieve relevant Web image by input of an image. In recent years, with the help of the development of Convolutional Neural Networks (CNN) [7, 8], the accuracy of reverse image search has greatly improved. Search engines can also generate textual descriptions for Web images directly [9, 10]. This enhances the ability of Web image search even without Web text support. A search engine uses these techniques to organize Web data and establish the associations between Web data items.

1.1.2 Research Motivation

Information on commercial stores is targeted in this thesis. Store information is one of typical instances that includes both real-world information (location, image, etc.) and Web resources (Web page, etc.). In a real-life scenario, a search engine can be used to find most store Websites in keyword search, thus obtaining useful information (address, business hours, product information, etc.) from Web pages. However, all the above-mentioned search engine techniques must be fed with a keyword query [11] to narrow down the target data range from a mass of Web data. In this way, a search engine returns a search result. The limitation of this is that users must have enough prior knowledge to provide an appropriate seed. This makes searching store Web data in an unfamiliar environment difficult. For example, let us consider a typical situation of searching information on stores nearby a train station. A user should firstly search for a list of nearby stores using the area name as a keyword. Then he/she should check each Website of stores for their detailed information. However, if a passerby who comes to a city for the first time is waiting for the next train that departs two hours later, how can

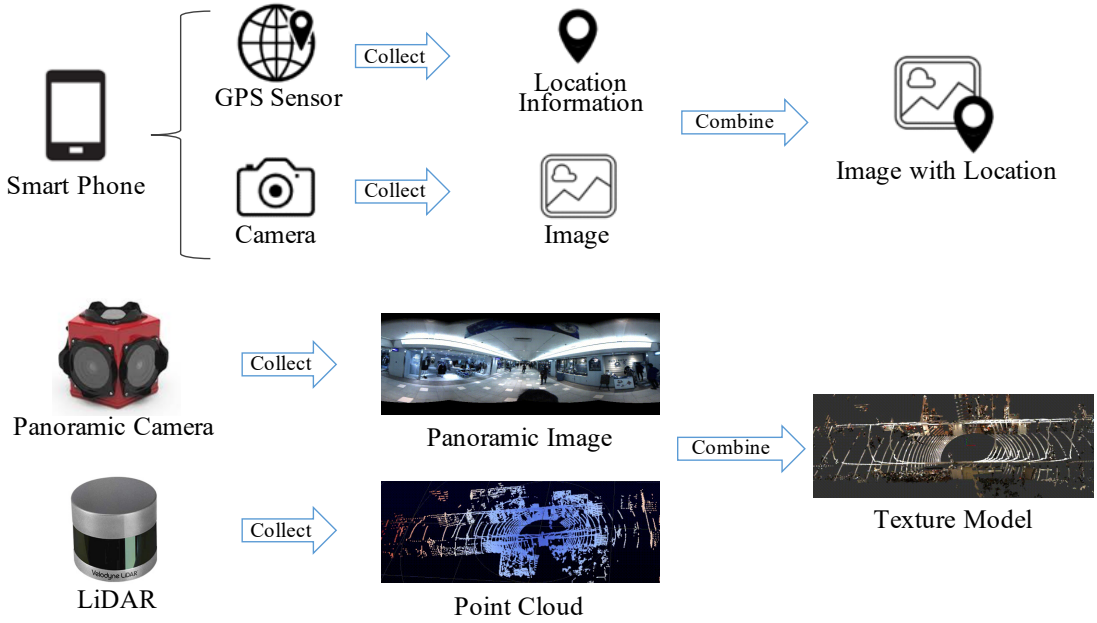


Figure 1.2: Sensors collect and organize real-world data automatically.

he/she find a nearby event or store information in an unfamiliar environment to spend the free time more efficiently? In the absence of a priori knowledge, using a traditional search engine is difficult to do so. Therefore, a concept that uses sensor data to provide the seed for the Web data search is considered. Before explaining this concept in detail, the real-world data collected by sensors are analyzed firstly.

Similar to Web data, real-world data collected by sensors can also be organized automatically. The rapid growths of the Internet of Things (IoT) and the ubiquitous computing in the past decade have facilitated the large-scale real-world data collection. Miniaturized devices make collecting data from the real-world at low cost. Smartphone as a prominent data collection device can take photos and collect GPS data. As shown in Figure 1.2, a smartphone can take an image by a built-in camera with the location data collected by a GPS sensor. A panoramic camera combined with a Light Detection and Ranging (LiDAR) device which can collect 3D point cloud data [12], through calibration [13], can automatically yield point cloud data corresponding to street images. Therefore, the same as Web data organized by search engines, it is also possible to organize real-world data by the above-mentioned matching techniques. As a result, using one kind of real-world data can retrieve on other kinds of real-world data. For instance, Hays et al. [14] proposed estimating geographic information from an urban image. This approach links street image with location information. It retrieves similar GPS-tagged images and assign relevant location information to test dataset of urban images. Qian et al. [15] used geo-tagged images to generate visual descriptors, which can be used to estimate the location of images without geo-tags.

1.1.3 Thesis Structure

In this thesis, the discussion starts from the point whether it is possible to mine relevant Web data through real-world data. The store information is focused as the research target, because it typically includes real-world data (location, photos, etc.) and Web data (Web page, etc.). Figure 1.3 shows the composition of this thesis. Two types of real-world data are used to link relevant Web data. Firstly,

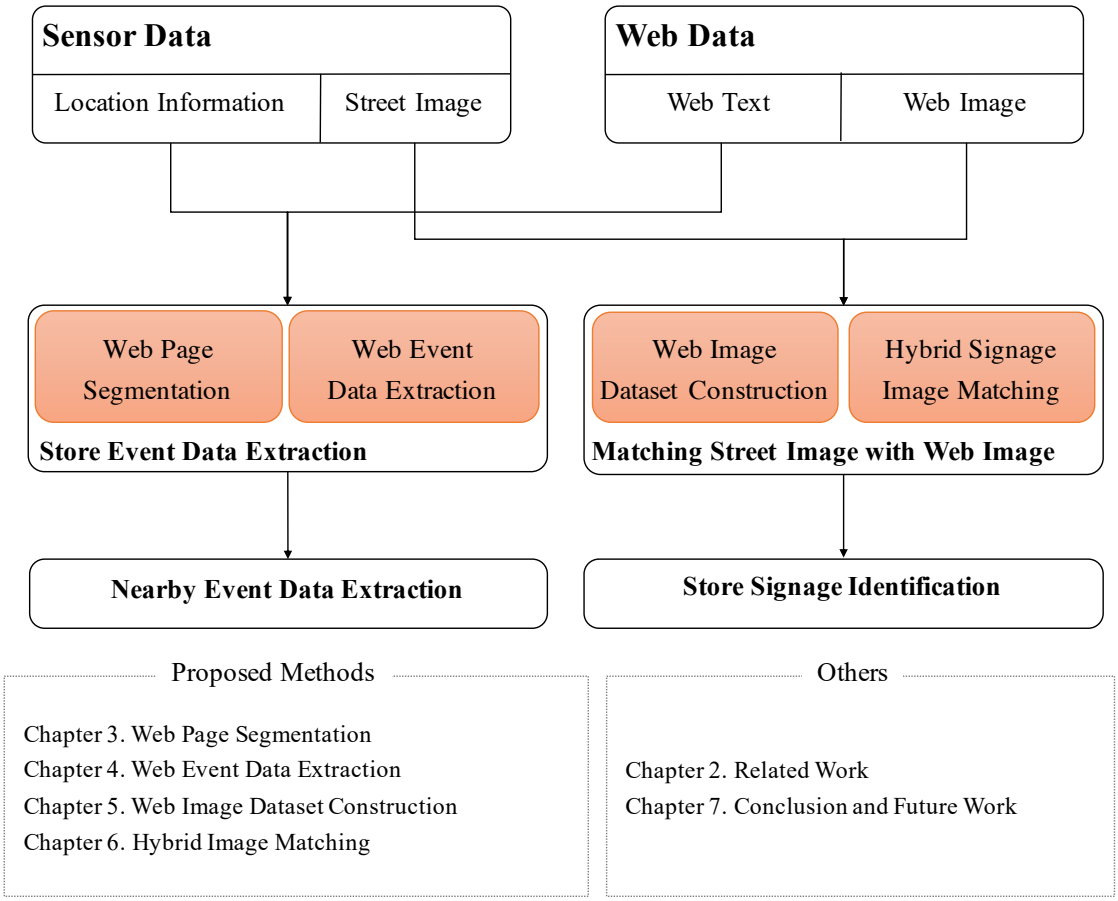


Figure 1.3: Outline of this thesis.

GPS location data are used to extract nearby store event records from Web pages. Secondly, street images are used to identify store signages to link them with corresponding Web pages.

The store event data extraction involves two academic issues. First, the event records should be structured data, which is extracted from stores Web pages. Since the Web page is represented as semi-structured data, a Web page segmentation algorithm that converts the HTML documents into processable structured data is proposed. This algorithm ensures each event record is an atomic data item as in a relational database. The source code of the Web page segmentation algorithm is published, as introduced in Chapter 3. Second, not all records extracted from Web pages are event records. So, a Web event extraction algorithm that extracts only event records from Web records and filters out other irrelevant records is proposed. This algorithm is introduced in Chapter 4. The Web page segmentation and Web event data extraction are combined to implement an application named Event.Locky, which is introduced in Section 4.4. A more detailed introduction of these algorithms is introduced in section 1.2.

For linking a street image with a Web image, an image-matching based identification method of store signage using Web-crawled information [17] is proposed, in which automatic matching of street images with relevant Web resources to enable the identification of store signage in street images is addressed. This task involves two issues. First, identification methods for signage usually involve image matching, which attempts to match query images to other similar viewings using pre-labeled copies from a target dataset. This target dataset is automatically generated from Web-crawled information as clean as possible and at low cost. In Chapter 5, This Web image dataset construction is introduced, that self-generates a high-quality dataset through automated Web mining, including data filtering and pruning strategies, which effectively reduce the identification error caused by noise, imbalance, and insufficient data. In Chapter 6, a Hybrid Image Matching (HIM) method for the matching of Web-crawled images and street images is introduced. This can effectively avoid the impact of target dataset noise and imbalance

from Web mining. Even without applying Optical Character Recognition (OCR), it can handle an arbitrary signage design whether or not the signage is illustrated in a special font or a logo. Because there is no specialized training involved, the same process should also work for any other location without manual adjustment. The source code of HIM is published as an open-source software introduced in Section 6.4. A more detailed introduction of these algorithms is introduced in the Section 1.3.

In Chapter 2, related literature, which aim at similar tasks, are introduced. These related literature is analyzed in detail and compared with the works proposed in this thesis. In Chapter 7, the conclusion of this thesis is presented. The range of the application and prospects of the proposed methods are discussed. In the end, the shortcomings and motivating future work of this thesis are discussed.

1.2 Store Event Data Extraction from Web Pages

Event data, such as coupon, happy hour promotion, special menu, exhibition or party notifications, are hidden in millions of Web pages on the Internet. Usually, store owners update their event data on their top pages as a list of event titles, dates, and links to detailed information. Although event data may help users spend their free time more efficiently, it is very time-consuming for a user to find them from thousands of Web pages. In addition, conventional search engines demand users to have a priori knowledge on the location to input the appropriate keywords as query. For example, a user who wants to browse the event data of a nearby store must input the store name as a query. This mechanism is unfriendly to a foreign tourist or a passerby. So, automatic extraction of nearby event data from Web resources according to a users location information and push them to his/her mobile device should be considered. The innovative point of this is local event data retrieval that provides spatial-temporal event data to a user on the spot.

Existing event Web resources are firstly investigated. There are event search Application Program Interface (API) services [19], which provide structured event data with their RESTful API. Event search APIs (e.g., ATND [20], Peatix [21], etc.) collect spatial-temporal event data through user uploads. However, these services provide event crowd-sourcing platforms that are focused on individual events (almost about seminars or lectures), but they do not provide certain popular events (such as sales promotions, exhibitions, or happy hours), which are more attractive to users.

Other than event search APIs, users also voluntarily share event data that they feel interesting through social network services (SNSs). For example, a number of approaches address event data extraction on Twitter [1] [22], [23], [24]. Extraction of structured event data from a single Website (Twitter, in these cases) can depend on a role-based crawler. This greatly reduces the difficulty of Web data collection. However, most event data on SNSs are not official. Although some stores also tweet event data on their official blogs, most of them tend to share event data on

[1] Twitter: https://twitter.com/

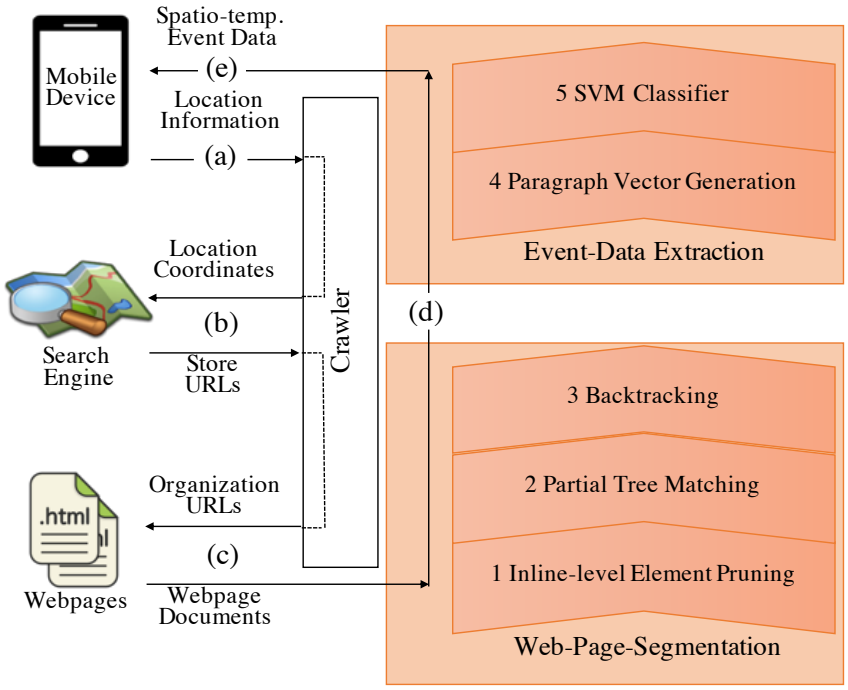


Figure 1.4: The pipeline of nearby event data extraction from Web pages.

their Websites. Official Web pages contain accurate event data of stores. Ichien et al. [25] demonstrated it is possible to discover data from stores Web pages. This thesis explores a specific concept: Can Web mining techniques be applied for event data extraction? Web mining [26] is a sub-concept of data mining, which focuses on the extraction of useful contents from the Web.

Figure 1.4 shows the pipeline of the proposed event data extraction scheme. Ahmed et al. [27] introduced a geographical topic discovery method via smartphone GPS locations. Inspired by this approach, a scheme for nearby event extraction is proposed in this thesis. First, (a) a users device inputs location data (coordinates) collected by an built-in GPS sensor or by indication to the server. Second, (b) Website URLs of nearby stores are retrived through a search engine according to the location data. Third, (c) a crawler in the server downloads Web page documents. Forth, (d) an Web page segmentation algorithm then converts the HTML documents into processable structured data, and an Web event record extraction algorithm extracts event records. Web page segmentation is discussed

in Chapter 3, which is divided into three components: Inline-level Element Pruning, Partial Tree Matching, and Backtracking. Meanwhile, event data extraction is discussed in Chapter 4, which is discussed into the paragraph vector generation method and the textual classifier. Finally, (e) extracted event records including locations, times, images, and contents are pushed to the user's client. A client application in the users device displays these event records as spatial-temporal data.

Gupta et al. [28] mentioned about the proposed method in this thesis. There are 'two tasks for extracting meaning from a Web page. The first task is segmentation of the Web page into blocks where a block refers to an area of a Web page. Since the page is written in HTML, this task involves HTML code being organized into different blocks. The second task is event data identification. After segmentation, these blocks consist of relevant as well as irrelevant data. To eliminate the irrelevant data, Web page block segments are parsed, and the event data items are identified.' In summary, extraction of event data from Web pages is constructed by solving two core issues as follows.

First, a semi-structured HTML document cannot be processed directly. Arasu et al. [29] proposed the concept of extracting structured data from Web pages by a template-based method. They mentioned that the structured data must be stored and processed in a relational database and considered Web page generation through pre-defined templates in the backend. The Web page segmentation algorithm proposed in this thesis converts semi-structured HTML document into structured data including segments and records, which correspond to data tables and rows in the table in a relational database, respectively. It can be seen as a reverse engineering of Web page generation. The main process of Web page segmentation is a partial tree matching, which reverses the processing of HTML document generation. Through matching similar HTML structures in an HTML document, it can extract data records from a document. In addition, two algorithms: inline-level element pruning and back-tracking, which can also extract independent Web records that the partial tree matching cannot, are proposed. As

a result of an experiment, the proposed algorithm could segment 98.76% of event records from 96 stores in an actual shopping mall. Detailed discussion of Web page segmentation is provided in Chapter 3.

Next, atomic Web records are extracted from a Web page including massive amount of non-event data. For event data extraction from Web pages, a feasible text classifier is proposed for this task. It is more suitable for classifying short sentences such as Web event records. F1-score [30] [2] is used for evaluation of the proposed event records classification. As a result of evaluation experiment, it achieves a higher F1-score of 91.61% of event record classification than others. Detailed discussions of event data extraction are provided in Chapter 4.

[2] F1-score is an evaluation index for binary classification. It considers both the precision and the recall as $F_1 = 2 \times \frac{precision+recall}{precision \times recall}$, where *precision* is the number of correct positive results (here, the number of correctly classified event records) divided by the number of all positive results (here, the number of all classified event records) returned by the classifier, and *recall* is the number of correct positive results (here, the number of correctly classified event records) divided by the number of all relevant samples (here, the number of all event records)

1.3 Signage Identification using Web-crawled Information

Identification of signage in street images and linking them to relevant Web resources can provide friendly experience to users. Even if users do not know anything about the area, they can also access the Website of the store for detailed information by uploading a street image. Therefore, automatic matching of street images with relevant Web resources is addressed in this thesis to make the identification of store signage in street images possible. It is a crucial task with various potential applications, as it can provide an interactive user experience, such as a virtual shopping environment [32].

The EXIF information with location data embedded in a street image metadata recorded by a built-in GPS sensor may also be useful for store matching [31]. However, its location is not accurate enough. For example, indoor, where the GPS waves are unreachable, the device can only get a rough location such as the building location. In this case, an image-based signage identification is necessary. Although the location information of a street image could not be used for

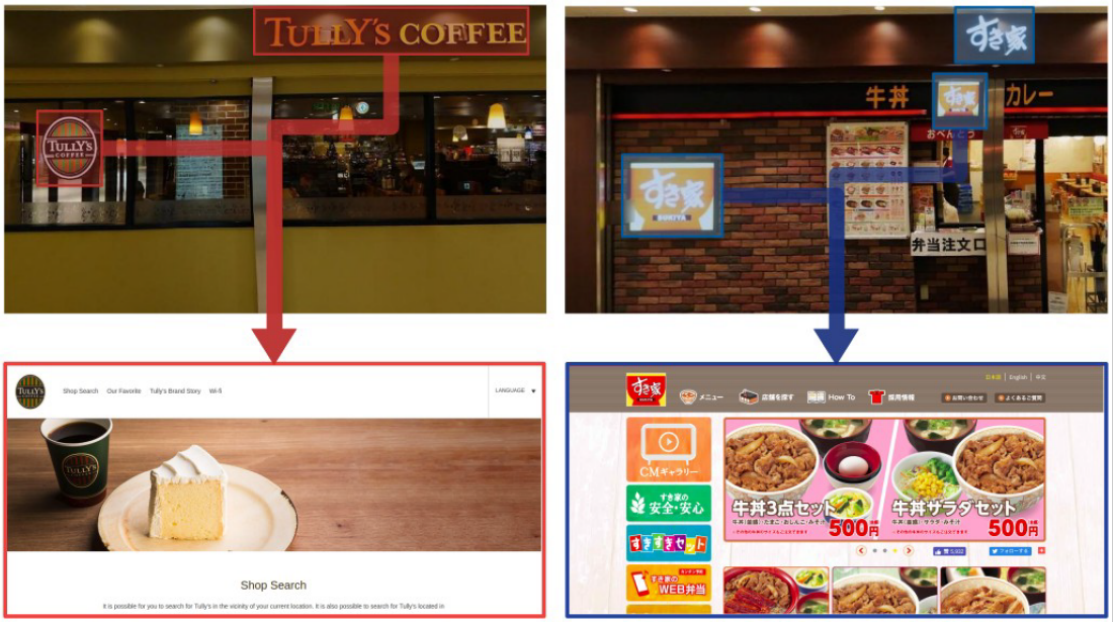


Figure 1.5: Example of signage identification in street images. Signage is identified and linked to store Websites. The user can access the Websites to obtain information or order products from the store.

exact store matching, it can narrow down the range of matching candidates. For example, using street image location, a list of stores in the building such as from the official Website of a shopping mall can be easily obtained. Once the list of nearby stores is obtained, a relevant target matching dataset from Web data can be generated. The signage patches, such as shown in Figure 1.5, which is extracted from street images, are automatically identified and linked to their store Websites.

This identification process must be fully automated. However, real-world data are not rigid. Spatial data could change over time [33]. For example, in the case of constructing a virtual shopping system, the store's profiles in a shopping mall may change every quarter. The most straightforward means for establishing links is to annotate the relevant metadata manually, in which a technician must manually re-adjust and re-collect annotations every time when a tenancy or store changes. Such manual process introduces an ongoing maintenance issue, which may add significant cost to the otherwise relatively cheap process of collecting street images by cameras, which can be done reliably through automatic procedures. Therefore, the challenge of this task is to match street images with relevant Web data in an automatic procedure and to do so in a way that is as cost-efficient and accurate as possible.

There exist some automated methods that aim to resolve this issue. As discussion in Section 1.1.1, Web resource contains both textual data and image data. Matching of Web resources can be classified into the following two categories: textual-matching based methods, and image-matching based methods. Textual-matching based methods depend on an Optical Character Recognition (OCR) process, which focuses on recognizing text characters from the signage and compares with a local candidate store lexicon. It converts the image processing into a textual retrieval task. This method is not suited for cases involving characters with special font types, non-characters, or characters from a language group different from what the OCR engine specializes in. Meanwhile, the image-matching based method is dependent on specialized target datasets, which contain a group of pre-collected and pre-annotated images that it will then attempt to match the

query image with. For signage identification, the images in the dataset need to be labeled with their associated store IDs. Manual dataset establishment such as fingerprinting [34, 35] can create well balanced and accurate target datasets, but they must be created and kept up to date manually. Deep learning methods [7, 36] have achieved human-level accuracy recently but require millions of high-quality training data. Web-mining [37] can be used to assemble the target dataset from Web resources automatically, but noise and the lack of sufficiently balanced data affect the accuracy of signage identification. A more detailed analysis of the limitations facing existing methods can be found in Chapter 2.

These limitations are addressed in this thesis. Since the Web mining approach is considered as the most cost-efficient approach. An automatic store signage identification process is proposed, which requires only a list of store names and their corresponding Web addresses as the input. The list is easily obtained from official Websites of an experimental shopping mall. The street images from the same shopping mall are input without labeled signage, which is to be identified. In Chapter 5, an automatic Web-image dataset collection method is proposed, which crawls images related to stores in the shopping mall based on the input URL and Web-search images from the input store names. In this stage, the naive crawling process yields a massive amount of unrelated information, such as irrelevant images from an image search engine or non-logo images from a store's Website. Several data cleansing methods are introduced to resolve these problems: A two-step method first filters relevant facade images from Web-search images using VGG16 fine-tuning [7], and then crops the desired signage patches using YOLO [38, 39]. Meanwhile, a structure mining method and a statistics-based method extract Webpage logos from Websites. These two methods can effectively reduce the amount of irrelevant information obtained from data crawling.

In Chapter 6, a Hybrid Image Matching (HIM) method is proposed that combines the deep learning approach with the feature point matching for signage identification. A pruning algorithm is proposed to match each signage patch from the query dataset to their candidate matching datasets. As each dataset is only a small

subset of the total number of images obtained from the Web mining results, this effectively reduces the processing time required during the feature point matching process [40, 41], which is based on RANSAC [42]. As a result, the query signage in street images is output with identified store names and Web addresses. Finally, the proposed signage identification method is evaluated in a real-life scenario.

1.4 Summary

In this thesis, mining relevant Web data through sensor data is discussed in two tasks. Firstly, location information collected from users' mobile devices is used to retrieve nearby stores and their Web pages. Then the event data are extracted from stores' Web pages and pushed to users' mobile devices [16, 17]. Secondly, the Web search images and Web page images are used to generate a target matching database at low cost. A Hybrid Image Matching method, which implements the identification of store signage in street images [18], is proposed. Aiming at the two tasks, four methods are proposed: Web page segmentation that converts a semi-structured HTML data into processable structured data, Web event data extraction that filter out non-event data from structured Web data, Web image dataset construction that automatically generates target matching database in Web mining, and Hybrid Image Matching that can match the street images with Web images. The main contributions of this thesis are listed in Table 1.1.

Table 1.1: Main contributions of this thesis.

Noun	**Description**
Web Page Segmentation	A partial tree matching based algorithm is proposed to segment a Web page into segments and records analogous to tables and rows in a relational database. The method can convert a semi-structured HTML document into structured data, which can be reused, processed and published. By a back-tracking mechanism, this algorithm cannot only extract continuous records, but also extract isolated records in a Web page.
Web Event Data Extraction	Extracting event records from Web is considered as a text classification task. A sample weighted classifier is proposed, which achieves better results in the classification of short sentences such as Web records than others. It successfully extracts store events from Web pages at a high accuracy of F1-score 91.61%.
Web Image Dataset Construction	Web-mining is utilized to automatically generate target Web image datasets for signage matching at a low cost. Because most Web-crawled images contain massive amount of irrelevant information, and because of the sheer number of possible combinations between Web-crawled images and street images, a series of data pruning and filtering methods for data cleansing is proposed. The proposed method can reduce 83.46% and 98.86% of irrelevant images from 55,783 Web-search images and 18,381 Web page images, respectively.
Hybrid Image Matching (HIM)	Hybrid Image Matching (HIM) method is proposed, which is a pure image-matching based method of store signage identification without involving OCR processes. The method combining the deep learning approach with the feature point matching can effectively cope with noise, unbalanced data from Web mining generated datasets and achieves 91% accuracy in a real-life application. This method overcomes the limitations that textual matching methods have regarding special font types, non-characters, and multilingual texts in the signage identification tasks. Since the proposed method does not depend on a specialized training process, it is designed to be deployed to other various locations without any manually annotated training data.

Chapter 2

Related Work

2.1 Extraction of Event Data from Web Pages

Foley et al. [46] proposed an event content extraction method from Web pages. They defined the schema of an event record with its name, datetime, and location. Their method extracts an event list from a Web page based on probability score on which Web page fields are filled in with event data. They proposed the principles of extracting Web page records from large areas to small areas and extracting semantic Web data by textual classification.

Norvag et al. [29] proposed a news item extraction method from Web pages. They introduced key-points for news item extraction. Web pages on news are dynamic and changes continuously and the news items are short. These key-points also apply to event data extraction. It reminds us that event extraction should be performed in real-time and the event text in a Web page is generally composed of a short paragraph.

Magdy et al. [48] gave an advice about query speed evaluation. The performance of a Web mining system can be evaluated in number of processable data items per second. They also mentioned the use of spatial-temporal data can narrow-down target data for decreasing query processing time. These ideas inspired the works presented in this thesis.

Win [49] gave a standard scheme for this task including Web page segmenta-

tion and informative content extraction. The Web page segmentation splits a Web page document into blocks or regions, which includes Web data that are different from other regional topics. Informative content extraction is the process of determining the parts of a Web page which contain the main textual content of the document. These two steps correspond to the Web page segmentation and event data extraction in the task discussed in this thesis, respectively. In this section, related literature is introduced from these two aspects.

2.1.1 Web Page Segmentation

An Web page is composed of semi-structured data, which cannot be directly processed. Backend records are transformed into such data following certain rules. Intuitively, a region in a Web page including one or more elements wraps a data record from a database. A Web page segmentation algorithm generates wrappers that identify regions of data records and extract data records from Web pages [50].

Some approaches cover the task of Web page segmentation based on manual, tag, page-layout, vision, and tree. Manual [51, 52] and tag-based [53, 54] approaches are fundamental for Web page segmentation. By observing the structure of individual Web pages, a programmer writes a program (wrapper) to extract data records by regular expressions or specifying particular paths. Manual approaches can accurately extract data records from an HTML document. They are frequently used to obtain periodic data (such as stock movements or price trends) from multiple Web pages. Since a programmer must define different extraction patterns for each Web page, these approaches are not suitable for automatic data records extraction from heterogeneous Web pages.

Kovacevic et. al. [55] proposed a page-layout approach by observing the Web page layout. There are some design page-layout patterns (such as header, footer, left, right, and center) that allow a page to be segmented into fixed regions. This approach is effective for normal layout patterns of most Web pages. However, it cannot be applied to all Web pages. The event Web page of a shop is most likely designed unconventionally; therefore, this page-layout-based approach is unsuit-

able.

Win [49] introduced the concept of Web page blocks, which refers to a range in a Web page representing contents of a homogeneous topic, such as a news list or an event list. This method represents a segment-and-record-based Web segmentation, which not only finds a similar Web data region (a segment), and also subdivides the region into records, is proposed.

Cai et al. [56, 57] proposed a vision-based page segmentation (VIPS) [58] approach, which simulated the human visual perception to segment Web page blocks, and distinguished different parts of a Web page, such as lines, blanks, images, and colors. This approach can be applied to automatic block segmentation of Web pages. However, the extracted data records with VIPS are sometimes not atomic. For example, a list of titles in a $< div >$ element may be estimated as one block (one record) by VIPS, rather than segmented into individual titles. Also, some hidden items in a Web page (such as a slider bar) cannot be extracted with VIPS; thus, it is not suitable for extracting event records from Web pages.

Zhai and Liu [59, 60] developed a tree-based Web page segmentation approach called DEPTA [60]. This approach extracts Web data from the viewpoint of Web page generation. Data records from a data table are typically presented in continuous regions on a Web page and formatted using fixed HTML templates. By matching partial tree structures, data records can be extracted; thus, it is suitable for converting from semi-structured data to structured data. However, this approach does not take into account how to extract an independent data record without any similar continuous region on a page. It also does not consider specific HTML elements which are unrelated (such as inline elements) to partial tree matching but increase the complexity.

2.1.2 Event Data Extraction

Data records extracted from Web pages contain large amounts of unrelated data. A textual classification algorithm is commonly used for unrelated data cleaning. Ritter et al. [47] and Muhammad et al. [62] introduced data extraction from Twit-

ter using a supervised textual classifier and its evaluation index using precision, recall, and F1-score. The scheme of spatial-temporal event data, extraction concepts, and evaluation index are also applied to the task presented in this thesis. In this thesis, non-event data are filtered out in a text classifier. For a text classification task, mapping a paragraph to a dimensional vector is the key to maintaining classification performance. A one-hot paragraph vector [63] can solve many text classification problems. However, it may be sparse and have high dimensionality, which may increase classification errors. Mikolov et al. [66] proposed a neural network model called Word2Vec, which clusters similar words (e.g. 'coupon' and 'campaign') with smaller cosine similarities. It effectively builds the semantic relationship between each word and solves the sparse problem of training a dataset. However, Word2Vec works on the word level, which does not offer any direct implementation that yields a paragraph vector.

Le et al. proposed a paragraph vector framework called PVDM [67]. It considers a paragraph token as another word called 'memory' in a paragraph and trains the paragraph token as the paragraph vector in Word2Vec processing. However, in this task, the paragraph as an event record on Web pages tends to be short. In the case of training with few words, the convergence of PVDM is not complete. Repetitive training can solve this question but requires more computing resources. On the other hand, based on Word2Vec, PVDM solves some semantic problems. Nevertheless, it lacks an optimization for a specific classification task. Some of the most frequently used methods for classification are the Naive Bayes classifier [68] and the k-Nearest Neighbors (k-NN) [68] classifier. However, because there is noise in training data, both Naive Bayes and k-NN easily over-fits when the sample size increases. Thus, it is difficult to ensure their robustness.

2.2 Related Techniques for Signage Identification

Related literature about store signage identification from natural scenes is introduced in this section. Because signage in a natural scene is typically matched with text or images in data resources, the literature is organized into groups of textual-matching based methods and image-matching based methods.

2.2.1 Textual-matching Based Signage Identification Methods

Textual matching-based methods [70, 71] use OCR engines, which convert signage into text. A local lexicon is generated in advance using store names and other relevant text information. The system then searches for the recognized text within the local lexicon. When the text of a signage matches that in the lexicon, the system assigns the corresponding store ID to the signage.

The performance of textual matching methods relies upon the performance of OCR engines. Under ideal conditions, in which the OCR engine can accurately recognize signage text from natural scenes, textual methods are closer to human behavior and better than image matching both in accuracy and speed as symbolical retrieval methods. However, signage from the natural scenes is not always presented in an easy-to-recognize style. There is also a large percentage of signage with multi-language characters, specialized fonts, and non-character logos. Multi-language and individual fonts significantly limit the performance of OCR engines and non-character logos cannot be recognized by OCR. Therefore, there are obvious limitations when applying textual matching to signage identification tasks.

2.2.2 Image-matching Based Signage Identification Methods

Image-matching based methods attempt to match the query image with visually similar images in a target image dataset. It is important to choose appropriate image matching methods according to various types of target image datasets.

Fingerprint data collection is a method for target image dataset establishment.

The main idea of fingerprinting is to identify user-uploaded street images by matching them against manually labeled fingerprint street images [34, 35]. Manually labeling the data is the most straightforward way to clean up a dataset. For example, in Xu et al.s approach [34], three facade images of each store are taken and labeled with store IDs manually, to build the fingerprint database. When a user uploads an image that is taken by a mobile phone to a server, the server retrieves the most similar signage from the fingerprint database and returns the corresponding store information to the user. The advantage of a fingerprinting method is that it can identify diversified target objects by matching them with a small number of labeled target images. Also, finger-printing methods avoid many visual artifacts from natural scenes and can reach high identification accuracy because user-taken images and fingerprint images are usually shot from the same location. Therefore, classic feature point matching methods using local features such as SIFT [40], ORB [82] and their extended algorithms such as Bag of Visual Word (BoVW) are efficiently used for these fingerprint databases. However, it requires manual collecting and labeling image data in advance, so the technique cannot automatically re-adjust labels when the store location or tenancy changes.

They [7, 36] have achieved unprecedent success on object identification recently. They, in particular, are known for requiring large quantities of training instances to avoid over-fitting. Existing datasets support high quality and large target datasets by manual labeling or data filtering. Wojna et al. [72] benchmarked their street sign identification method with the French Street Name Signs (FSNS) dataset [73], which contains over one million labeled images of visual text. Movshovitz-Attias et al. [74] focused on identifying store categories (restaurant, gas station, etc.) from street level imagery. They extract three million target images consisting of 2,000 categories from Google Street View [1]. To ensure enough training data per category, they omit labels whose frequency is very low, resulting in 1.3 million samples in 208 unique categories. Yu et al. [36] addressed the problem of detecting facades in street-level imagery. They label approximately two million

[1]Google Street View: https://www.google.co.jp/intl/ja/streetview/

panorama images through a crowd-sourcing system. Deep learning methods avoid many visual artifacts from natural scenes and can achieve human-level accuracy. However, acquiring a large target dataset of high-quality labeled data for training is a challenging task, and it is also impossible to ensure that the ready-made datasets can cover all scenes, as store identification is sensitive to location. This leads to unavoidable human intervention and specialized training to deploy the process in locations.

Another feasible means for target dataset collection utilize the Web-mining technique. By 'recycling' Web resources, Web-mining can dynamically establish a local target dataset at lower costs. Zamir et al. [37] presented a multimodal method that combines a textual matching model and an image matching model to identify stores in an urban image automatically. The image matching model also utilizes Web mining images from Web resources. In the identification step, the two models separately generate the Probability Distribution Function (PDFs) for each store. The two PDFs obtained from the textual matching and the image matching models are fused. The multimodal method is more robust than textual matching or image matching alone. The two models compensate for each others shortcomings; When the OCR engine has difficulty recognizing characters in signage, the image matching model may locate a match from corresponding Web-images. Likewise, when a facade image does not exist on the Web, the OCR engine may be able to recognize the store name directly.

Another way for target dataset collection is crowd-sourcing [74, 75]. Although it may produce noise, the costs are much lower than the data collected by technicians. It can also ensure data to be up-to-date.

2.3 Summary

In this chapter, the methods of related approaches that aim to solve similar issues to these presented in this thesis were analyzed.

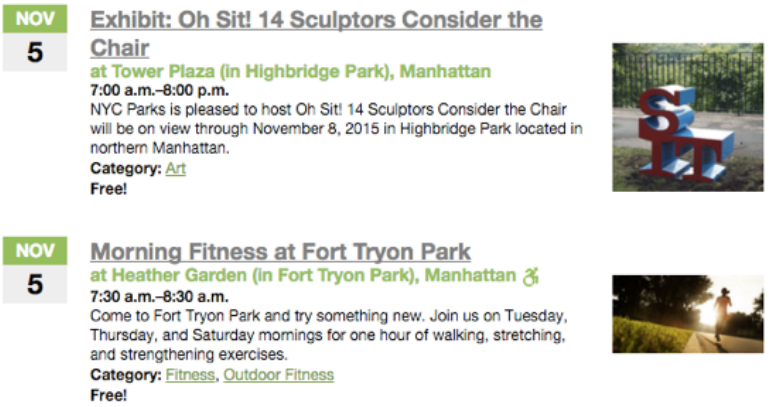
There are two tasks for extracting informative content from a Web page. The first one is segmentation of a Web page into segments and records where a record contains an atomic data item. The related approaches of this can be categorized into tag-based, vision-based, and partial tree-based methods. The tag-based method is suitable for accurate Web page mining, but it cannot support the data extraction tasks of many heterogeneous Web pages. The vision-based method can extract visually significant areas in a Web page, but it is difficult to grasp the granularity of data records. The tree-based method can extract repetition structures in a Web page, but it is difficult to extraction an independent structure. The second task is extracting event data from segmented records.

For matching the store signage appearing in street views with Web images, there are two approaches: textual and image matching. The shortcoming of textual matching methods is they are subject to OCR performance, which is sensitive to the font style and language groups. The image matching approaches attempt to match query images to other similar viewings using pre-labeled copies from a target dataset. Manual approaches, such as a fingerprinting database can ensure high-quality data but collected data must be fed manually, which significantly adds costs. Deep learning approaches requires a large amount of high-quality training data. Utilizing Web-crawled information is a way for automatic dataset generation at lower cost, however, imbalanced and noisy data can adversely affect identification accuracy.

Chapter 3

Web Page Segmentation

For extracting event records from Web pages, a Web page segmentation algorithm is proposed in this chapter. It identifies records as semi-structured data on Web pages and converts them into structured data. The challenge is how to generate wrappers that segment an entire Web page into records and ensure each record includes an atomic data record. Here, a wrapper indicates a program that describes a set of extraction rules suitable to extract information from a Web page [100]. Partial tree alignment is an important concept in wrappers generation. Figure 3.1 shows an example of an event segment in a Web page [101], where each data record (in the list) has the date, address, description, categories, etc. They align continuously with a similar structure. Therefore, it is possible to detect records according to matching similar structures in the Web page. This process is introduced by first explaining inline-level element pruning, which helps us prune the HTML tree to improve the accuracy of the wrapper generation and reduce computational complexity of the Web-data-record extraction algorithm. After that, the partial tree matching algorithm is introduced. Finally, by a backtracking process, some independent records without any continuous sibling can also be extracted. The process flow of these three steps is shown in Figure 3.2.



(a) Event-record list in a Web page

Date	Title	Position	Description	Categories	Price
Nov 5 7:00-20:00	Exhibit...	Tower Plaza	NYC Parks...	Art	Free
Nov 5 7:30-8:30	Morning...	Heather Gar...	Come to Fort...	Fitness, Out...	Free

(b) Event records table in a database

Figure 3.1: Event records in a Web page. (a) Event-record list in a Web page is generated from (b) an event record table in a database. Each row in the record table is filled with the same HTML structure to generate the list of event records. Therefore, event records have the same HTML structure. Wrappers can be segmented by matching HTML structures.

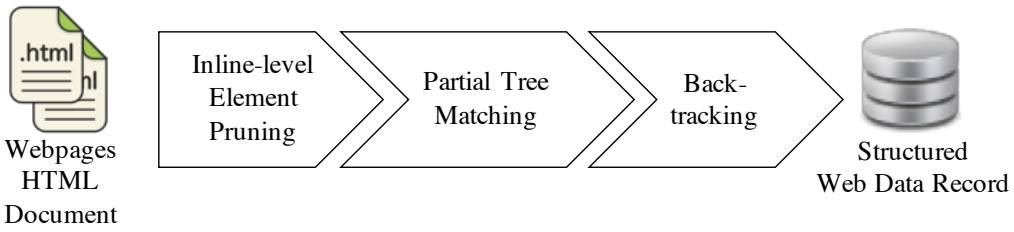


Figure 3.2: Process flow of Web page segmentation.

3.1 Inline-level Element Pruning

The pruning process is for removing elements, which are not related to the HTML structure matching. Since the target of Web designing is to enable users to identify data records on a Web page as easily as possible, designers keep data records identifiable through space separation. Some HTML elements affect the spatial structure of Web pages; but others do not. In the HTML 2.0 [102] standard, HTML elements are divided into block-level elements and inline-level elements. Because inline elements normally do not significantly affect the Web page structure, they should not be used for partial tree matching in a Web page. Then inline elements in a HTML document may affect the result of matching. Consequently, pruning of inline elements is performed to reduce the computational complexity for partial tree matching.

Algorithm 1: Inline-level Element Pruning

input : An HTML document D.
output: A pruned elements queue $Q[]$.

```
1 Q[]={D.Body()}, i=0;
2 while i < Q[].Size() do
3 |   E[] = Q[i].Children();
4 |   foreach element e of E do
5 |   |   if e.isBlock() then
6 |   |   |   Q[] ⟵ e;
7 |   i = i + 1;
8 return Q[];
```

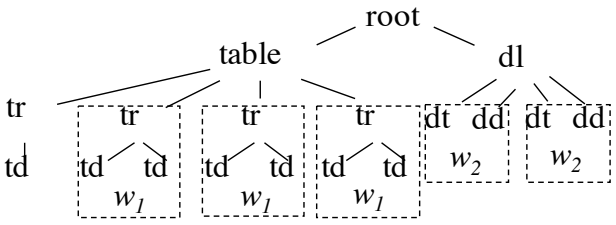
The algorithm for inline-level element-pruning is introduced in Algorithm 1. Pruning is started from scanning an HTML document D in breadth-first search at lines 2 to 7. Here, Body is the $< body >$ element of the HTML document D, and a queue $Q[] = D.Body()$ is initialized with a single element. $Q[]$ has a cursor i from 0 at line 1 and is initialized with only one element, the $< body >$ element of the HTML document D. At line 2, i is a cursor for iteration of $Q[]$. At line 3, children elements of $Q[i]$ are stored in the queue $E[]$. At lines 4 to line 6, each element in $E[]$ is iterated. At lines 5 and 6, if the child element is a block-level element, it is added into the end of the queue $Q[]$. At line 7, i is increased by 1.

The pruned tree is stored $Q[]$, which is used for partial tree matching in the next step.

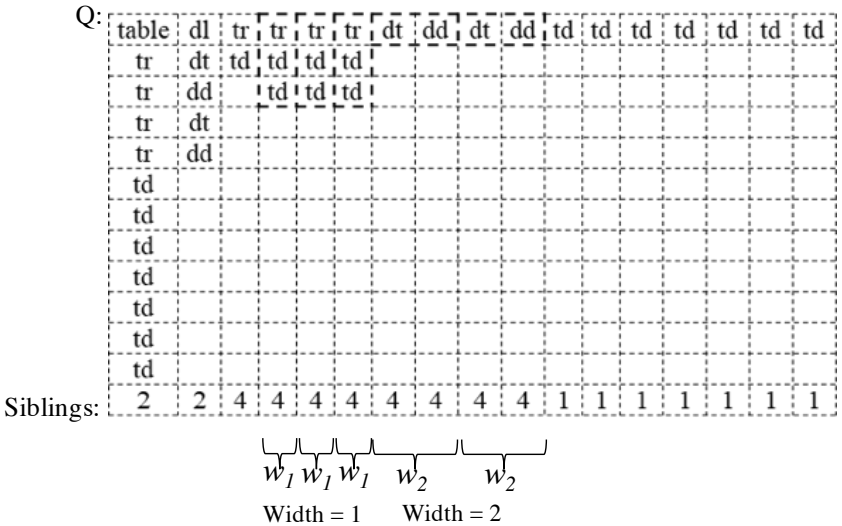
3.2 Partial Tree Matching

The reason of adopting breadth-first search in Section 3.1 is that a Web page is designed from outline to details, namely from a large to a small area [46]. A breadth-first search precisely scans a Web page from a large area to a small area. $Q[]$ from the pruning process happens to be a sequence arranged in a breadth-first search, so it can be directly used for partial tree matching.

The process of partial tree matching and wrappers generation is explained here through an example of a tree structure, which is shown at the top of Figure 3.3. Let's assume there are two wrappers W_1 and W_2 for five partial trees. For the



(a) Example of an HTML tree structure



(b) The map extends each element of the tree structure

Figure 3.3: An Example of partial tree matching with two types of wrapper structures. The (b) map illustrated in (b) extends each element in Q[] in the first row. From the second row, there are corresponding child elements. The last row shows the number of siblings of the element that determines the width of the matching window. We can observe that wrappers can be extracted by matching columns in each window (the dotted region in the table).

clarity of explanation, the tree $Q[]$ is extended into the first row of the map, as shown at the bottom of Figure 3.3. The first row is assigned to each element in $Q[]$ in a breadth-first search. From the second row, there are corresponding descendant elements, which are also aligned through breadth-first search, namely the partial tree structure of the element. The matching process is from left to right.

Obviously, the three $< tr >$ records can be extracted by matching each column. However, this is not sufficient in some cases: Some records may be constructed with multiple elements in the same level, such as two $< div >$ or more. Figure 3.3 gives an example in which a $< dt >$ element and a $< dd >$ element construct one record. In this case, the matching function needs to compare two elements. A matching window mechanism is designed to address this case. It determines how many columns of same-level-elements are combined to match. The width of the window loops from one to half the number of sibling elements, which is shown as the last row in Figure 3.3. In this case, when the window width increases to two, wrapper W_2 is generated.

Algorithm 2 shows the details of the partial tree matching. $Q[]$ is given from the inline-level element-pruning process as the input. At line 1, because a record may include multiple elements, a two-dimensional array $R[][]$ to store extracted records is initialized, and an i for $Q[]$. At line 3, if $Q[i]$ has been marked as an extracted element, skips it and continues to the next loop. The extracted-element-marking process is shown at lines 12 to 14. At line 6, it initializes a w of the matching window size from value 1. At line 7, the w increases by 1 (at line 18) until it reaches half the number of $Q[i]$ siblings. At lines 8 and 9, it matches the columns in the current window with the left and right windows and stores the results in two Boolean variables l and r. At line 10, if l or r is true, elements of the current window are added into array $R[]$ as an extracted record (line 11). At line 12, all the elements of the current window are marked as 'extracted'. At line 13, all the child elements of the current window are marked as 'extracted'. At line 14, all the parent elements of the current window are marked as 'extracted'. Line 15 skips the extracted elements of the current window and updates i to $i + w - 1$. Line 16

Algorithm 2: Partial Tree Matching

input : $Q[]$ is from inline-level element-pruning.
output: $R[][]$ stores extracted records, and $Q[]$ with marked elements.

```
1  R[][] = {}, i = 0;
2  while i < Q[].Size() do
3      if Q[i].isExtracted() = true then
4          continue;
5      else
6          w = 1;
7          while w < Q[i].SiblingsNumber() / 2 do
8              l =Match({Q[i] to Q[i + w − 1]}, {Q[i − w] to Q[i − 1]});
9              r =Match({Q[i] to Q[i + w − 1]}, {Q[i + w] to Q[i + 2w + 1]});
10             if l or r then
11                 R[] ⟵ {Q[i] to Q[i + w − 1]};
12                 {Q[i] to Q[i + w − 1]}.MarkExtracted();
13                 {Q[i] to Q[i + w − 1]}.MarkAllChildrenExtracted();
14                 {Q[i] to Q[i + w − 1]}.MarkAllParentsExtracted();
15                 i = i + w − 1;
16                 break;
17             else
18                 w = w + 1;
19     i = i + 1;
20 return R[][], Q[];
```

breaks the loop in line 7 and goes to line 19. If it is not matched at line 10, the w increases to 1 at line 18. At line 19, i increases to 1. The extracted records in the array $R[][]$ and marked $Q[]$ are returned at line 20.

This partial tree matching algorithm extracts major data records on a Web page. However, independent records without any continuous or similar sibling are not extracted. At the next stage, a backtracking process performs an independent records extraction.

3.3 Backtracking: Independent Record Extraction

When a Web designer tries to emphasize an important event data record, the record is most likely a special structure on the page, although it may be selected from the same data table. Figure 3.4 shows a typical independent record in a Web page [103]. It can be seen that the designer draws the event record 'Genome Integrity Discussion Group in a larger area than the others. Because this event element has no sibling, the partial tree matching algorithm does not work in this case.

Fortunately, from many samples like this, a common feature of independent records is observed. Records from a data table tend to be drawn at the same depth in the HTML tree with the same parent element. If each extracted parent element is marked during partial tree matching, some independent wrappers with a marked parent element can be extracted. Because this process is after partial tree matching and scans the tree in breadth-first search one more time, it is named as the backtracking process. Therefore, by backtracking extracted records, these independent records may be extracted to a certain extent. As shown in Figure

Figure 3.4: Common structure of an independent record. Three event records are listed on the right side. They can be extracted by partial tree matching. Another highlight event record is drawn on the left side independently, which cannot be extracted. Both of them have the same parent element. The parent element is marked when three records are extracted, and then, the independent record can be extracted via the parent element.

3.5 the wrapper W_1 is generated by partial tree matching and W_2 wraps an independent record. W_2 can be generated by marking the parent element P, which is marked when W_1 is generated.

Algorithm 3: Backtracking

input : $R[][]$ and $Q[]$ are from partial tree matching.
output: $R[][]$ stores extracted records.

```
1 i = 0;
2 while i < Q[].Size() do
3     if Q[i].isExtracted() = false then
4         if Q[i].Parent().isExtracted() = true then
5             R[] ⟵ Q[i];
6             Q[i].MarkExtracted();
7             Q[i].MarkAllChildrenExtracted();
8     i = i + 1;
9 return R[][];
```

As shown in Algorithm 3, queue $Q[]$ is scanned again in breadth-first search. The queue $Q[]$ and extracted $R[][]$ are output from partial tree matching as the input. At line 1, an i for $Q[]$ is initialized. At line 2, i is looped from 0 until the size of $Q[i]$. At line 3, if $Q[i]$ has not been extracted yet, the program runs lines 4 to 6. At line 4, if the parent element of $Q[i]$ is marked as an extracted element, $Q[i]$ is extracted and added into $R[]$ at line 5. Note that, the marking is done during partial tree matching as shown in Algorithm 2 at lines 12 to 14. At line 6, $Q[i]$ is marked as 'extracted'. At line 7, all the children elements of $Q[i]$ are marked as

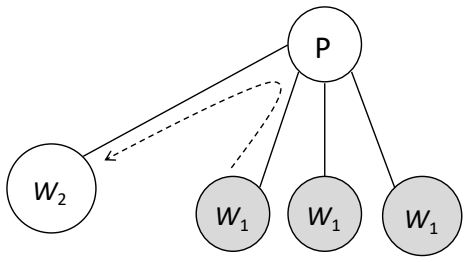


Figure 3.5: The backtracking mechanism for independent records extraction. Wrapper generation by backtracking the marked parent element. When wrapper W_1 is generated, the parent element P is marked as 'extracted'. After that, all tree elements are backtracked to generate wrapper W_2, which has not been extracted yet but has marked the parent element.

'extracted'. At line 11, all the extracted records in array $R[]$ are returned.

3.4 Evaluation Experiment

3.4.1 Evaluation Criterion

In this section, the criterion for Web page segmentation evaluation is defined. In related approaches, each Web page segmentation algorithm has its separate evaluation criteria for its particular task. It needs to be unified into one criterion.

VIPS [56] is mainly used for Web search engine. It requires the algorithm capable of segmenting associated Web information into blocks, such as the semantic relationship of the images and their surrounding texts, and then these text representations could be used in a Web image search system. Therefore, the goal of VIPS is to find out blocks (similar to our segments), which convey semantic meanings and ensure the semantics between blocks are independent of each other. Figure 3.6 shows an example of VIPS segmentation. It could be seen that each block contains homogeneous information, such as the header, link menu, news list,

Figure 3.6: VIPS segmentation result of an IEEE Web page. The left part shows the page with different regions (different visual blocks in VIPS algorithms) marked with rectangles.

login form etc. The navigation bar is split as a block instead of individual buttons. This criterion does not apply to the extraction of event records, because a list of event records should be treated as a block.

Zhai et al. proposed DEPTA [58] to identify atomic data records in a Web page. This criterion also applies to the extraction of event records. In their experiment, they deliberately ignore un-informative regions in a Web page. For example, they did not include navigation areas, which may also have regular patterns. The DEPTA evaluates its performance in precision and recall. They first annotate records in a Web page manually. The number of manually annotated records is n. Then they run the algorithm on the page. They calculate the x as the number of extracted results that are incorrect, and y as the number of results that are not extracted, so that there are $precision = \frac{(n-x)}{n}$ and $recall = \frac{(n-y)}{n}$. The DEPTA achieves higher precision and recall than others.

Unlike DEPTA, the proposed algorithm focuses on event records extraction. Same as DEPTA, un-informative regions in a Web page should also be ignored. But in the evaluation of DEPTA, there is no clear definition of un-informative regions in a Web page. In this thesis, an 'un-informative regions' is clearly defined as a region from which non-event record was. Thus, the criterion for this event extraction task can be defined. The event records in a Web page are annotated manually and the number of event records is calculated as n. Then the proposed algorithm runs on the Web page and m which is the number of extracted records can be calculated. Then, the recall = $\frac{x}{n}$, which means the number of extracted event records in a Web page. Achieving higher recall may increase the performance of event records extraction. The precision = $\frac{x}{m}$ is also calculated. It could be deducted that the numerical value of precision is small because a mass of non-event records are also extracted in this stage. The results are introduced in the next section in detail.

3.4.2 Evaluation Results

For evaluating the proposed Web page segmentation algorithm, 87 Web pages of 96 stores were chosen as sample. The 96 shops are in mall A and mall B, two underground shopping streets in Nagoya, Japan. The other nine Web sites were off-line or unavailable.

The VIPS [56], DEPTA [59], and the proposed algorithms are compared using the criterion described above. As shown in Table 3.1, the first and the second columns are the serial number and the URL of each Web page, respectively. The third column is the number of event records on the Web page. In the last three columns, VIPS, DEPTA, and the proposed algoithm are run on each Web page and the current number of event records extraction is calculated, respectively.

The recall in the last row is calculated with the rate of correct extraction number of event records in the total number of event records. As a result, VIPS can correctly extract 41.64% event records from 87 Web pages. Since VIPS is developed for relevant data clustering in a Web page, it seems that it is not suitable for event extraction. Meanwhile DEPTA can correctly extract 94.15% event records, and the proposed method, 98.71%.

Analyzing the extraction results in the experiment, there are three cases as follows.

Figure 3.7: Example of event records on the Website #5.

Table 3.1: Experiment results of Web records extraction. x/m: m is number of all extracted records by the algorithm; x is the number of correctly extracted event records.

#	URL (http://)	Event	VIPS	DEPTA	Proposed
1	www.hitsumabushi.co.jp/	1	1/7	0/17	1/10
2	diamond-shiraishi.jp/	1	1/57	0/28	1/27
3	chronos.chicappa.jp/chronscythe/...	6	6/15	5/17	6/24
4	www.fujiya-peko.co.jp/	6	6/21	5/46	6/10
5	www.csmen.co.jp/	6	6/17	5/33	6/32
6	ryoguchiya-korekiyo.co.jp/	1	1/8	0/14	0/17
7	www.lushjapan.com/	19	19/47	17/93	19/111
8	www.e-kaban.co.jp/	1	1/19	0/15	0/25
9	www.fivefoxes.co.jp/brand/	1	1/25	0/27	0/30
10	www.i-polaris.jp/shop/	7	7/49	4/40	7/23
11	www.abc-mart.net/shop/	13	10/40	13/352	13/311
12	www.exelco.com/	3	0/12	3/30	3/37
13	www.ukiya.co.jp/top_sozai/	5	5/24	2/40	3/14
14	www.heartflower.co.jp/	5	0/13	5/9	5/17
15	www.godiva.co.jp/	16	11/42	16/32	16/66
16	yamamotoyahonten.co.jp/	5	0/14	5/28	5/21
17	www.uniqlo.com/jp/	44	44/108	39/175	44/74
18	www.francetei.com/	7	1/31	7/21	7/42
19	www.honeys.co.jp/	6	0/3	6/13	6/43
20	www.botejyu.co.jp/index.html	6	0/17	6/35	6/16
21	www.bodywork.co.jp/	11	0/89	11/126	11/149
22	www.bodywork.co.jp/brand/...	6	0/27	6/103	6/153
23	www.erina-t.com/	14	8/23	14/32	14/27
24	pastel-pudding.com/	6	0/14	6/16	6/46
25	www.starbucks.co.jp/	6	0/6	5/68	6/79
26	www.komeda.co.jp/	9	1/14	8/37	9/20
27	www.ebidote-shokudo.jp/	9	0/19	9/18	9/15
28	www.leilian.co.jp/	3	0/6	0/27	0/17
29	www.robe-de-tisse.jp/	10	0/21	10/17	10/14
30	kireidechu.com/	10	0/119	10/31	10/63
31	top.dhc.co.jp/dshop/store/	14	4/13	14/43	14/35
32	www.fancl.jp/index.html	10	0/9	10/74	10/95
33	www.regal.co.jp/shoes/	10	0/7	10/56	10/36
34	www.van.co.jp/	13	0/13	13/27	13/14
35	yomenya-goemon.com/	12	2/29	10/33	10/24
36	kikuchi-megane.co.jp/	25	9/152	23/86	25/38
37	www.heart-up.com/	19	0/4	19/92	19/61
38	wakashachiya.co.jp/	18	0/23	17/36	18/36
39	www.proportionbd.com/	19	1/79	18/49	19/31
40	www.mcdonalds.co.jp/	20	0/7	20/28	20/73
41	suppondo.co.jp/	26	6/31	26/58	26/58
42	watashin.jp/	20	0/66	20/52	20/91
43	www.kazki.co.jp/	24	0/15	24/25	24/57
44	www.makelet.com/	29	0/90	29/41	29/30
45	www.peakmanager.com/pcWeb/	34	0/72	33/69	33/71
46	www.pokkacreate.co.jp/	53	6/57	53/352	53/437
47	www.n-rs.co.jp/	82	10/55	82/130	82/194
	(Other 42 Web pages)	184	184/1408	184/1304	184/1527
	(7 Web pages were disable)	0	0	0	0
	Total	1209	356/3008	795/4064	844/4441
	Precision		11.84%	19.74%	26.89%
	Recall		41.64%	94.15%	**98.71%**

Figure 3.8: Example of records that cannot be correctly extracted by neither DEPTA nor the proposed method.

1. `Event records that can be extracted by the proposed method but not by VIPS.`

   For example, in Website #21, VIPS extracts the event segment as one block by not records. The partial tree matching of VIPS and the proposed method can solve this issue.

2. `Event records that can be extracted by the proposed method but not by DEPTA.`

   For example, an event segment in Website #5 is shown in Figure 3.7. Both DEPTA and the proposed method can extract the event list at the right by partial tree matching mechanism. But, the independent record at the left can only be extracted by the proposed method using the backtracking mechanism.

3. `Event records that cannot be extracted by neither DEPTA nor the proposed method.`

   A few Web pages cannot be correctly segmented with the proposed method because of their unusual designs. Figure 3.8 is an example of an unusual Web design. In this page, records 1, 2, and 3 should be extracted as event records. However, because records 1 and 2 have different HTML structures, the proposed method cannot locate then correctly. As a result, record 1, 2, and 3 were extracted as one record. Because of this issue, the recall of the proposed method drops to 98.71%. Notice that, it is only an unusual situation on Web page segmentation.

At this stage, since the target of Web page segmentation algorithm is to extract as many event records out of an Web page as possible, the low precision rate is not a serious problem that a mass of noise data (non-event records) are also extracted from Web pages. Therefore, an event record classification is proposed to extract event data correctly from Web records in the next chapter.

3.5 Open Source: Web Page Segmentation Algorithm

The source code of the proposed Web page segmentation algorithm is published as an open source code on GitHub [112]. The algorithm requires a URL of the Web page as input. A built-in GUI-less browser Selenium [113] with Chrome kernel crawls the HTML document from any available URL. According to the process proposed in this Chapter, the algorithm outputs a JSON file with segments and records extracted from the specified Web page.

Figure 3.9 shows an example for segmenting the official Website of the Tokyo Olympic Games [1]. There is a segment about news (the blue box) with four records 1 to 4 (the red boxes) in the Web page. The program segments them and outputs a JSON file as show in the gray box. It gives a segment with a unique segment id 4 in the Web page. The red box in the JSON file corresponds to record 1 in the Web page including texts, images, and links in it. Users can use these data to build their own database or projects.

For testing the performance, the algorithm is implemented a PC with Intel Core i7-7700HQ 2.80GHz CPU and two DDR4 2400MHz 16GB memories with 100 Mbps network. Extracting all 74 top-pages in the shopping mall A, takes about 370 seconds in single thread; about 5 seconds for each Web page. This can support a real-time online service such as the Event.Locky introduced in Section 4.4. However, the size of the Web page can significantly impact the extraction time. Although the proposed Web segmentation algorithm starts breadth-first searching from the $< body >$ element, so that it can filter out some irrelevant HTML elements such as the *head* data before the Inline-level Element Pruning process, segmentation of a large Web page is still time-consuming. Figure 3.10 shows a scatter chart, which draws processing time of proposed Web page segmentation and scales (number of elements) of Web pages in the shopping malls A and B. This process excludes the Web page downloading. We can see more than half of Web pages have less than 1,000 elements and the process time is less than 0.1 seconds. But if elements of a Web page are more than 3,000, it costs

[1] https://tokyo2020.org/en/

Input: URL (https://tokyo2020.org/en/)

Output:


```
{
  "segment_id": 4,
  "css_selector": "html > body:nth-child(2) > div > div:nth-child(5) > div:nth-child(2) > main > section:nth-child(5) > ul:nth-child(3)",
  "records": [
    {
      "record_id": 12,
      "texts": ["Sponsors", "11 Oct. 2018", "Tokyo 2020 Welcomes AOKI Holdings Inc.  as an Official Supporter"],
      "images": [
        {
          "src": "https://tokyo2020.org/assets/img/pages/top/photo_noimage_01.jpg",
          "alt": "",
          "bg_color": "255,255,255",
          "path": "data/tokyo2020/images/12_0.jpg"
        }
      ],
      "css_selector": ["html > body:nth-child(2) > div > div:nth-child(5) > div:nth-child(2) > main > section:nth-child(5) > ul:nth-child(3) > li"],
      "links": [{"href": "https://tokyo2020.org/en/news/sponsor/20181011-01.html"}]
    },
    ...
  ]
}
```


Record 1

Figure 3.9: Example of Web page segmentation.

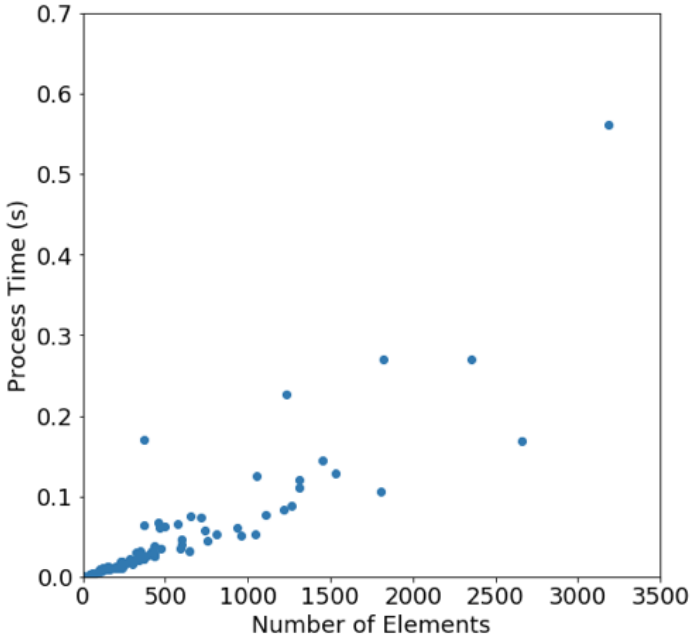


Figure 3.10: Scatter chart on processing time and number of elements of Web page segmentation.

about 0.6 seconds. Therefore, future work should consider more efficient filtering of irrelevant information strategy to reduce the processing time.

3.6 Summary

In this chapter, a Web page segmentation algorithm was proposed to convert a semi-structured HTML document into structured data. The assumption is that structured data in the backend of the Website is generated into an HTML document using the same template or wrappers with the similar and continuous HTML structures. A partial tree matching can find similar and continuous HTML structures and segment them as records. Several records with the same structure construct a segment. It can be regarded as a reverse engineering of HTML generation so that the segments and the records in a Web page corresponds to the tables and the rows in a relational database. With this algorithm, processable structured data can be obtained from a Web page, which benefits to Web mining for reusing the Web resources. Compared with DEPTA [59], which also applies partial tree matching, the proposed method introduces a back-tracking mechanism, which allows the extraction of individual elements. In the experiment, it was shown that it advances the extraction rate.

A limitation of this algorithm should be mentioned. In the experiment, only the segmentation of the top level, namely the top page, in a Website was investigated. Typically, a Website is constructed in a three-levels structure: the top page, several list-pages, and many content-pages. Although extracting from the top pages can support the task of event data extraction, it will be difficult to extract segments and records from it without any repeated list or elements. The future work should consider extracting structured data from not only one Web page, but also involving the whole Website with hyperlinks. Accordingly, in Chapter 5, an example attempts to extract store logo using hyperlinks and sub-Webpages in the whole Website.

Web page segmentation produces a mass of non-event data from Web pages. Thus, applying Web page segmentation alone is not enough to complete the event data extraction. It requires filtering of the data. Accordingly, in Chapter 4, event data extraction from a segmented Web page is discussed as a text classification task.

Chapter 4

Web Event Data Extraction

In Chapter 3, a Web page segmentation algorithm was proposed, which segments a Web page into atomic records. However, this process did not contain any semantic filtering, so non-event data useless to the user should be automatically deleted from them.

Thus, in this chapter, a Web event data extraction method is proposed. The assumption is that text in an event record could present some patterns at the semantic level. Intuitively, human beings can pick up keywords in the text to distinguish whether it is event information or not. Under this assumption, the record with event information should contain some decisive vocabularies, such as 'sale' or 'promotion' etc. These features can be learned by a text classifier. For example, Ritter et al. [47] and Muhammad et al. [62] introduced data extraction from Twitter using a supervised text classifier. The event information filtering method proposed in this thesis handles this as a text classification problem to extraction of event records from a Web page.

4.1 Background of Paragraph Vector Generation

Informative record from a Web page are extracted through a text classification task. Here, the text of an event record is called a 'paragraph'. As a semantic classifier for text, it is essential to start from mapping the paragraph into a math-

ematical model. One-hot representation [65] is the most commonly used model. A paragraph $\vec{X}$ is mapped into a vector $\mathcal{X}$ as follows.

$$\mathcal{X} = [\omega_1, \omega_2, ..., \omega_i, ..., \omega_n]^T \qquad \omega_i \in \{0, 1\} \tag{4.1}$$

where the dimensionality n is predefined to the size of a dictionary. If a word ω_i appears in the paragraph, it is set to 1; otherwise 0. Although this model can solve many problems in text classification, it has two serious disadvantages. First, due to sparseness, each word is independent. The one-hot representation model cannot build semantic relationships between words (e.g., 'coupon' and 'campaign' are unrelated in the one-hot representation model, although they have similar linguistic functions). Therefore, the classification accuracy is restricted to the coverage of training data. Second, paragraph vectors are mapped to a high-dimensional space with a dimensionality of n ($n > 200,000$ in Japanese). The model with a high-dimension vector may reduce the efficiency of the classifiers.

Distributed representation models have been recently presented. Word2Vec [104] is the most common. By observing a word ω_i that appears near its context $[\omega_{(i-c)}, \omega_{(i-1)}]$ and $[\omega_{(i+1)}, \omega_{(i+c)}]$ with a probability $P(\omega_i|\text{context}(\omega_i))$, it clusters similar words with smaller cosine similarities in a low-dimensional vector space (usually n =50, 100, or 200). It mitigates the disadvantages with the one-hot representation model. However, Word2Vec works on the word level, which does not offer any direct implementation to yield a paragraph vector. Furthermore, as a word clustering algorithm, Word2Vec has no optimization for specific classification tasks. In this thesis, a weighted optimization paragraph vector mapping method that works on word vectors and improves the classification of paragraph vectors is presented.

4.2 Optimization Method: Weighted Paragraph Vectors

First, a paragraph vector is created from word vectors. A paragraph is mapped into an n-dimensional vector $\mathcal{X}$ using the mean of word vectors which constitute the paragraph. This is described as

$$\begin{aligned} \mathcal{X} &= \frac{1}{m}\sum_{i=1}^{n}\omega_i e_i \\ e_i &= [\delta_{0i}, \delta_{1i}, ..., \delta_{ni}]^T \end{aligned} \tag{4.2}$$

where m is the number of words in a paragraph, ω_i are the words constituting the paragraph, and n is the dimensionality of a paragraph vector, which equals the dimensionality of the word vector because paragraphs are mapped to the same vector space as words. It can be seen that the paragraph vector is mapped to the mean of words vectors. With this method, each word vector has a uniform weight of 1. In other words, each word vector equally contributes to the paragraph vector. However, some words in the dictionary are strongly related to the polarity of event-data classification (such as the words 'firework show', 'exhibition', etc.). In comparison, some words are weakly related to classification (such as 'the', 'and', etc.). If a word is strongly related to the classification, it should be given a higher weight. Geometrically, the paragraph vector should 'drift' close to the higher-weight word vectors; otherwise, far from the lower-weight word vectors. This is the concept of the weighted paragraph vector. Therefore, the equation of a weighted paragraph vector is given as follows.

$$\mathcal{X} = \frac{\sum_{i=1}^{n}\theta_i\omega_i\epsilon_i}{1+\sum_{i=1}^{n}\theta_i} \qquad \theta_i \in [0,1] \tag{4.3}$$

where θ_i is the weight of the i-th word ω_i in a paragraph. The domain of definition θ_i is a closed interval from 0 to 1. Because in some cases, the sum of weights may be 0 in a paragraph, 1 is added to the denominator to avoid an infinite value.

The training event dataset is given as $(p(j), y(j)); j = 1, 2, , M$, where $p(j)$ is

the j-th paragraph in the dataset (with size M) corresponding to a target variable $y(j) \in [0, 1]$. The weight θ_i is obtained by using Naive Bayes, which calculates the probability of ω_i that belongs to a target variable $y(i)$ as

$$p(y|\omega_i) = \frac{p(\omega_i|y)p(y)}{p(\omega_i)} \tag{4.4}$$

where the target variable y is set to 1 if the paragraph is an event record; otherwise, set to 0. When the training dataset is large enough, the probabilities can be approximatively considered as follows.

$$\begin{aligned} p(\omega_i) &\approx \frac{M(\omega_i)}{M} \\ p(y) &\approx \frac{M_y}{M} \\ p(\omega_i|y) &\approx \frac{M(\omega_i, y)}{M_y} \end{aligned} \tag{4.5}$$

where the integer M is the number of training datasets, $M(\omega_i)$ is paragraph in which the word ω_i appears. M_y is that with y, and $M(\omega_i, y)$ is that including ω_i also with y. By substituting Eq. (4.4) with Eq. (4.5), the probability of y given ω_i is obtained as follows.

$$p(y|\omega_i) = \frac{M(\omega_i, y)}{M(\omega_i)} \qquad p(y|\omega_i) \in [0, 1] \tag{4.6}$$

Note that, $p(y = 0|\omega_i)$ and $p(y = 1|\omega_i)$ are complementary. Therefore, any y can be used to obtain the probability $p(y|\omega_i)$. When $p(y|\omega_i)$ can be approximate to 0.5, ω_i is weakly related to the classification; When it is approximated to 0 or 1, ω_i is strongly related to the classification. Therefore, the following mapping function is given to calculate the weight θ_i.

$$\theta_i = |1 - 2p(y|\omega_i)| \qquad \theta_i \in [0, 1] \tag{4.7}$$

Substituting Eq. (4.3) with Eq. (4.7)), the weighted paragraph vector can be obtained.

4.3 Evaluation Experiments

The event record classification algorithm is evaluated by combination of the Web data extraction algorithm proposed in Chapter 3 and the event-record-classification algorithm proposed is this chapter. First, the event classifier must be trained. For event record classification evaluation, 23,000 records were labeled as training data, which were extracted from top-pages and their sub-pages of 96 stores in the Web data extraction experiment introduced in Chapter 3. These training data are used for the pre-training of Word2Vec and the proposed method. Second, about 4,000 out of 23,000 records extracted from top pages of 96 stores were used for evaluation.

4.3.1 Evaluation of Event-record Classification Algorithm

Cross validation was adapted on the event record classification evaluated by dividing 90% of the dataset as training data and 10% as the test data. A Japanese morphological analyzer Kuromoji [107] was applied for word segmentation and an open source SVM library, LIBSVM, developed by Chang et al. [98] as the classifier. The evaluation of the classification algorithms involved *precision*, *recall*, and F1-score $= 2 \times \frac{precision+recall}{precision \times recall}$.

The One-Hot Representation model, Word2Vec Mean Vector model, PVDM [67], were also implemented to be compared with the proposed Weighted Mean Vector model. PVDM is the most advanced paragraph vector generation method proposed by the Word2Vec team. It introduces training of a paragraph vector as a word into Word2Vec network. Training Wrod2Vec in a small dataset may cause under-fitting. Therefore, an additional experiment 'PVDMx10' that trains ten epochs on training data is also presented. The mapped paragraph vectors and their target variables were input into the SVM for supervised training. The kernel function adopted was the RBF, which has two undetermined parameters that are the penalty factor C and the influence factor γ. Parameter optimization is applied for each C and γ. It searches logarithm of RBF parameters C and γ until it achieves the best F1-score.

Table 4.1: Evaluation of Web event data extraction for each classification method.

Model	**$\log_2 C$**	**$\log_2 \gamma$**	**Precision**	**Recall**	**F1-score**
ONE-HOT	4	-6	81.62%	79.42%	0.81
W2V MEAN	5	1	86.92%	88.70%	0.88
PVDM	5	1	88.46%	88.94%	0.89
PVDM x10	8	1	88.73%	89.32%	0.89
Proposed	6	1	92.48%	90.46%	**0.92**

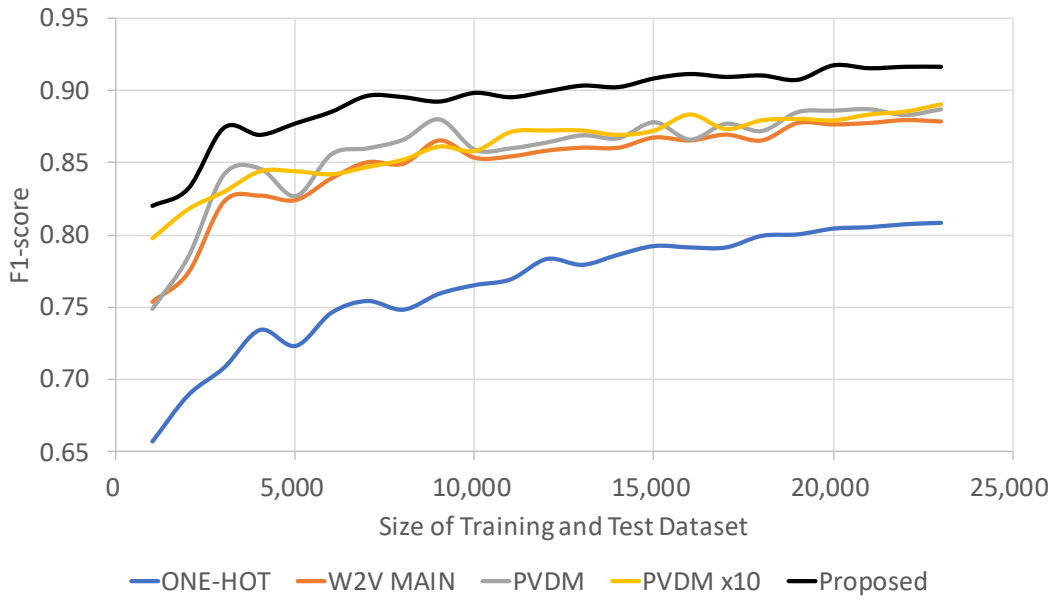


Figure 4.1: F1-score transition of classification models in each 1,000 training data.

Table 4.1 lists the experiment results. There was a significant improvement when a distributed representation was adopted instead of one representation. It can be seen that PVDM is significantly better than One-hot Representation and performs at the same level as Word2Vec Mean Vector. There must be mentioned an important feature of event records on the Webpage. These event records are as short as one or two sentences, or even a phrase. From Figure 4.1, when the PVDM model is trained on Web event records, the model is under-fitting with a low F1-score. The classification results are not satisfactory. By observing gray and orange lines in Figure 4.1, there is a improvement from the single PVDM training process. On the other hand, based on Word2Vec, PVDM solves some semantic problems. Nevertheless, it lacks the optimization for a specific classification task. The proposed method solves this lack through weighting each word for a given class. As a result, it achieves a higher F1-score than others. Thus we can say that the proposed algorithm is more specifically suited to the event data classification task.

4.3.2 Joint Evaluation

The crawler is developed with a Java library jsoup [105]. It is used to download Web pages from the Internet as HTML documents. The downloaded Web pages are input to the Web page segmentation algorithm proposed in Chapter 3. The most suitable event record classification algorithm is combined with each Web page segmentation algorithm, and the final event extraction result is compared. In this experiment, VIPS, DEPTA, and the proposed Web page segmentation algorithm are compared. For quantitative evaluation of 'backtracking', an algorithm is divided into two experiments: the algorithm without backtracking and another algorithm with backtracking.

As shown in Table 4.2, it can be seen that the proposed method performed identically as DEPTA without the backtracking process. The backtracking processing improved the recall by 2.04% and the F1 score by 1.6% from DEPTA.

Compared with the experiment of the event-record-classification algorithm,

Table 4.2: Evaluation of event record extraction capability.

Model	**Precision**	**Recall**	**F1-score**
VIPS	89.89%	40.23%	0.56
DEPTA	98.72%	70.98%	0.82
Proposed Method without Backtracking	98.72%	70.98%	0.82
Proposed Method with Backtracking	98.76%	73.02%	**0.84**

while the precision is higher, the recall is lower. This is because the density of event record in top-pages is much greater than that in sub-pages, it has a higher hit ratio. In summary, this experiment verified the combination of the two proposed methods is suitable for the event data extraction task.

4.4 Application: Event.Locky

An application Event.Locky [108] is developed to demonstrate the usability of the proposed Web page segmentation and Web event data extraction from Web pages. The aim is to publish event data for mobile users at anytime and anyplace throughout Japan. Figure 4.2 shows the process flow. The application installed on a mobile device sends to the server the device location information, which is obtained with the built-in GPS sensor or indicated by the user (step 1). As suggested by Sappelli et al. [93], the server searches nearby stores' information including their coordinates, addresses, Website URLs, and the type from a Point of Interests (POI) obtained through the search engine Google Places [83] (step 2). Next, the server requests these Websites and downloads their Web pages by using an inner crawler (steps 3 and 4). Because there may be multiple search results, the crawler is designed to be a multi-threading program that can download Web pages from all the Websites simultaneously. After that, the proposed event-record-extraction algorithm works on the Web documents (step 5). Finally, the server sends the event records to the user client and displays these records on the mobile device (step 6). Note that an event record includes the description, time, images, and hyperlinks. The time of event is extracted by the regular expression and the image of the event is extracted from HTML elements.

Figure 4.2: Processing flow of the Event.Locky application.

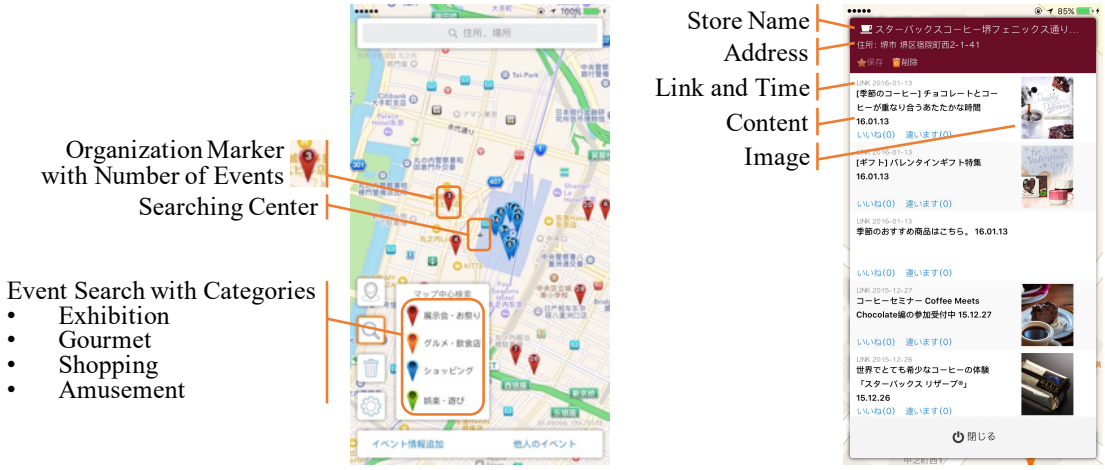


Figure 4.3: Main interfaces of the Event.Locky application. (a) markers with the number of event records of stores, museums, etc. on an electronic map; search coordinate is set at the center of the map (the cross marker), which by default is set to the current location obtained from a built-in GPS sensor. It can also be indicated by dragging on the map. (b) Detailed event records are listed after touching a marker sorted by event time.

The server of Event.Locky is deployed at a public data center with two Intel Xeon E3 CPUs and 2 GB memory. The programs are written in Java and run in Apache Tomcat 8 with Open-JDK 8.0. The maximum network throughput is 100 Mbps. The client runs on an iPhone 6 plus with 54 Mbps Wi-Fi network. The communication between the server and the client passes through the Internet.

Figure 4.3 shows the main interfaces of Event.Locky client application. Event data are categorized according to the type of sites into four categories: Exhibition (museums, parks, galleries, etc.), Gourmet (restaurants, cafes, bars, etc.), Shopping (malls, stores, markets, etc.) and Amusement (bowling alleys, cinemas, clubs, etc.).

Figure 4.4 shows the processing time of Event.Locky at train stations or downtowns of ten cities in Japan. At each area event data is extracted in the four categories. The average retrieval time was 2.1 seconds for Exhibition, 1.84 seconds for Gourmet, 2.44 seconds for Shopping and 4.61 seconds for Amusement. These should be acceptable for users. The geographical distance to Event.Locky's server does not evidently impact the response time, but the bottleneck is on the

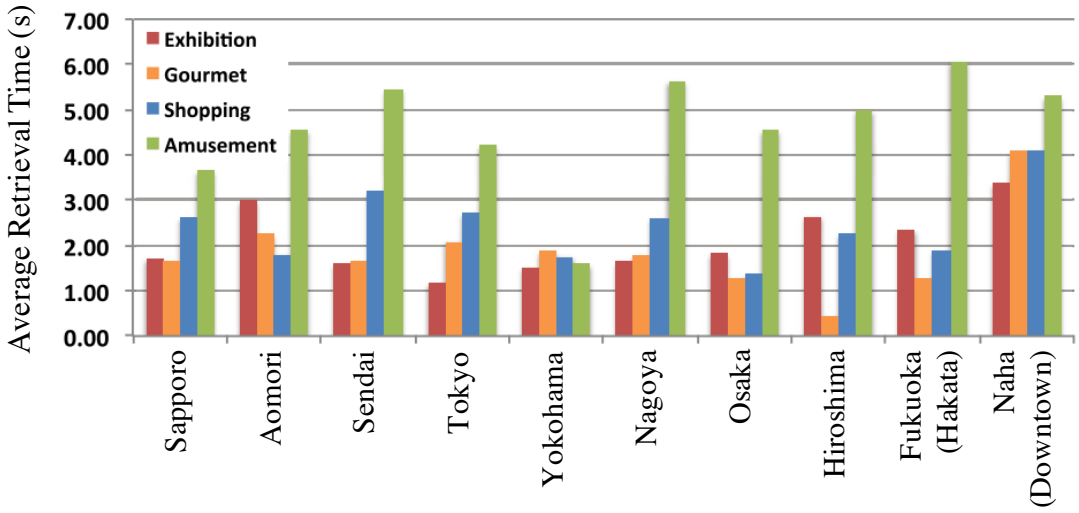


Figure 4.4: Processing time at train stations or downtowns of 10 cities in Japan.

crawler. Some Websites were not updated, so the crawler timed out. By upgrading the bandwidth and network performance, this problem can be properly solved. Nonetheless, the proposed event-data-extraction algorithm has enough capacity to support high-speed online retrieval.

4.5 Summary

A Web event data extraction method that filters out non-event data from extracted data records using machine learning was proposed. The algorithm was trained and evaluated with an actual dataset composed of 96 stores in Nagoya, Japan. As a result, the proposed algorithm achieved an F1-score of 0.92, an increase of 0.03 from the state-of-the-art event-classification algorithm. The combination of the proposed event-classification algorithm with the data-record-extraction algorithm achieved an F1-score 0.84 to extract event records from Web pages, which increased 0.02 from the state-of-the-art algorithm on event classification task. The feasibility of the proposed method in an actual online environment was demonstrated through the implementation of a demonstration application Event.Locky.

Chapter 5

Web Image Dataset Construction

Form this chapter, the matching of store logos with signage in relevant Web images is discussed. Since it requires a target image dataset. An automatic target image dataset generation method from Web images is proposed. The Web images include Web-search images and Web page images, which are obtained from image search engines and the store's official Websites, respectively. This chapter introduces the extraction of signage patches from Web-search images (Section 5.1), and store logos from Web-page images (Section 5.2).

The terminologies are defined in Table 5.1. The pipeline of the proposed method is shown in Figure 5.1. A store list including store names and URLs is required as input. The store list is easily obtained from the Website of a mall or current geographic information service such as Google Places [83]. A search engine crawler downloads *Web Images* from Google Images for each store using their store names as keywords. Each *Web Image* is then assigned a store ID. These raw *Web images* may include many irrelevant images such as those of their product images. The facade image extraction process attempts to extract only the *Facade Images* from the original Web-search images. Then, signage extraction is applied to crop out *Signage Patches* from *Facade Images*. At the same time, a Web page crawler downloads all *Web Page Images* from each stores official Website. The *Web Page Logos* are extracted from a mass of *Web Page Images*. The signage patches and Web page logos are combined to generate the target signage dataset.

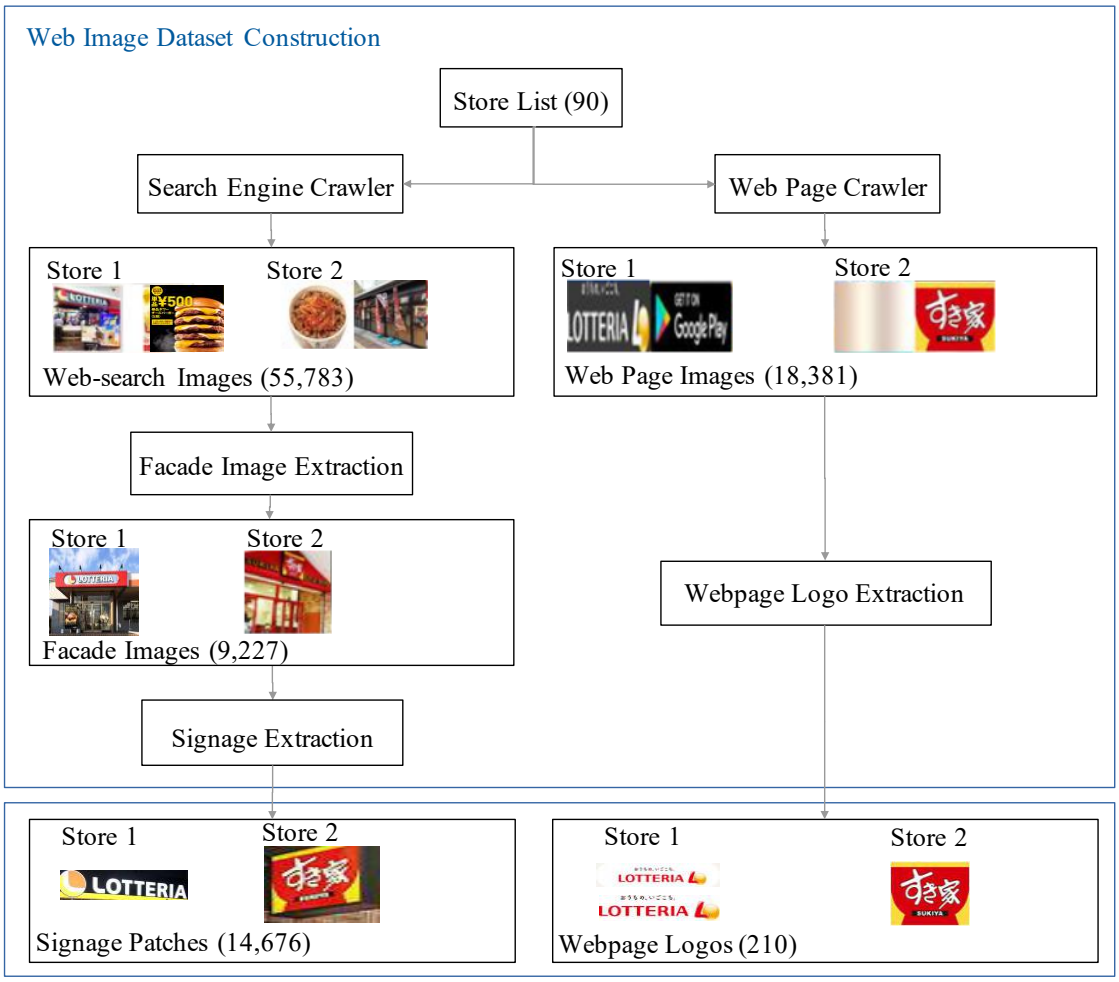


Figure 5.1: Pipeline of the Web image dataset construction. The method automatically collects data from two Internet resources: the search engine and stores' official Websites using Web-mining to construct a Web image dataset as the target dataset.

Table 5.1: Terminology used in the signage identification.

Noun	Description
Street Panoramic Images	The original panoramic images shot from the shopping street without store ID labeling.
Street Image	Image extracted from the Street Panoramic Image. These contain Signage Patches that need to be identified with store IDs.
Web Images	Images collected from the Web, which include both Web-search Images and Web page Images.
Facade Images	Images that show the facade of a store. Street images by definition all meet this criterion. However, the term Facade Images when used in the context of this work refers specifically to the subset of Web-search Images that also meet this criterion.
Signage Patches	The portion cropped from Street Images or Facade Images that has the stores signage on it.
Web Page Images	Images extracted from store's official Web pages.
Web Page Logos	A subset of Web Page Images that contains the representing logo of a store or company that owns the Website.

5.1 Signage Patch Extraction from Web-search Images

Web-search images are downloaded from a search engine by a crawler (Section 5.1.1). From them, uninformative Web-search images, such as images of products sold by the store, must be deleted. Actually, only the facade images, which most likely contains the stores signage is selected (Section 5.1.2). After that, signage extraction (Section 5.1.3) is applied to crop signage patches from the facade images.

5.1.1 Web-search Image Crawler

A search engine crawler automatically downloads Web-search images using target queries. Icrawler [87] is used to download Web-search images from Googles image search engine. In regard to the choice of queries, Zamir et al.s method [26] of appending queries such as 'store location' to the store names is applied to a larger number of matching images. Also, if a store name is represented in Japanese, its phonetic spelling (Romaji spelling) is added as an extended keyword using a morphological analysis tool (MeCab [88]). Note that also the sequence of search results is informative, since search engines rank the search results based on the correlation with the query. Therefore, search results near the top have higher relevance to the keyword. Effort should be made so that the relative sequences of

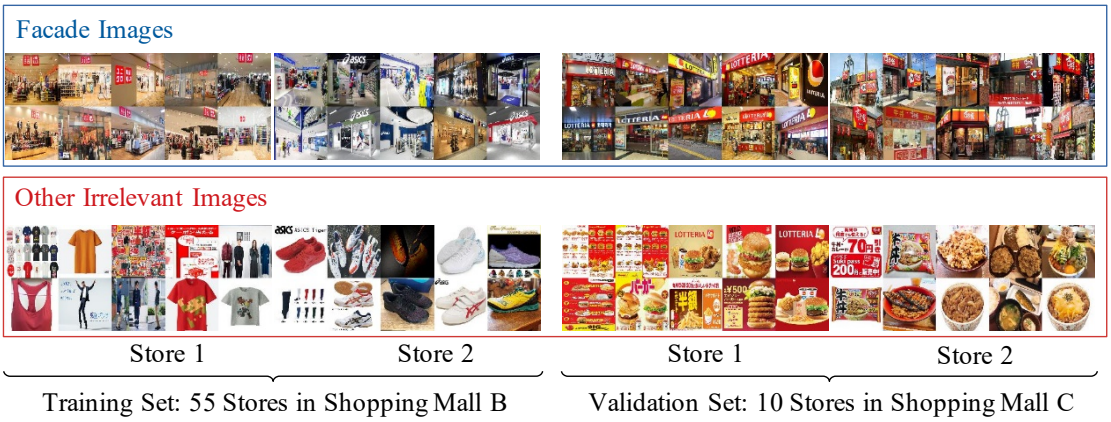


Figure 5.2: Examples of Web-search images. They are arranged to highlight the difference between the desired facade images and other irrelevant images.

all search results from all queries are preserved when those results are merged into one dataset.

In this work, the experiment is carried out in shopping mall C in Nagoya, Japan. There are 90 stores in shopping mall C, all of which have their names listed conveniently for accessing their Websites. The search results to the first 300 images larger than 200×200 pixels are downloaded. Since signage patches are cropped from the Web-search images under this condition, the size setting ensures the minimum resolutions. As a result, a total of 55,783 images are obtained from the 90 stores. The selection of the query can be considered as one of the factors that affects the number and the quality of images in the Web-search images dataset.

5.1.2 Facade Images Extraction

Figure 5.2 illustrates examples of Web-search images. Web-search images include both the useful facade images (the top portion of the figure) which most likely contain the store signage, and other images irrelevant to our task (the bottom portion of the figure). The facade images should be extracted, and the irrelevant images should be filtered out. For this, the proposed method adopts a general facade image classifier using a fine-tuned VGG16 [7] deep neural network. It is a binary classifier that predicts whether a Web-search image falls into the facade

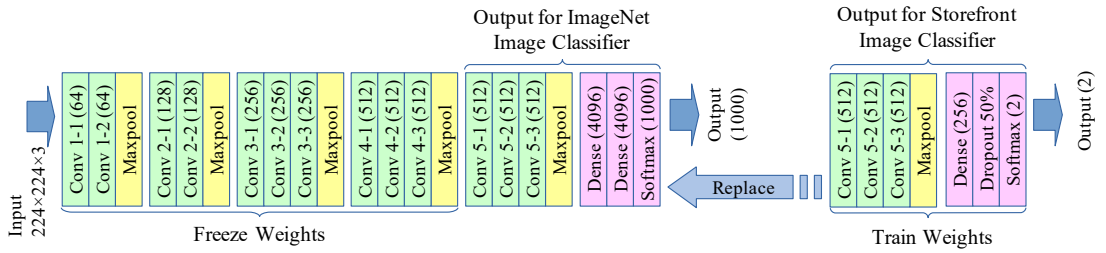


Figure 5.3: Network architecture for fine-tuning on VGG16. The weights of VGG16 are pre-trained by ImageNet. Layers from convolutional layer 5-1 to the output layer are by replaced without original architecture. The weights on the first four groups are frozen and the weights from convolutional layer 5-1 are trained using the proposed dataset.

image category or not. To ensure generality, the training dataset are taken from 55 stores in shopping mall B, which is another, unrelated shopping mall also in Nagoya. Note that all of these 55 stores do not overlap with stores in shopping mall C. First, Web-search images are crawled using these 55 store names as queries via the same method explained in Section 5.1.1. Then 20 facade images and 20 irrelevant images for each store are manually labelled. For validation purposes, 200 facade images and 200 irrelevant images belonging to 10 different stores from shopping mall C are selected. Figure 5.2 shows examples of the images in the training and validation sets.

The network architecture is shown in Figure 5.3. The weight values of the VGG16 network are pre-trained using the ImageNet dataset [89], which has a 1,000 neuron softmax layer corresponding to 1,000 class outputs. Transfer learning is performed using this pre-trained ImageNet. The last two dense layers and the last softmax layer are replaced into a 256 neurons dense (fully connected) layer, a 50% dropout layer, and a two neurons softmax layer. This adapts the output layer to correspond to the desired binary output: facade image or not (irrelevant images). During training, the first four out of VGG16s five convolutional layers are freezed and adaptations are only performed from the 5-1 layer to the output layer.

In the training stage, the batch size is set to 16 so that there are 138 iterations in one epoch. The performance of the facade image classifier is evaluated on the validation dataset based on its *precision*, *recall*, and F1-measure where F_1 =

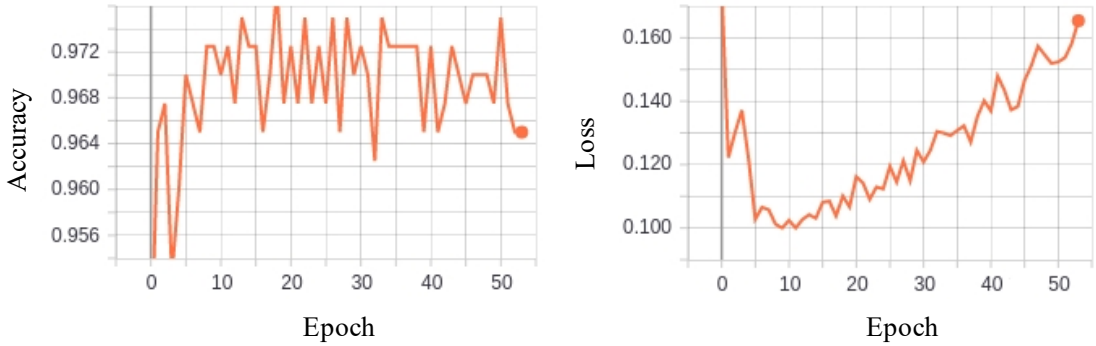


Figure 5.4: Validation accuracy (left) and the validation loss (right) of the facade classifier. The network converges at the 18-th epoch with a precision of 98.48%, recall of 97.00%, and F1-score of 97.73%. From the 19-th epoch on, over-fitting occurs, so the loss increases on the validation set.

$2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$. The validation results for each epoch are shown in Figure 5.4. The network converges at the 18 th epoch with a precision of 98.48%, recall of 97.00%, and F1-score of 97.73%. The classifier thus effectively filters out irrelevant images from the Web-search results.

In this way, the keywords selection mentioned in Section 5.1.1 is verified. The first 20 Web-search images from each store are crawled and facade images are selected by applying the classifier discussed in this section. As a result, the query 'store name' yields 604/1800 (33.56%) facade images and the query 'store name + shop' yields 772/1800 (42.89%) facade images. Stricter keywords can yield more accurate results but reduce their absolute number.

5.1.3 Signage Extraction

Not all areas on a facade image can be used to identify the store. Specifically, the stores signage is the only interest. Thus, signage patch extraction is applied to crop signage patches from facade images.

This task is similar to the saliency detection task [91, 92], which detects the human-focused part from an image or separating the foreground from the background. However, general saliency detection does not distinguish objects. For example, when saliency detector processes an image of a man who stands alone in

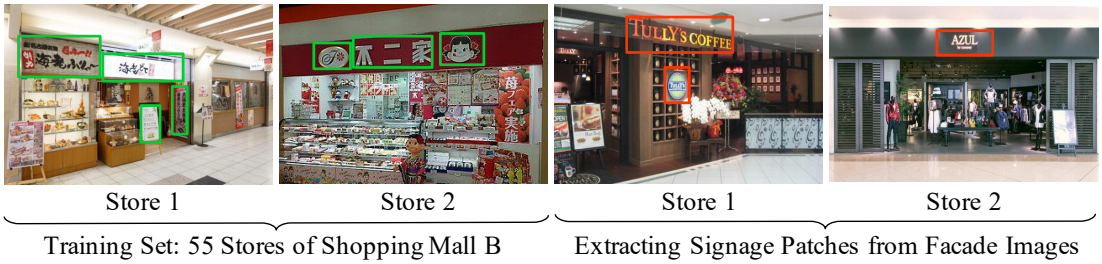


Figure 5.5: Examples from the 1,100 manually labeled facade images from shopping mall B (left two images). These are used as training data for YOLO. The right two images are examples of signage patches extracted from storefront images by the trained YOLO.

front of a store, the man may be extracted rather than the store signage, which is not suitable to our task. Meng et al. [29] suggested detection of traffic signs from images using Single Shot Multibox Detector (SSD) [85] as an object detection task. Here, a state-of-the-art object detector YOLOv2 [39] for fast store signage detection is used from facade images.

As proposed with the above-mentioned facade image classifier in Section 5.1.2, 1,100 facade images from shopping mall B that does not contain on overlapped store in shopping mall C are used for generality purposes. The example of facade images are shown in Figure 5.5. 1,063 bounding boxes for signage patches are manually annotated in those 1,100 facade images for training the YOLO. This training process also involves transfer learning. The initial YOLO network is pre-trained with the Pascal-VOC dataset [90]. A training with 82,000 batches with a batch size of 64 (4,800 epochs) are applied to the training set. The validation loss converges to less than 0.05.

The signage extraction is extracted based on precision, recall, F1-measure, and Intersection over Union (IOU). The IOU is defined as follows.

$$IOU(B, B') = \frac{\text{Overlap Area of } B' \text{ and } B}{\text{Union Area of } B' \text{ and } B} \tag{5.1}$$

where B is the manually annotated, correct bounding area while B' is the bounding area output by the detector. A prediction is considered correct when its IOU exceeds an arbitrary threshold θ. As shown in Figure 5.6, θ is set to 0.33 when

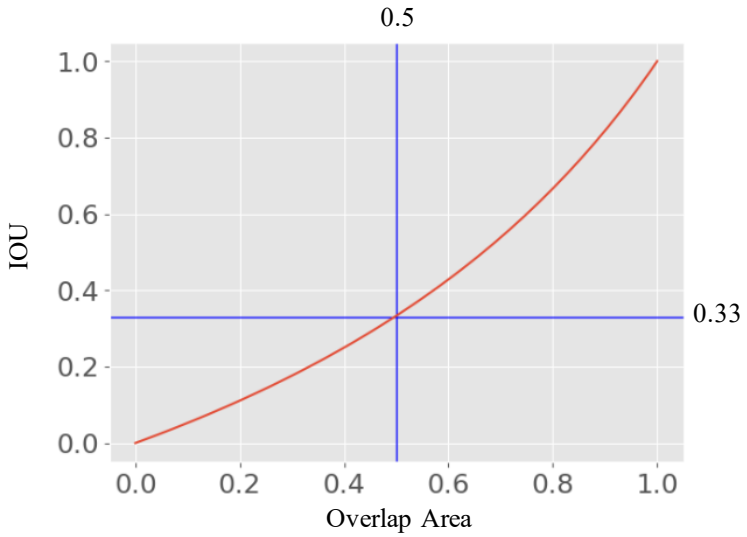


Figure 5.6: Relation between IOU and overlaps area. When overlap is 50%, the threshold of IOU is 0.33.

the overlapping area is 50%.

200 facade images from 10 stores in shopping mall A are manually annotated as a validation set. When YOLO predicts the signage patch, it outputs a confidence value in $[0, 1]$ for each bounding box. Results with a confidence value less than the threshold value are discarded. The experiment is iterated multiple times for each threshold value between 0 and 1, with an increase of 0.01. The results are evaluated based on precision, recall, F1-measure, and average IOU as shown in Figure 5.7. The optimal result of 87.45% precision, 76.42% recall, and 81.56% F1-measure are obtained when the confidence threshold is set to 0.17. However, a higher recall is desirable to obtain more samples for the signage dataset. Although this will also inevitably increase signal noise, the noise can be filtered out during the image matching phase as will be discussed later. A confidence threshold of 0.03 is settled where signage patches can be extracted with a precision of 77.12%, recall of 83.18%, and F1-measure of 80.04%.

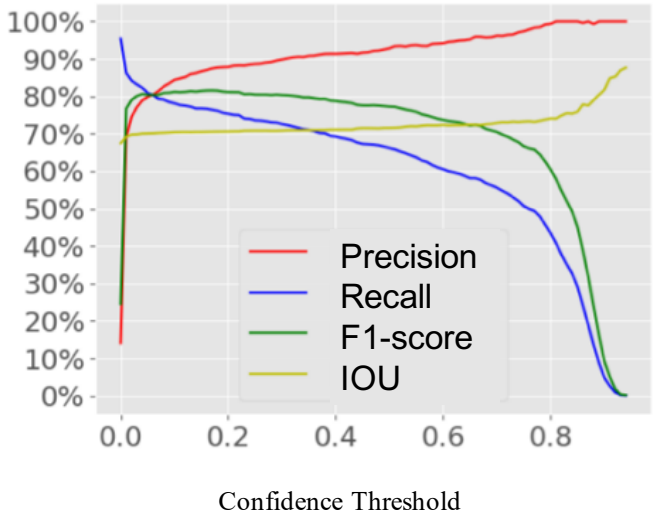


Figure 5.7: The performance of YOLOv2 on the validation set at different values of confidence thresholds.

5.1.4 Results of Signage Patch Extraction from Web-search Images

Figure 5.8 shows the resulting number of facade images and signage patches for each store in the target shopping mall C. The facade image extraction process effectively prunes 46,556 (83.46%) irrelevant images from the 55,783 Web-search images. The signage extraction process extracted 14,676 signage patches from the

Figure 5.8: Number of Web-search images (blue bars), facade images (red bars), and extracted signage patches (green bars). The data from each store sorted by number of signage patches from left to right. Some stores tend to yield more signage patches than others and the resulting image sets are significantly biased in size. The imbalanced dataset significantly affects the performance of a training-based image retrieval method. A feature point matching method was introduced to resolve this issue.

remaining 9,227 facade images. The graph is sorted in descending order of several signage patches, which ranges from 3 to 657, indicating a significant imbalance. It may be reasoned that this is caused by famous stores such as a food chain or a major bank being much better represented on the Internet compared to a relatively obscure local store. Another interesting phenomenon is that a Web search for a store providing food or services may yield more images related to its facade and signage in the first place.

In comparison, if the name of a store is also the name of a particular brand of goods or clothes, the most common Web-search images may yield more of its products rather than its facades. Fortunately, while these stores return fewer signage images, they usually also feature distinctive brand name logos on their Websites. Thus, Web page logo extraction is introduced for this issue.

5.2 Store Logo Extraction from Web Page Images

The Web page image crawler downloads Web-page images and their relevant data, including the image URL and hyperlink from the Web pages. The relevant data of Web-page images is represented as follows.

$$(src, link) \text{ in } I_{i,j} \text{ in } w_{j,k} \in W_k \tag{5.2}$$

where W_k is the Website of the k-th ($k \in [0, 73]$) store in shopping mall C, and $W_{j,k}$ represents the j-th Web page in W_k. Here, the top-page and up to ten sub-Webpages for each Website are crawled. Due to the different scale of each Website, this maximum setting can effectively control the crawling time. The ten sub-Webpages are linked by hyperlinks from the Web page, excluding off-site links. $I_{i,j}$ is the i-th Webpage image on the j-th sub-Webpage. The src is the source URL of $I_{i,j}$, which can uniquely identify a Web page image on the Web. The link is the hyperlink that the image is linked to.

The process for extracting $I_{i,j}$ from $W_{j,k}$ is as follows. An $< img >$ element in an HTML document must be an Web page image. Moreover, some Web page images are also hidden in the elements background image attribute defined in its Cascading Style Sheets (CSS). The crawler scans all nodes in the HTML document and extracts the background image if the node has one. In other cases, a Web page image might be a PNG image with a transparent background. It relies on the background color of its parent nodes to form part of the color scheme of the logo. Therefore, when the crawler finds an image type capable of having transparency, such as a PNG image, the crawler then scans all its parent nodes in the proper sequence to locate the one which has the same background color attribute in CSS and converts the image into JPG with the matching background color. The link of $I_{i,j}$ is extracted by scanning the image node and all its parent nodes in the proper sequence and finds the first hyperlink as the link of $I_{i,j}$.

5.2.1 Web Page Image Crawler

Webpage logos are a small subset of Webpage images. A Website may have only one or two logos but thousands of other images. In addition, each Website has its distinct structure. More traditional approaches such as manual or tag-based methods [51, 53] that rely on information specific to a particular site or a particular class of sites cannot always guarantee success. Therefore, a more general approach that is not affected by the Webpage structure and can effectively extract logos from almost all Websites is needed. The following features are observed regarding a logo.

- **Feature 1**: If a Web page image links to its top-page, it probably is a logo.
- **Feature 2**: If an image frequently appears in most of a Websites Web pages, then it probably is a logo.

Feature 1 usually applies to a logo located at the top left of most Web pages. When the user clicks this logo, the browser redirects he/she to the top-page of the Website. The same as facade image extraction, to ensure generality, Web page logo extraction is tested on 48 stores with official Websites in the non-overlapping shopping mall B. In shopping mall B, 44/48 (92%) of Website logos agree with this feature. By Feature 1 alone, 119 images from 13,091 Web-page images are extracted, including all the 44 stores logos.

For the remaining 4/48 Websites that disagree with Feature 1, the statistics-based Feature 2 is used to extract the logos. Via hyperlink mining, a Web crawler can extract Web pages from a node page to the whole Website [45]. Feature 2 is inspired by Li et al.'s approach [94], which sought to eliminate noisy elements from Websites. They note that noisy blocks usually share some common content and presentation styles. Therefore, the repeating elements in a Webpage are usually uninformative content or noise. While a Website's logo can hardly be considered noise, it does share the characteristics of being repetitive and uninformative, so the frequency of an images appearance in a Website could thus be used as a potential means to identify it is a logo or not. The frequency of a Web-page image is

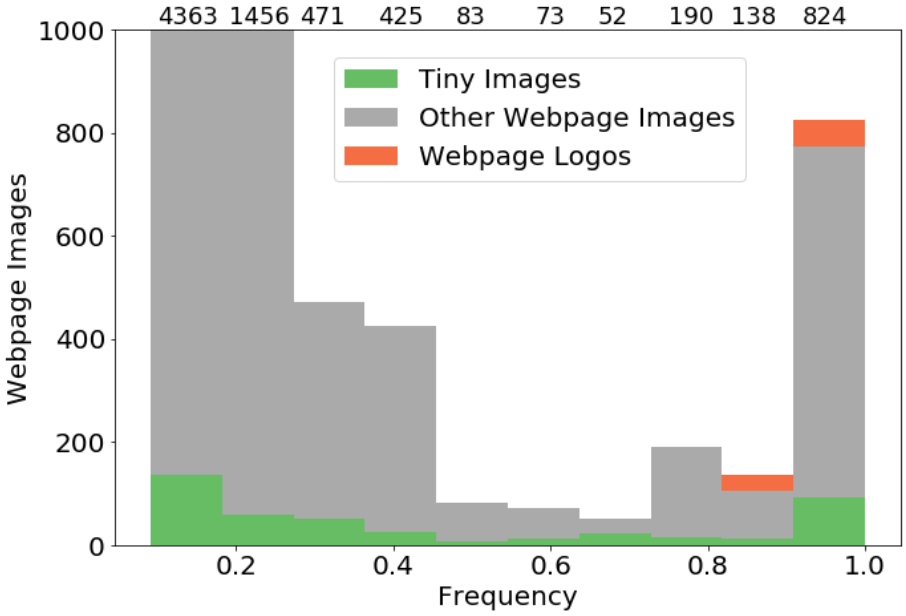


Figure 5.9: The histogram of Web page image frequency $f_{i,k}$ on 48 stores' Websites. The numbers on the top of each bin indicates the number of all Web images. Since most logos appear in high frequencies, they can be narrowed down simply by setting a threshold on $f_{i,k}$.

measured using the following formula:

$$f_{i,k} = \frac{\sum_{j=1}^{|W_k|} C(I_{i,j}, w_{j,k})}{|\{w_{j,k} : w_{j,k} \in W_k\}|}$$
$$C(I_{i,j}, w_{j,k}) = \begin{cases} 1 & (I_{i,j} \text{ in } w_{j,k}) \\ 0 & (I_{i,j} \text{ in } w_{j,k}) \end{cases} \tag{5.3}$$

where the frequency $f_{i,k}$ is calculated similar to Term Frequency (TF) [95]. $|w_{j,k} : w_{j,k} \in W_k|$ represents the number of Web pages $w_{j,k}$ in the Website W_k, and $\sum_{j=1}^{|W_k|} C(I_{i,j}, w_{j,k})$, the number of Web pages $w_{j,k}$, which include $I_{j,k}$. Here, function C returns 1 if the Web page image $I_{i,j}$ is in the Web page $w_{j,k}$, and otherwise returns 0. The range of frequency is $f_{i,k} \in (0, 1]$.

For determining the threshold on $f_{i,k}$, all 48 Websites and manually labeled

Figure 5.10: Style of Web page logos and irrelevant Web page images. The green bars in Figure 5.9 represent image elements such as a placeholder or a tiny button, which can sometimes occur with a frequency close to actual store logos. However, most are also significantly smaller than logos, so they can be easily filtered out based on size.

Web page logos in all the Web page images are investigated. Figure 5.9 shows the histogram of $f_{i,k}$ on all 48 Websites. Because of the large number of low-frequency images (4,363 images for 0.1 and 1,456 images for 0.2) that range from 0 to 0.2, the Y-axis is scaled into range from 0 to 1000. All Web page logos (the orange bars) have a frequency larger than 0.8. Therefore, a threshold $f_{\min} = 0.8$ is set for $f_{i,k}$ to extract Web page logos is set. For the four Websites which disagree with feature 1, 82 Web page images are extracted including all logos of the four Websites by feature 2. However, some noise, such as the tiny buttons and placeholders, are also extracted by Feature 2 as shown in Figure 5.10. Thus, a size threshold is set to filter out these irrelevant Web page images.

5.2.2 Using Favicon as Web Page Logos

A favicon (short for favorite icon) is a small rectangular image, which is displayed in the browser's address bar. This is usually the same as an Web page logo, so this could also be used. In shopping mall A, 47/74 (64%) Websites had unique favicons. The favicon is typically loaded via a fixed HTML tag, which can be accurately extracted in CSS selector such as a 'link[rel = shortcut icon]'. This rule-based extraction method can avoid noise. However, due to size limitation, a

Figure 5.11: Style of Signage, favicons, and Web images. The left row shows the example of signage in a natural scene. The middle row shows corresponding favicons. The right row shows the example of Web page logos extracted from Web page images. There are 31/47 (66%) such favicons that are not suitable as Web page logos in shopping mall A.

favicon always illustrates a general image of the Web page. As shown in Figure 5.11, abbreviated favicons are not suitable as Web page logos. Based on the above considerations, here, favicons were not adopted as the Web page logos.

5.2.3 Summary of Web Page Logos Extraction

The Web page logo extraction was applied to shopping mall C, which yielded 18,381 Web page images from 74 stores Websites that have their Web pages. 70/74 (95%) of Website logos agree with Feature 1, and 165 images were extracted including all 70 stores logos. For the 4/74 Websites which disagree with Feature 1, 45 Web page images including all logos of the four Websites were extracted by Feature 2. In the end, a total of 210 images were extracted, which includes all logos of the 74 stores at shopping mall C.

Chapter 6

Hybrid Image Matching

Once the target image database is obtained, the goal now becomes to correctly identify a query signage patch with its counterpart in the target image dataset so that it can be assigned a correct store ID. Figure 6.1 shows the pipeline of signage matching. However, at this stage, there exists many target signage. Directly matching each query signage to each image in the target signage dataset takes an unreasonable amount of processing time. Therefore, a Hybrid Image Matching (HIM) method is proposed, which creates an individual candidate matching dataset for each query signage. This narrows down the candidate matching images to only a relatively small number.

Conventional methods [7, 34, 37] apply feature vector techniques on signage identification. BoVW or deep learning-based image retrieval methods convert the query signage patches and target signage patches into fixed length feature vectors. They then match them based of the similarity on their feature vectors. Such a method allows the quick retrieval of thousands of images in a dataset.

However, compared to a manually created image dataset, the self-generated dataset through Web-mining contains more noise and imbalanced samples, which significantly affects the accuracy of image retrieval methods. For resolving this issue, the proposed HIM method replaces feature vectors with feature point matching, which matches a query signage patch with all target signage patches based on corresponding feature points. Such a feature point matching method can com-

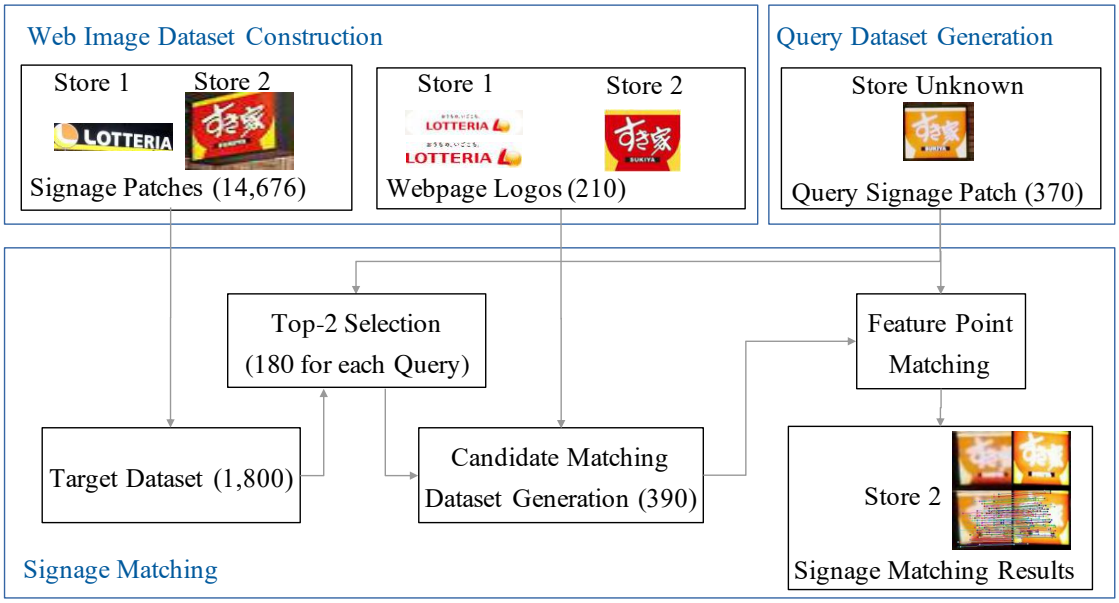


Figure 6.1: Pipeline of the signage matching. The goal is to identify and assign the correct store ID to each Query Signage by matching them with target signages. The numbers in the brackets represent the numbers of each object.

pensate for a noisy and unbalanced dataset because the correct query signage and target signage pair usually have more corresponding feature points than an incorrect one, regardless of image noise.

Computing time is the main concern when applying feature point matching on a large-scale. Matching query signage with all the thousands of target signage patches requires a long computing time. Therefore, the proposed method generates a candidate matching database, which is a smaller subset of all target signage patches, for each query signage. The above-mentioned feature vector method is well suited for this task because even under ideal circumstances the output from feature vector methods needs to undergo some form of further screening, as the correct pair does not always appear as the pair with the highest vector similarity score. These two methods, when used in tandem, serve well to complement each other's disadvantages. Since the method combines a deep learning approach with the feature point matching for signage identification, this combined method is named as the Hybrid Image Matching (HIM) method.

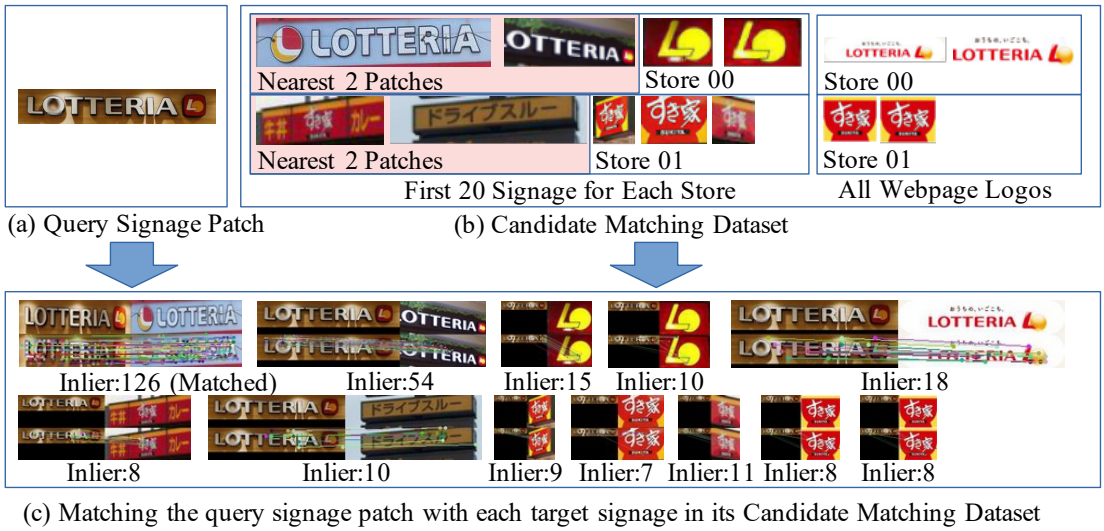


Figure 6.2: Pipeline of the Hybrid Image Matching (HIM) method. First, a candidate matching dataset (b) is generated for each query signage (a). Next, the query signage patch (a) is matched with each target signage patch in the candidate matching dataset, and then selects the inlier maximum computed by RANSAC (c).

6.1 Candidate Matching Dataset Generation

For each query signage, an individual candidate matching dataset is generated. As shown in Figure 6.2, the dataset consists of both signage patches (from Section 5.1) and Web page logos (from Section 5.2). For the signage patches, the first twenty signage images for each store are extracted. This is because search engines rank the search results based on their correlation with the query. This brings the total number of candidate target signage patches to 1,800 (20 signage patches from each store's Web-search images). Next, these signage patches, along with the query signage, are presented through RESNET50 [8]. This RESNET50 is pre-trained by ImageNet to convert each input image into an array of 2,048-dimensional feature vectors. The cosine similarity between two sets of feature vectors is highly correlated to the image similarity of their parent images. By selecting only the top two target signage patches from each store based on cosine similarity of their feature vector to the query signage, the number of candidate signage patches are narrowed down to just 180, which contains two signage patches from each store. As for the Web page logos, all 210 logos are inserted, which are extracted from 18,381

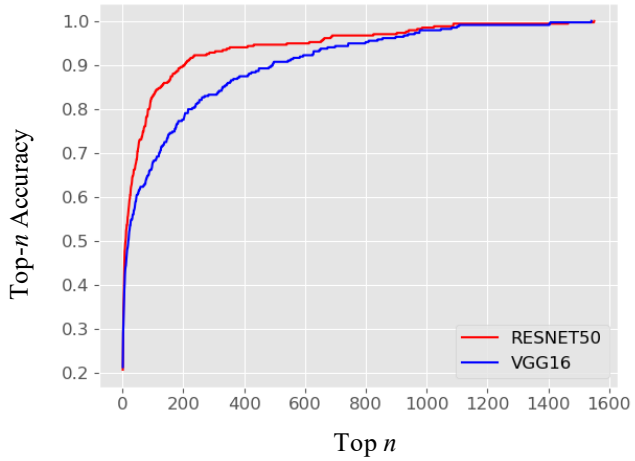


Figure 6.3: Top-n experiment to prove the RESNET50 feature is better for similar signage extraction.

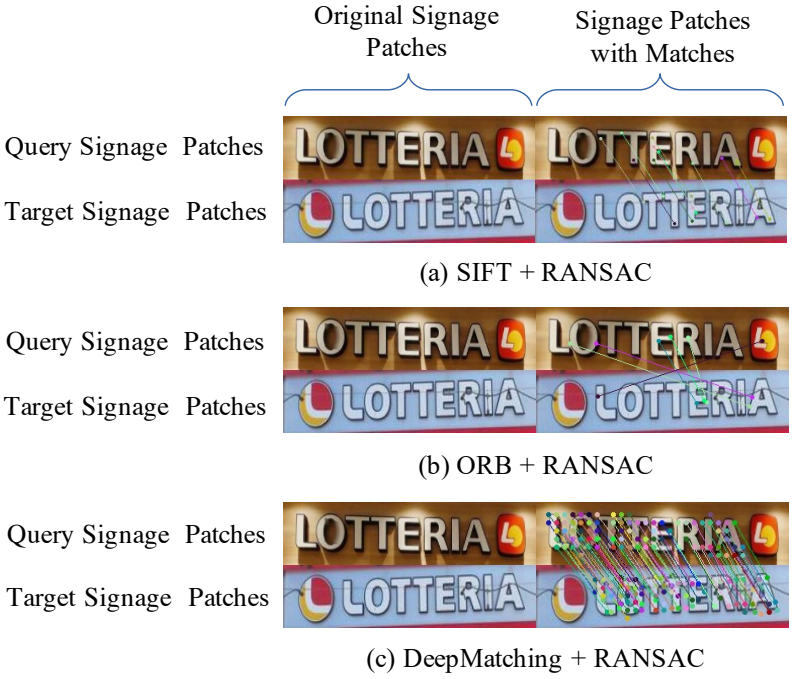


Figure 6.4: Example of SIFT matching, ORB matching, and DeepMatching on signage. The top image patch is the query signage. The bottom image patch is the target signage from Web-search image. The left patch shows the original images. The right patch shows the images with matches. The matches of SIFT is better than ORB. DeepMatching is better to handle image matching under the different conditions or domains.

Web images, with their store IDs into the candidate matching dataset. Therefore, for each query signage, the total number of image matching steps are reduced to a maximum of 390, which represents reasonable computing time.

As for the reason for choosing RESNET50 instead of VGG16 in Section 5.1, the top-n experiment shows that the RESNET50 feature allows us to extract more similar signage with a smaller n. Figure 6.3 shows the result of the top-n experiment using VGG16 feature and RESNET50 feature. The ground-truth that maps 370 query signage patches to 1,800 candidate target signage patches with corresponding store IDs are manually created. For each query signage patch, all 1,800 target signage patches are sorted according to the cosine similarities in RESNET50 feature and VGG16 feature. From the top-n similarities, if correct $m(m \leqslant n)$ correspondences are obtained, $m/370$ is calculated as the accuracy. As a result, when $n = 1$, the accuracy values of 27.74% in RESNET50 and 29.04% in VGG16 are obtained. When $n = 10$, 50.00% in RESNET50 and 44.01% in VGG16 are obtained. When $n = 100$, 82.93% in RESNET50 and 67.96% in VGG16 are obtained. When $n = 200$, 89.82% in RESNET50 and 77.54% in VGG16 are obtained. Therefore, when the top-n (n is set to 2 in this thesis) is set, the RESNET50 feature is easier to get more similar signage.

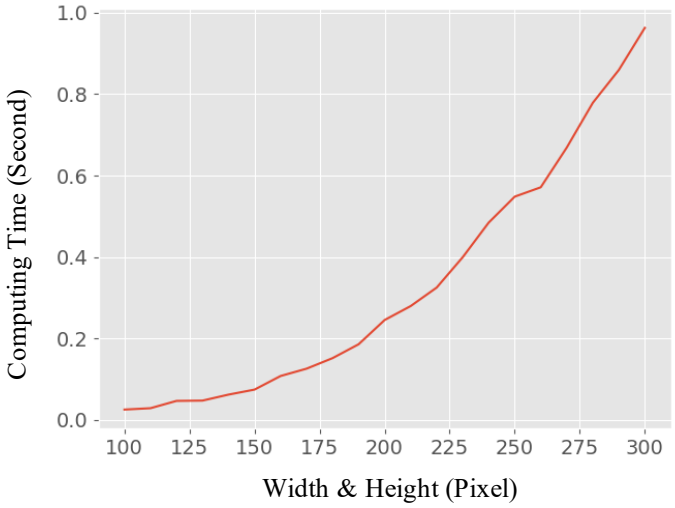


Figure 6.5: Computing time for various image sizes. Two square images of the same size gradually increasing their dimensions are matched. The computing time increases quadratically with the increase in dimension size.

6.2 Signage Matching

All the target images in the candidate matching dataset are already labeled with their corresponding store IDs. By correctly matching a query signage patch with the appropriate target signage patches, the correct store ID for the query signage can be obtained.

The matching algorithm extracts common visual feature points between the query signage and the target signage. Conventional local image features, such as SIFT [40] and ORB [82], extract feature points with descriptors, which are represented by fixed vectors, independent of the source image or the target image. The feature point matching is then performed by computing the distance of these descriptors. Because of this independent feature point extraction, traditional matching algorithms ignore the transformations and deformations from the source image to the target image. This may work well on images shot in the same scene or domains, such as in an image mosaicking task. However, in this work, the query signage and the target signage are mostly shot from different scenes. In addition, the signage shot by a camera and the Web page logo synthesized as

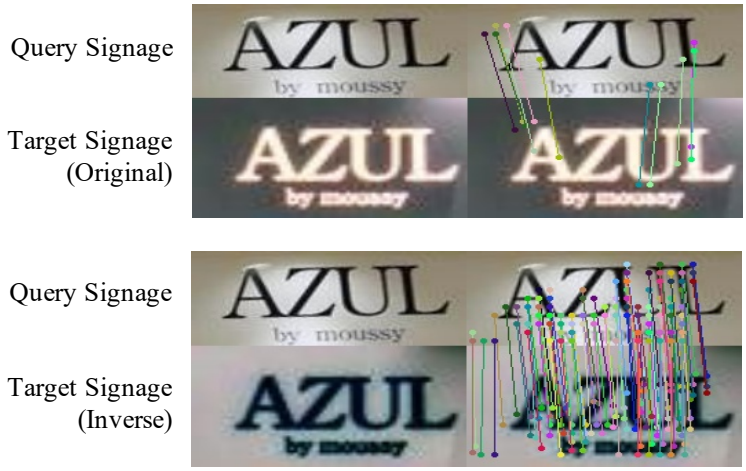


Figure 6.6: Matching query image with original and inverse target images. Dark character with bright background cannot normally be matched with a bright character with dark background but can be matched with the inverse image of it.

Computer Graphics (CG) belong to different domains. The image features may thus be different even from the same locations.

Here, DeepMatching [41] is used to match a query signage with its candidate matching dataset as shown in Figure 6.5. DeepMatching is a matching 'in the wild' algorithm that explicitly handles non-rigid deformations through bounds on deformation tolerance. For example, in Figure 6.4, SIFT and ORB matching are not result to images from different domains or different shooting conditions, while DeepMatching sidesteps these concerns.

The challenge in applying DeepMatching is its computing time. It computes a correlation pyramid in convolution for each non-overlapping 4×4 atomic patches in the entire image. The larger an image is, the more of these patches it will have to process, thus increasing computing time. Specifically, the computing time increases quadratically with respect to the width (W) and height (H) of an image. As shown in Figure 6.5, matching two square images of 300×300 pixels each will take about 1 second. If those two images are only 200×200 pixels, then it would only take about 0.2 seconds. Since, restricting the size of images can effectively

reduce computing time. The following resizing strategy is applied.

$$W = \begin{cases} 200 & W > H \\ 200 \times \frac{W}{H} & W < H \end{cases}$$
$$H = \begin{cases} 200 & H > W \\ 200 \times \frac{H}{W} & H < W \end{cases} \tag{6.1}$$

where W the width is resized and H is the height of an image. The larger of either W or H is resized to 200 pixels, and then scale the short side accordingly. It ensures 25 atomic grids along the long side. If, however, this results in the short side is reduced to less than 32 pixels, the bottom levels in the correlation pyramids may cease to be discriminative, where DeepMatching will not output any match. Therefore the scaling of the short side is restricted to a minimum of 64 pixels.

Furthermore, because DeepMatching computes the descriptor as a histogram of oriented gradients, a dark character with bright background cannot be matched with a bright character with dark background, as shown in Figure 6.6, because although they have the same contours, their gradient orientation is reversed. Therefore, the query signage is matched with both the original and its inverse of the target signage patches. The results are saved independently along with the rest of the candidate results. This homography can be computed by RANSAC according to corresponding matching points. If a matching point agrees with its homography, it is considered as an inlier point. The correct image should have a higher number of inlier points than incorrect images. Therefore, for each query image, all its target signage matching results are sorted based on the number of inlier points and only the first one is extracted as the correct target signage. Then, that signages store ID is assigned to the query signage. Thus, achieve the initial goal of assigning a store ID to all signage patches in the street images.

6.3 Evaluation Experiment

6.3.1 Accuracy Evaluation

In this section the accuracy of the proposed method is evaluated. Figure 6.7 illustrates a complete example of the signage identification method in action. Two image patches ('LOTTERIA' and 'SUKIYA') are cropped out from street images as query images. The 'LOTTERIA' patch is matched with a similar looking signage patch, which is separated and cropped it to a small size from thousands of other Web search images. The 'SUKIYA' image is matched with the official 'SUKIYA' logo, which proposed logo extraction method effectively extract from its Website.

The query signage images are generated from street panoramic images as shown in Figure 6.8. Panoramic images are collected by a NavVis M3 Trolley [32]. The NavVis M3 Trolley is a fully comprehensive indoor data capture device. Designed for frequently indoor scanning and mapping work, NavVis can automatically generate 3D indoor maps using built-in LiDAR and panorama camera. However, the POIs on the 3D indoor maps must be manually annotated in this application.

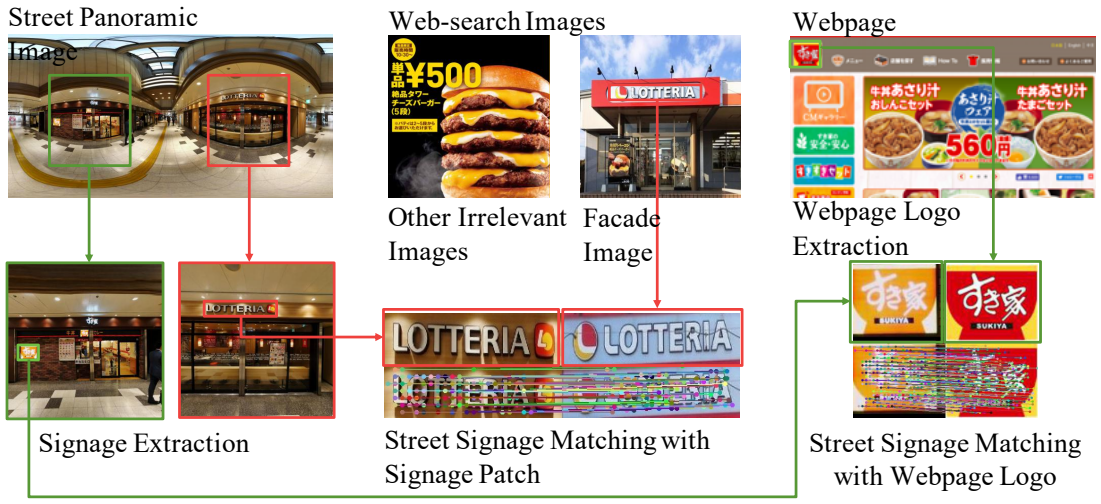


Figure 6.7: Example of the signage identification process. Two image patches ('LOTTERIA' and 'SUKIYA') are cropped out from street images as query images. The 'LOTTERIA' patch is matched with a similar looking signage patch, which is extracted and cropped from thousands of other Web search images. The 'SUKIYA' image is matched with the official 'SUKIYA' logo, which proposed logo extraction method had effectively extracted from its Website.

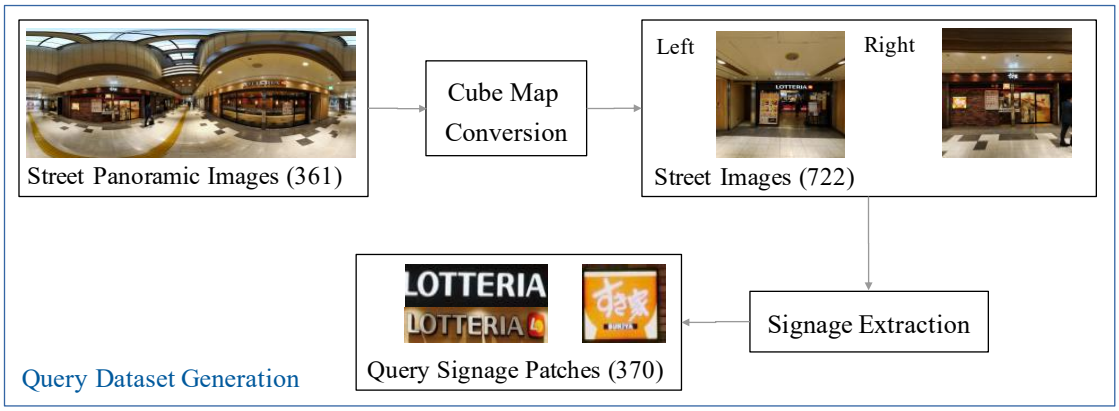


Figure 6.8: Pipeline of query generation.

A cube map conversion algorithm converts the street panoramic images into a cube-map of the six cardinal directions - left, right, front, back, top, and bottom. Because the shooting direction of NavVis is parallel to the passage, only the left and right images are used, which face the facade, as query street images. Using the same signage extraction, query signage patches are cropped from the street images. All the 370 query signage patches extracted from panoramic images are proposed and the resulting list of matched store IDs are compared with a manually-produced ground truth to verify its accuracy.

The candidate matching dataset for each query signage contains 180 signage patches (two signage patches from each store) extracted from 55,783 Web-search images and 210 Web logos extracted from 18,381 Web images.

Table 6.1 presents the results of the accuracy evaluation. The proposed method could correctly match a total of 337 signage patches out of 370 from 90 stores, achieving an accuracy of 91.1%. When only the Web page logos are used as the target signage set, the result is 182 correct matches (57.0%) of 321 signage patches from 74 stores that have a Websites. On the other hand, if only Web-search images are used on the same 321 signage set, the correct matches are 266 (82.9%). Combining these two matches raises the overall correct matches to 298 (92.8%). This illustrates the benefits of combining multiple different sources of sample images for improving the overall matching result.

Table 6.1: The evaluation result on signage identification. The numbers in the brackets represent the accuracy on signage in each pattern. The numbers out of the brackets represent the number of correctly predicted signage patches.

Query Signage / Target Signage	321 Signage Patches of 74 Stores with Websites	49 Signage Patches of 16 Stores without Websites	370 Signage Patches of All 90 Stores
Web-page Logos from Stores' Websites	182 Signage Patches (57%)	–	–
Signage Patches from Web-Search Images	266 Signage Patches (83%)	39 Signage Patches (80%)	305 Signage Patches (82%)
All	298 Signage Patches (93%)	39 Signage Patches (80%)	337 Signage Patches (91%)

Table 6.2: The evaluation result on store identification.

Query Signage / Target Signage	74 Stores with Websites	16 Stores without Websites	All 90 Store
Web Page Logos from Stores' Websites	40 Stores (54%)	–	–
Signage Patches from Web-Search Images	63 Stores (85%)	12 Stores (75%)	75 Stores (83%)
All	71 Stores (96%)	12 Stores (75%)	83 Stores (92%)

Figure 6.9: Examples of incorrect identification.

This method can also be used in annotating POIs with store IDs on the 3D maps established by NavVis. NavVis can automatically establish the 3D structure of a shopping mall, as long as the locations and directions of panoramic images are known. For a given store, if more than half of its signage patches can be correctly identified, the ID of the store can be considered to be correctly predicted. The result is presented in Table 6.2 where 83 of the total 90 stores are correctly identified by the proposed method with an accuracy of 92.2%.

There are two patterns that cannot be correctly identified in this experiment. Firstly, if the query store cannot find any online resource, including the Web search image and Web page logo, it cannot be identified. For example, as shown in the left part of Figure 6.9, there is a local store '10-FEET' without their own Website, therefore, the system cannot obtain any Web page logo of it. In addition, the name of '10-FEET' is covered by a famous rock band, so that the search results from Google are all about the band instead of the store facade image. In this pattern, the store without any online resource cannot be identified. A more detailed discussion of this can be found in Section 7.2.1. Secondly, if the quality of query signage is very low, it cannot be matched with target images. For the protection of privacy, NavVis automatically adds the mosaics to human faces in street images. As shown

in the right part of Figure 6.9, some of the signage patches are also mis-recognized as human face and mosaicsed, so the quality of the image is affected badly. In this pattern, the signage cannot be correctly identified by the proposed method. With improvement in image quality, the issue in the second pattern described above can be avoided.

6.3.2 Comparison with Other Algorithms and Evaluation of Individual Steps

A comparative experiment should be performed based on a common dataset, but there currently is no available open dataset for evaluating store signage identification. Therefore, a comparative experiment with the same dataset from shopping mall C is conducted, using methods similar to existing approaches and compared the results with the proposed method. Additionally, the effectiveness of individual steps of the method is evaluated. Detailed experimental results are shown in Table 6.3. The proposed method #9 is compared with other methods #1 to #6. In Experiment #1, a textual-matching based signage identification method on English street signage dataset using Tesseract OCR engine [73], which is used in approach [71] is applied. #2 to #4 verify image classification methods using BoVW [34, 37] and VGG16 fine-tuning [7], to classify all 370 query street signage patches into store IDs. The classifiers are trained by the original 'noisy' Web image dataset and manual 'clean' Web image dataset, which are crawled from the Internet, respectively. Meanwhile, #6 to #9 evaluates the individual steps of the proposed. By comparing #6 and #9, we can see the effectiveness of facade image extraction. #7, #8, and #9 compares the ORB [82], SIFT [40], and DeepMatching [41] as the image features, respectively. The result shows that the DeepMatching is suitable for matching street images and Web images.

- **Experiment #1: OCR-based Methods**

OCR approaches [73] are currently the most widely-used for signage identification. If an image frequently appears in most of Website's Web pages, then it probably is a logo. OCR engines are configured to work in specific language groups.

Table 6.3: Comparison evaluation results of signage identification.

#	**Identification Method**	**Query Signage Dataset**	**Training Signage Dataset**	**OCR Tool or Classification Model**	**Acc.**
1	Textual Matching	302 English Signage Patches	–	Tesseract OCR Engine [73]	29%
2	Image Classification	All 370 Signage Patches	Original 'Noisy' Web Image Dataset	BoVW [37]	39%
3				VGG16 Fine-tuning [7]	55%
4			Manual 'Clean' Web Image Dataset	BoVW [37]	47%
5				VGG16 Fine-tuning [7]	64%
			Target Dataset Generation Steps	**Utilized Matching Method**	
6	Proposed	All 370 Signage Patches	Without Facade Image Extraction	DeepMatching [41]	84%
7			With Facade Image Extraction	ORB [82]	48%
8				SIFT [82]	64%
9				DeepMatching [41]	**91%**

In this thesis, applying the TESSERACT [99] OCR engine on the English facade signages referencing [71], texts on 29% of them were correctly recognized. Figure 6.10 shows several signage patches with special fonts and in multiple languages, which significantly affect the performance of an OCR engine. Thus, the application of OCR-based methods is restricted, as they only operate properly within a rigidly defined environment.

- **Experiments #2 to #5: Image Classification-based Methods**

These methods all require a training set. The top 20 search results per store and all 210 Webpage logos are prepared as for the first training set to simulate the noisy environment prior to the candidate matching dataset generation process discussed. Next, signage patches and Web page logos are manually cleared, and 10 target signage patches for each store are used as the second training dataset as the 'clean' dataset. Note that, if a store has less than 10 'clean' target signage patches, the training images are over sampled to 10 images, as suggested by Paulina and

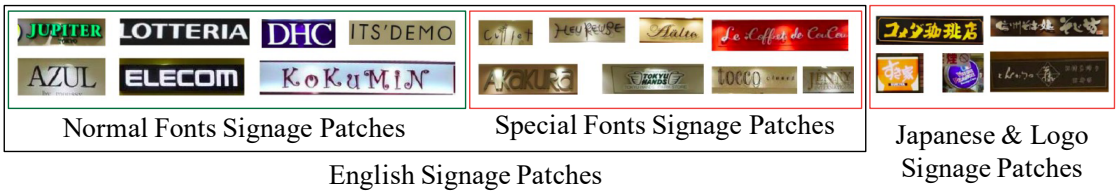


Figure 6.10: Applying OCR-engine TESSERACT on query facade signages. The OCR engine can recognize normal English fonts as shown on the left. Special fonts (middle) and multi-language characters cannot be recognized correctly.

David et al. [97]. This improved dataset simulates the situation after the candidate matching dataset has been generated. The two training datasets are supplied to both the BoVW [34, 37] and VGG16 [7] fine-tuning schemes and their accuracies are evaluated.

For BoVW Experiments #2 and #4, SIFT descriptors are extracted from query signage patches and target signage patches. They are then clustered by k-Means into 2,000 clusters for visual words extraction. Then, visual word histogram for each signage patch is computed. The store IDs of query signage patches are predicted by an SVM [98], which is trained by BoVW histogram of the target signage set. The BoVW model achieved an accuracy of 39% with the 'noisy' dataset and an accuracy of 47% with the 'clean' dataset.

For VGG16 fine-tuning Experiments #3 and #5, the same network structure as shown in Figure 5.3 is used, except for the output softmax layer that is set to 90 neurons to fit to the 90 stores in shopping mall A. The weights are initialized by ImageNet. The VGG16 fine-tuning achieved an accuracy of 55% with the 'noisy' dataset and an accuracy of 64% with the 'clean' dataset.

By observing the pairs of Experiments #2 and #4, and Experiments #3 and #5, the noise from Web-mining results reduces the accuracy by 8%. Other than noise, a lack of sufficiently balanced training datasets can also adversely affect traditional signage identification methods. However, noise and a lack of quality data is to be expected when dealing with automated Web mining. Compared with Experiment #9, the proposed HIM method could tackle these problems effectively.

- **Experiments #6 to #9: Effectiveness of Individual Steps**

The effectiveness of individual step is verified. Experiments #6 and #9 evaluate the effectiveness of the facade image extraction process discussed in Section 5.1.2. In Experiment #6, the facade image extraction is skipped and the signage extraction extracts 58,923 signage patches from the 55,783 Web-search images. Note that only the first 100 signage patches are used for each store. The RESNET50 feature is used to find the two most similar signage patches for a query street signage to generate the candidate matching dataset. Without facade image extraction, the noise in the target dataset increases, and the accuracy drops to 84%. We can see here that the facade image extraction process effectively filters out irrelevant images from a large set of Web-search images. Applying facade image extraction improves the proposed method by almost 7%.

Experiments #7, #8, and #9 evaluate the effectiveness of different image features used in the proposed method. In Experiment #6 with ORB [82], the best matches are found from the top 70% of the nearest pairs between feature point descriptors extracted from query images and target images. In Experiment #7 with SIFT [40], the ratio recommended in Lowes paper (0.7) [40] is used as a threshold to find the best matches. As a result, SIFT proves to be about 15% more accurate than ORB. However, in the task considered here, the scenes and Web images are mostly obtained from different views or belong to different domains. This significantly affects effective feature extraction. This issue is resolved by using DeepMatching [41] in the proposed method.

6.4 Open Source: Hybrid Image Matching Algorithm

The source code of the proposed Hybrid Image Matching (HIM) algorithm is published as an open source code on GitHub [109]. TensorFlow [110] with Keras [111] is used in the backend to extract the image feature vector to establish the candidate matching dataset. DeepMatching [41] is used for feature point extraction and matching. As shown in Figure 6.11, the system requires un-labeled query images and labeled target images as input. The query images are stored in a directory without distinction. Target images are stored in directories of their respective categories. Although the program accepts noisy and unbalanced target image dataset, in this example, an ideal state without noise is demonstrated. After the matching process, the program outputs the number of inlier-matches for each query image on the top n target image. Users can use the top one as the category of the query image.

For testing the performance, the algorithm is implemented on a PC with Intel Core i7-7700HQ 2.80GHz CPU and two DDR4 2400MHz 16GB memories. Both query image and target images are resized so that the long edge becomes 200 pixels. Matching one query image with 180 target images in 90 classes on both normal and inverse images, takes about 100 seconds. Although the calculation time is acceptable considering the high accuracy of the image matching, it does not support real-time processing at this stage. As the fastest target recognition algorithm in the industry, YOLO v2 [38, 39] supports object detection in about 40 FPS. Thus, future work should focus on increasing speed while ensuring the high matching accuracy.

Figure 6.11: Example of inputs and outputs of the Hybrid Image Matching algorithm.

6.5 Summary

In this chapter, a Hybrid Image Matching (HIM) method was proposed for signage matching using Web-crawled dataset. It combines the deep learning approach with the feature point matching for signage identification. It can effectively cope with noise, imbalance of data from Web-crawled datasets and achieves 91% accuracy in experiments. Without the help of Optical Character Recognition, it can also handle arbitrary signage designs. Because there is no specialized training involved, the same process should also work for any other location without manual adjustment.

A limitation to the HIM algorithm is that it cannot give a confidence score to the matching results. The algorithm must output a class label ranking for each query signage according to inlier points. Due to this, since it is difficult to normalize the numbers of inlier points, it cannot judge the situation that the query signage cannot be matched by setting a threshold on confidence.

Chapter 7

Conclusion and Future Work

In this chapter, the works introduced in this thesis are summarized. The contributions in this thesis and the improvements from related approaches are high-lighted. In addition, the shortcomings that can be foreseen in the course of this study or the work not done at this stage are pointed out, which should provide inspiration for future work. Frankly speaking, the proposed methods are not way-size-fits-all and have a certain scope of application. This is to remind readers of the environment and conditions that should be mentioned when applying similar methods.

7.1 Conclusion

7.1.1 Nearby Event Data Extraction

This thesis proposed a scheme for extracting meaningful data from Web pages. As a specific case for this, a demonstration for nearby Web page event extraction is introduced. Three key technologies were proposed and evaluated for this. First, for converting semi-structured Web documents into processable structured data, a Web page segmentation algorithm was proposed. Second, for event data extraction, a textual classifier was proposed. Third, a real-world application Event.Locky was implemented based on these methods and theories. The Event.Locky application extracts event data from nearby stores' Web pages and displays on users' mobile devices as spatial-temporal data. It allows to easily collect and reuse event data from store' Web pages in a geographical area.

Through an experiment involving Web pages of 96 shops at shopping mall B and C in Nagoya, Japan, the proposed event record extraction method achieved an F1-score of 91.61%, an increase of 3.07% from current methods. The combinations of event extraction algorithm and Web page segmentation algorithm achieved an F1-score of 83.96% to extract event records from Web pages, an increase of 1.6% from the current algorithm.

7.1.2 Signage Identification Using Web-crawler Information

This thesis proposed an image-matching based automatic signage identification method using Web-crawled information. In contrast to existing textual-matching based methods, this method is not limited to just text or character-based signage. It can process an arbitrary signage design without the help of an OCR process. In addition, the proposed method replaces manual data collection with an automated Web-mining and data cleansing process, which can self-generate entire matching databases from very simple inputs. By applying data pruning and filtering methods, it effectively reduces the number of images selected for matching and ensures that the process can be performed within reasonable computing time. The Hybrid

Image Matching (HIM) method, which combines a deep learning approach and a feature point matching approach for street image matching with Web image was proposed. It effectively avoids the impact of sample noise and imbalance on image recognition. HIM is also not dependent on specialized training, thus making it a highly portable system that should work at other locations with no fine tuning required.

7.2 Future Work

The effectiveness of Web mining techniques may depend on the country or region because a lot of stores, especially small services, publish profiles on social networks rather than on their own Websites. This is discussed in Section 7.2.1, which reminds readers that it is important to choose a specific method for a given situation. In Section 7.2.2 and Section 7.2.3, the shortcomings of the proposed event data extraction method and signage identification method are also mentioned. Some suggestive ideas are also proposed for future work.

7.2.1 Application Range of Web Data Mining

In this thesis, official Websites were selected as data sources to collect event data and images. This strategy is based on the consideration of geo-economic reasons and the behavior of Japanese business owners and customers.

In '2013 White Paper on Small and Medium Enterprises in Japan' [114], Figure

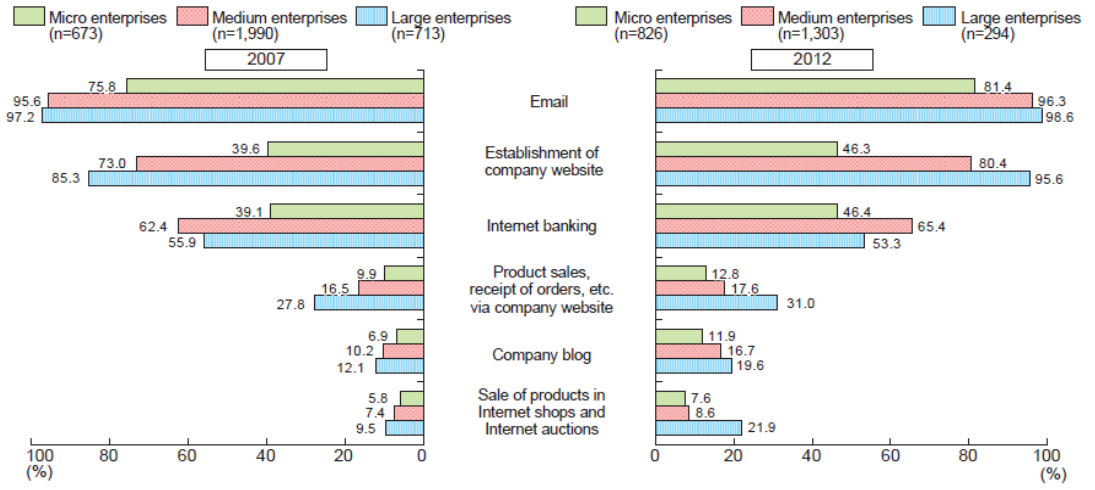


Figure 7.1: Status of introduction of IT by size of company and mode of use of technology (2007 and 2012.) [114].

7.1 shows a survey on the establishment of company Websites. It shows that 95.6% of large enterprises, 80.4% of medium enterprises, and 46.3% of microenterprises maintain their official Websites in 2012. Such a very high level of informatization benefits from the habits of Japanese business customs. In Japan, Website construction and the store decoration are of the same importance for store owners because the customers or investors are accustomed to obtaining official information through the store's Website. However, in other countries or region, lots of stores, especially small services have profiles on social networks then on their own Websites. In such cases, different Web sources and Web-mining strategy should be adopted. For example, Chinese store owners are more willing to use SNS or third-party e-commerce platforms, such as Weibo [1] (similar to Twitter), or Alibaba [2] (similar to eBay [3]), etc., to promote their stores rather than build their own Websites. So, if deploying a similar system in China mainland, it must give priority to SNS and third-party E-commerce platforms as Web resources.

On the other hand, the strategy is also limited by the experimental subject i.e. the shopping mall C. There are more international, famous, and chained brands in the given mall. Actually, in shopping mall C, there are only two local and new brands, where there was no Web search image or Web page logo. This lack causes the accuracy of signage identification dropped down to 91%. For such un-famous signage identification, the OCR approach which can read the text on the signage may be a good solution in the future. But the premise is that the OCR engine can handle cases involving characters with special font types, non-characters, and characters from multiple language groups. Thus, improvement of OCR engine is one important future work for signage identification.

In general, there is a significant difference in confidence between obtaining the stores official Website or user-uploaded images such as from SNS platform or Web search. Crawling logos from official Websites are promising, but it may be with low coverage. Crawling user-uploaded images can obtain more abundant resources, but

[1] Weibo: https://www.weibo.com/
[2] Alibaba: https://www.alibaba.com/
[3] eBay: https://www.ebay.com/

it may be noisy (some users may upload wrong or irrelevant images). Specialized advisory services are good solutions to balance advantages and disadvantages of these two aspects. Roshan et al. [37] used the advisory services. They 'use the APIs of Yellow pages [4] and Yelp [5] to retrieve and aggregate the nearby businesses automatically.' It can find out more accurate and abundant resources from the specialized advisory services rather than a comprehensive search engine such as the Google Image Search only if the data in the service can cover the region where the system demands.

7.2.2 Multimedia Event Data Extraction

Limitation of the nearby event data extraction study involves the multimedia extraction. The proposed event data extraction leverages Web mining techniques based on text classification. However, some event information on Web pages is presented as multimedia data (such as event images or videos of event advertisement). In this case, a text-based method cannot extract information. Combining image processing and deep learning with Web mining may be promising to address this limitation.

Same as VIPS, the proposed Web Page Segmentation algorithm can also reveals the relationship of the images and their surrounding texts in a Web page. We can see that by implementing the algorithm presented above, training data could be provided based on images and related texts at low cost. According to automatic generation of the description of images, it may support the above-mentioned event image extraction task.

Thus, for future work, multimedia event data extraction from Web pages and attempt to combine image processing techniques (such as deep learning and OCR) with Web mining should be investigated. Also, for non-Japanese speakers, machine translation techniques should be incorporated to help them obtain valuable event data.

[4] Yellow pages: https://www.yellowpages.com/
[5] Yelp: https://www.yelp.com/

7.2.3 Future Work of Signage Identification

An inherent limitation of the proposed signage identification method lies in its dependency on Web search results and Web page images for data. If a target store does not have a signage that features images available on the Web, then it cannot be identified by this method. Additional research should focus on methods other than Web-mining through search engines and Websites.

Another ideal scenario for image collection can be considered in the future work. An advisory service can be built, in which the store owners actively upload their facade images onto the platform as the finger-printing database. It is not necessary to manually annotate the signage because the proposed signage detector can automatically do so. The system also automatically constructs the target image database and identifies the signage. In this way, it can ensure both high confidence and low noise.

Another shortcoming of Hybrid Image Matching is that the algorithm dose not output a confidence score on each matching. Therefore, a threshold value to judge if an image cannot be matched in this stage cannot be set. In the future work, calculating a confidence score using the inlier numbers should be considered.

7.2.4 Summary

In this thesis, the concept of linking real-world information with Web resources is investigated. Two instances: extraction of Web event data using GPS information, and identification of store signage using street image were implemented to verify the feasibility. As a result, it shows that it is possible to use sensor data to retrieval Web data instead of conventional keyword search.

This may be one of the future directions for search engine techniques to evolve. In the age of conventional Internet, the keyword search was majority of Web data retrieval. Nowadays, with the development of the IoT and ubiquitous computing, more and more smart terminal devices are abandoning keyboards. Accordingly, search engine companies have tried to provide their 'hands-free' services. For example, Google proposed a Voice Search App [6], which frees the users typing keywords. They just say 'Okey Google' to activate the voice engine and say what they want to search. The search engine can convert the voice into text namely keywords and returns relevant Web data to user device. Products similar to this include Apple Siri [7], Microsoft's XiaoIce [115], etc. Although the core component of them is still based on keyword search techniques, it can be regarded as an instance of using sensor (microphone) information to retrieve Web data. Those dynamics of search engine companies hint they are considering this option seriously. In the future, more sensors, such as camera, LiDAR, etc. more scenes, such as VR, autopilot, etc. may join in this line. Considering more kinds of real-world data collected by sensors to link them with Web data may be a potential task with plenty of applications.

[6] Google Voice Search: https://play.google.com/store/apps/details?id=com.prometheusinteractive.voice_launcher
[7] Apple Siri: https://www.apple.com/siri/

References

[1] "RFC2616: Hypertext transfer protocol - HTTP/1.1" [online] Available: `https://tools.ietf.org/html/rfc2616` Accessed: Nov. 08, 2018.

[2] "RFC1866: Hypertext Markup Language - 2.0" [online] Available: `https://tools.ietf.org/html/rfc1866` Accessed: Nov. 08, 2018.

[3] "Googles search knows about over 130 trillion pages" [online] Available: `https://goo.gl/hd5aw3` Accessed: Dec. 21, 2018.

[4] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the Web," World Wide Web Internet and Web Information Systems. pp. 1-17, 1999.

[5] B. Liu, "Web Data Mining (Second Edition)." [Online] Available: `https://goo.gl/3LiVah` Accessed: Dec. 20, 2018

[6] Z. Wang, Y. Mei, and F. Yan, "A new Web image searching engine by using SIFT algorithm", Proceeding of the International Conference on Web Information Systems and Mining (WISM), pp. 366-370, 2009.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv:1409.1556, pp. 1-14, 2014.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proceeding of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016.

[9] A. Karpathy, and F.F. Li. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3128-3137, 2015.

[10] G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, and T. L. Berg. "Babytalk: Understanding and generating simple image descriptions," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), val. 35, no. 12, pp. 2891-2903, 2013.

[11] V. Abhishek, and H. Kartik. "Keyword Generation for Search Engine Advertising using Semantic Similarity between Terms." Proceedings of the 9-th International Conference on Electronic Commerce, pp.89-94, 2007.

[12] W. Wang, K. Yamakawa, K. Hiroi, K. Kaji, and N. Kawaguchi, "Velobug: A mobile system for 3D indoor mapping." Proceedings of 2015 Conference on Ubiquitous Computing (UbiComp), pp. 337-340, 2015.

[13] W. Wang, K. Sakurada, and N. Kawaguchi, "Reflectance intensity assisted automatic and accurate extrinsic calibration of 3D LiDAR and panoramic camera using a printed chessboard," Remote Sensing, vol. 9, pp. 1-24, 2017.

[14] J. Hays, and A. A. Efros, "IM2GPS: Estimating geographic information from a single image." Proceedings of 2009 Conference on Computer Vision and Pattern Recognition (CVPR 2009), pp.1-8, 2009.

[15] X. Qian, Y. Zhao, J. Han, "Image Location Estimation by Salient Region Matching." IEEE Transactions on Image Processing, vol. 1, pp.4348-4358, 2015.

[16] C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "An event data extraction method based on HTML structure analysis and machine learning," 2015 IEEE 39th Annual Computer Software and Applications Conference (COMPSAC), vol. 3, pp. 217-222. 2015.

[17] C. Liao, W. Wang, K. Sakurada, and N. Kawaguchi, "Image-matching based Identification of Store Signage using Web-Crawled Information," IEEE ACCESS, vol. 6, pp. 45590-45605. 2018.

[18] C. Liao, K. Hiroi, K. Kaji, K. Sakurada, and N. Kawaguchi, "Event. Locky: System of event data extraction from Webpages based on Web mining," Journal of Information Processing, vol. 25, pp. 321-330. 2017.

[19] C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "Development of event information summarization system based on spatio-temporal-information," Proceedings of 2014 DICOMO Multimedia, Distributed, Cooperative, and Mobile Symposium, pp. 657-665. 2014.

[20] "ATND." [Online]. Available: `https://atnd.org/` Accessed on: Dec. 22, 2018.

[21] "Peatix." [Online]. Available: `http://peatix.com/` Accessed on: Dec. 22, 2018.

[22] B. Hila, N. Mor, and G. Luis, "Beyond trending topics: Real-world event identification on twitter," proceeding of the 2011 International AAAI Conference on Web and Social Media (ICWSM), pp. 438-441, 2011.

[23] K. Watanabe, M. Ochi, and M. Okabe, "Jasmine: A real-time local-event detection system based on geolocation information propagated to microblogs," Proceeding of the 20-th ACM International Conference on Information and Knowledge Management (CIKM), pp.2541-2544, 2011.

[24] A. Hamed, S. Christian and G. Michael, "Eventweet: Online localized event detection from Twitter," Proceeding of the 2013 Very Large Data Bases (VLDB) Endowment, pp.1326-1329, 2013.

[25] S. Ichien, K. Kaji, K. Hiroi, and N. Kawaguchi, "Proposal of an open and utilization method for store POIs with linked open data", Proceedings of 2014 DICOMO Multimedia, Distributed, Cooperative, and Mobile Symposium, pp.1930-1938, 2014.

[26] R. Cristobal and V. Sebastian, "Educational data mining: A survey from 1995 to 2005," Expert Systems with Applications, vol.33, no.1, pp.135-146, 2007.

[27] A. Ahmed, L. Hong, and A. Smola. "Hierarchical geographical modeling of user locations from social media posts," Proceedings of the 22nd International Conference on World Wide Web. pp.25-35, 2013.

[28] A. Gupta, S.S. Anand, and C.R. Manjunath, "A comparative study on data extraction and its processes," International Journal of Applied Engineering Research, vol. 12, no. 18, pp. 7194-7201, 2017.

[29] A. Arasu and H. Garcia-Molina. "Extracting structured data from Web pages." Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp.337-367, 2003.

[30] Y. Sasaki. "The truth of the f-measure." Teach Tutor mater, vol.1, no. 5, pp.1-5, 2007.

[31] C. J. Jang, J. Y. Lee, J. W. Lee, and H. G. Cho, "Smart management system for digital photographs using temporal and spatial features with EXIF metadata." Digital Information Management (ICDIM), vol. 1, pp. 110-115, 2007.

[32] "NavVis." [Online]. Available: `http://www.navvis.com/` Accessed on: Dec 22, 2018.

[33] J.P. Donna, "It s about Time: A conceptual framework for the representation of temporal dynamics in geographic information systems," Annals of the Association of American Geographers, pp.441-461. 2009.

[34] H. Xu, D. Zhao, J. An, and L. Liu, "Indoor shop recognition via simple but efficient fingerprinting on smartphones," Proceeding of the 2016 International Conference on Cloud Computing and Intelligence Systems (CCIS), pp. 27-31, 2016.

[35] J. Song and Y. Park, "Fingerprint-based user positioning method using image data of single camera," Proceeding of the 2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN), pp. 13-16, 2015.

[36] Q. Yu, C. Szegedy, M. C. Stumpe, L. Yatziv, V. Shet, J. Ibarz, and S. Arnoud, "Large scale business discovery from street level imagery," arXiv:1512.05430, pp. 1-9, 2015.

[37] A. Roshan Zamir, A. Dehghan, and M. Shah, "Visual business recognition: A multimodal approach," Proceeding of the 2013 ACM International Conference on Multimedia, pp. 665-668, 2013.

[38] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," Proceedings of the 2016 IEEE Conference Computer Vision and Pattern Recognition (CVPR), pp. 779-788, 2016.

[39] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger" Proceeding of the 2016 IEEE Conference Computer Vision and Pattern Recognition (CVPR), pp. 6517-6525, 2016.

[40] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision (IJCV), vol. 60, no. 2, pp. 91-110, 2004.

[41] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "DeepMatching: Hierarchical deformable dense matching," International Journal of Computer Vision (IJCV), vol. 120, no. 3, pp. 300-323, 2016.

[42] M. A. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Communications of the ACM, vol. 24, no. 6, pp. 381-395, 1981.

[43] K. Norvag, R. Oyri. "News item extraction for text mining in web newspapers." Proceedings of the 2015 International Workshop on Challenges in Web Information Retrieval and Integration, pp.195-204, 2015.

[44] J. Jaafar, U. D. Kamaluddeen, and M. S. Liew. "Web intelligence: a fuzzy knowledge-based framework for the enhancement of querying and accessing Web data." Big Data: Concepts, Methodologies, Tools, and Applications. IGI Global, pp. 711-733, 2016.

[45] C. I. Wong, K. Y. Wong, K. W. Ng, W. Fan, and K. H. Yeung. "Design of a crawler for online social networks analysis," WSEAS Transactions on Communications, vol. 13, pp.263-274, 2014.

[46] J. Foley, M. Bendersky, and V. Josifovski, "Learning to extract local events from the web." Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 423-432, 2015.

[47] A. Ritter, O. Etzioni, and S. Clark. "Open domain event extraction from twitter," Proceedings of the 18th ACM Conference on Knowledge Discovery and Data Mining (KDD), pp.1104-1112, 2012.

[48] A. Magdy, A. M. Aly, M. F. Mokbel, S. Elnikety, Y. He, and S. Nath. "Mars: Real-time spatio-temporal queries on microblogs." Proceedings of the 2014 International Conference on Data Engineering, pp.1238-1241. 2014.

[49] C. S. Win, "Web page segmentation and informative content extraction for effective information retrieval," International Journal of Computer & Communication Engineering Research, vol. 2, pp.35-45, 2014.

[50] E. Ferrara, P. Meo, G. Fiumara, and R. Baumgartner, "Web data extraction, applications and techniques: A survey," arXiv:1207.0246v4, pp.301-323, 2014.

[51] M. Jussi, "Effective web data extraction with standard XML technologies," Computer Networks, vol.39, no.5, pp.635-644, 2002.

[52] B. Robert, F. Sergio, and G. Georg, "Visual web information extraction with LIXTO," Proceeding of the 2001 Very Large Data Bases (VLDB), pp.119-128, 2001.

[53] S. Lin, J. Ho, "Discovering informative content blocks from web documents," Proceeding of the 2002 ACM Conference on Knowledge Discovery and Data Mining (KDD), pp. 588-583, 2002.

[54] F. Crivellari, and M. Melucci, "Web document retrieval using passage retrieval, connectivity information, and automatic link weighted," Proceedings of the 9-th Text Retrieval Conference (TREC-9), pp. 11-18, 2000.

[55] M. Kovacevic, M. Diligenti, M. Gori, and V. Milutinovic, "Recognition of common areas in a web page using visual information: A possible application in a page classification," Proceedings of the 2002 IEEE International Conference on Data Mining, pp.250-257, 2002.

[56] D. Cai, S. Yu, J. Wen, and W. Ma, "VIPS: A vision-based page segmentation algorithm," Microsoft Technical Report, MSR-TR-2003-79, 2003.

[57] D. Cai, S. Yu, J. Wen, and W. Ma, "Extracting content structure for Web pages based on visual representation," Web Technologies and Applications, Springer Berlin Heidelberg, pp.406-417, 2003.

[58] "VIPS" [Online], Available: `https://github.com/tpopela/vips` Accessed on: Nov 22, 2018.

[59] Y. Zhai and B. Liu, "Web Data Extraction Based on Partial Tree Alignment," Proceeding of the 2005 International Conference on World Wide Web (WWW), pp.76-85, 2005.

[60] Y. Zhai, and B. Liu, "Structured data extraction from the Web based on partial tree alignment," IEEE Transaction on Knowledge and Data Engineering, vol.18, no.12, pp.1614-1628, 2006.

[61] "DEPTA" [online], Available: `https://github.com/seagatesoft/sde/` Accessed on: Sep 11, 2015.

[62] K. Muhammad, M. Iwai, and K. Sezaki, "An improved classification strategy for filtering relevant tweets using bag-of-word classifiers." Journal of Information Processing, vol.21, no.3, pp.507-516. 2013.

[63] J. Thorsten, "Text categorization with Support Vector Machines: Learning with many relevant features," Proceedings of the 1998 European Conference on Machine Learning, Springer Berlin Heidelberg, pp.1-7, 1998.

[64] S. Fabrizio, "Machine learning in automated text categorization," ACM Computing Surveys (CSUR), vol.34, no.1, pp.1-47, 2002.

[65] J. Jie, T. Chan, and Q. Zhao, "Clustering large sparse text data: A comparative advantage approach," Journal of Information Processing, vol. 1, pp.242-251, 2010.

[66] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv:1301.3781v3, pp.1301-3781, 2013.

[67] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," Preceding 31st Intentional Conference on Machine Learning, pp.1188-1196, 2014.

[68] D.D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," Proceeding of the Machine Learning ECML-98, pp.4-15, 1998.

[69] Y. Yiming and L. Xin, "A re-examination of text categorization methods," Proceeding 22-nd Annual International ACMSIGIR Conference on Research and Development in Information Retrieval, pp.42-49, 1999.

[70] R. Meng, S. Shen, R. R. Choudhury, and S. Nelakuditi, "AutoLabel: Labeling Places from Pictures and Websites," Proceeding of the 2016 Ubiquitous Computing (UbiComp), pp. 1159-1169, 2016.

[71] S. S. Tsai, H. Chen, D. Chen, V. Parameswaran, R. Grzeszczuk, and B. Girod, "Visual text features for image matching," Proceeding of the 2012 IEEE International Symposium on Multimedia (ISM), pp. 408-412, 2012.

[72] Z. Wojna, A. N. Gorban, D. S. Lee, K. Murphy, Q. Yu, Y. Li, and J. Ibarz, "Attention-based extraction of structured information from street view imagery," Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 844-850, 2018.

[73] R. Smith, C. Gu, D. Lee, H. Hu, R. Unnikrishnan, J. Ibarz, S. Arnoud, and S. Lin, "End-to-end interpretation of the French street name signs dataset," Proceedings of the 2016 European Conference on Computer Vision (ECCV), pp. 411-426, 2016.

[74] X. Teng, D. Guo, Y. Guo, X. Zhao, and Z. Liu, "SISE: Self-updating of indoor semantic floorplans for general entities," IEEE Transaction on Mobile Computing, vol. 1, pp. 1-14, 2018.

[75] S. Fang, C. Liu, F. Zhu, and E. J. Delp, "cTADA: The design of a crowdsourcing tool for online food image identification and segmentation," Proceed-

ing of the 2018 IEEE Southwest Symposium Image Analysis and Interpretation, pp. 1-4, 2018.

[76] Y. Movshovitz-Attias, Q. Yu, M.C. Stumpe, V. Shet, S. Arnoud, and L. Yatziv, "Ontological Supervision for fine-grained classification of street view facades," Proceedings of the 2015 IEEE Conference Computer Vision Pattern Recognition (CVPR), pp. 1693-1702, 2015.

[77] R. Jones, "Learning to Extract Entities from Labeled and Unlabeled Text." Phd thesis, School of Computer Science Carnegie Mellon University, 2005.

[78] "Geocoding." [Online]. Available: `https://www.geocoding.jp/` Accessed on: Nov 03, 2018.

[79] J. Park, G. Lee, E. Kim, J. Lim, S. Kim, H. Yang, M. Lee, and S. Hwang, "Automatic detection and recognition of Korean text in outdoor signboard images," Pattern Recognition Letters, vol. 31, no. 12, pp. 1728-1739, 2010.

[80] B. Epshtein, "Detecting text in natural scenes with stroke width transform," Proceedings of the 2010 IEEE Conference Computer Vision Pattern Recognition (CVPR), pp. 2963-2970, 2010.

[81] C. Xilin, Y. Jie, Z. Jing, and W. Alex, "Automatic detection and recognition of signs from natural scenes," IEEE Transactions on Image Processing, vol. 13, no. 1, pp. 87-99, 2004.

[82] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," Proceedings of the 2011 IEEE International Conference Computer Vision (ICCV), pp. 2564-2571. 2011.

[83] "Google Places." [Online] Available: `https://developers.google.com/places` Accessed on: Dec. 22, 2018.

[84] Z. Meng, X. Fan, X. Chen, M. Chen, and Y. Tong. "Detecting small signs from large images." arXiv:1706.08574, pp. 1-8, 2017.

[85] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "SSD: Single shot multibox detector," Proceeding of the 2016 European Conference on Computer Vision (ECCV), pp21-37, 2016.

[86] "NavVis." [Online]. Available:
http://www.navvis.com/ Accessed: Dec. 22, 2018.

[87] "Icrawler." [Online]. Available:
https://github.com/hellock/icrawler Accessed: Dec. 22, 2018.

[88] T. Kudo, K. Yamamoto, and Y. Matsumoto, "Applying conditional random fields to Japanese morphological analysis," Proceeding of the 2004 Empirical Methods Natural Language Process (EMNLP), pp. 1-8, 2004.

[89] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and F.F. Li, "ImageNet: A large-scale hierarchical image database," Proceeding of the 2009 IEEE Conference Computer Vision Pattern Recognition (CVPR), pp. 248-255, Jun. 2009.

[90] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes challenge: A retrospective," International Journal of Computer Vision (IJCV), vol. 111, no. 1, pp. 98-136, 2014.

[91] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," Proceeding of the 2007 IEEE Conference on Computer Vision and Pattern Recognition, (CVPR), pp. 1-8, 2007.

[92] S. Goferman, Z. M. Lihi, and T. Ayellet. "Context-aware saliency detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 10, pp.1915-1926, 2012.

[93] M. Sappelli, S. Verberne, and W. Kraaij. "Recommending personalized touristic sights using Google Places." Proceedings of the 36th International ACM Conference on Research and Development in Information Retrieval (SIGIR), pp.781-786, 2013.

[94] L. Yi, B. Liu, and X. Li, "Eliminating noisy information in Web pages for data mining," Proceeding of the 2003 ACM SIGKDD International Conference Knowledge. Discovery Data Mining, pp. 296-305, 2003.

[95] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," J. Documentation, vol. 28, no. 1, pp. 11-21, 1972.

[96] E. Vincent and R. Laganiere, "Detecting planar homographies in an image pair," Proceeding of the 2nd International Symposium of Image Signal Processing, pp. 182-187, 2001.

[97] D. Masko and P. Hensman, "The impact of imbalanced training data for convolutional neural networks," PhD Thesis, Kth Royal Institute of Technology, 2015.

[98] C. C. Chang and C.J. Lin, "LIBSVM: A library for support vector machines," ACM Transaction on Intelligent System Technology, vol. 2, no. 3, Article no. 27, 2011.

[99] R. Smith, "An overview of the Tesseract OCR engine," Proceeding of the 2007 International Conference Document Annual Recognition. (ICDAR), pp. 629-633, 2007.

[100] S. Flesca, G. Manco, E. Masciari, E.Rende, and A. Tagarelli. "Web Wrapper Induction: A Brief Survey." AI Communications, vol. 17, no. 2, pp:57-61, 2003.

[101] "NYC Park." [Online]. Available: `http://www.nycgovparks.org/events/` Accessed: Dec. 1, 2015.

[102] "RFC1866." [Online]. Available: `http://www.ietf.org/rfc/rfc1866.txt` Accessed: Apr. 27, 2016.

[103] "NYAS." [Online] Available: `http://www.nyas.org/` Accessed: Dec. 01, 2015.

[104] T. Mikolov, I. Sutskever, K. Chen, and G. Corrado: "Distributed representations of words and phrases and their compositionality," Advances in Neural Information Processing Systems, vol. 1, pp.3111-3119, 2013.

[105] "JSOUP" [online] Available: `http://jsoup.org/` Accessed: Dec. 22, 2018.

[106] "Komeda" [online], Available: `http://www.komeda.co.jp/` Accessed: Dec. 22, 2018.

[107] "Kuromoji" [online], Available: `http://www.atilika.org/` Accessed: Dec. 22. 2018.

[108] "Event.Locky" [online] Available: http://event.locky.jp/ Accessed: Dec. 22, 2018.

[109] "Hybrid Image Matching" [online] Available: https://github.com/liaocyintl/HybridImageMatching/ Accessed: Dec. 22, 2018.

[110] M. Abadi, P. Barham, P. Chen, J. Chen, Z. Davis, A. Dean, "TensorFlow: a system for large-scale machine learning." Proceeding of the 2016 USENIX Symposium on Operating Systems Design and Implementation (OSDI) , vol. 16, pp. 265-283, 2016.

[111] M. Van, "Blocks and fuel: Frameworks for deep learning." arXiv preprint arXiv:1506.00619, 2015.

[112] "Web page segmentation" [online] Available: https://github.com/liaocyintl/WebSegment Accessed: Dec. 22, 2018.

[113] G. Satish; J. Rahul, and G. Dhanashree, "Analysis and design of Selenium WebDriver automation testing framework." Procedia Computer Science, vol. 50, pp. 341-346, 2015.

[114] "2013 white paper on small and medium enterprises in Japan." [Online] https://goo.gl/37f93Q Accessed: Dec. 22, 2018.

[115] Z. Li, J. Gao, L. Di, HY. Shum, "The Design and Implementation of XiaoIce, an Empathetic Social Chatbot." arXiv:1812.08989, 1-26, 2018).

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Professor Dr. Nobuo Kawaguchi, who has been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a researcher. I would also like to thank my committee members, Professor Dr. Takeshi Furuhashi, Associate Professor Dr. Tetsu Iwata, and Associate Professor Dr. Ichiro Ide for serving as my committee members even at hardship. I'm grateful for your brilliant comments and suggestions, thanks to you. I acknowledge for the comments and suggestions from my co-authors, Dr. Weimin Wang, Dr. Kei Hiroi, Dr. Katsuhiko Kaji, and Dr. Ken Sakurada. Thank you for your helping in my research and supporting me in writing.

A special thanks to my family. Words cannot express how grateful I am to my beloved wife Yue Liu who encourages me and take care of my life in my most difficult moments. She gave birth to our lovely daughter Weiyin Liao on August 14, 2018. I would also like to thank my mother-in law Yulan Peng who came to Japan to take care of my wife and daughter during the completion of the thesis. I would like to thank my mother and father. Although I have less time to be your side, your prayer for me was what sustained me thus far. I would also like to thank all my friends who support me and incent me to strive towards my goal.

Thanks for all your encouragement!

List of Publications

Journal Papers

1. C. Liao, K. Hiroi, K. Kaji, K. Sakurada, and N. Kawaguchi, "Event.Locky: System of event data extraction from webpages based on web mining," Transactions of Information Processing Society of Japan, vol. 25, pp. 321-330. 2017. (Chapter 3, Chapter 4)
2. C. Liao, W. Wang, K. Sakurada, and N. Kawaguchi, "Image-matching based identification of store signage using web-crawled information," IEEE ACCESS, vol. 6, pp. 45590-45605. 2018. (Chapter 5, Chapter 6)

International Conferences

1. C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "Design and implementation of event information summarization system," IEEE 38th Annual Computer Software and Applications Conference Workshop (COMPSACW), vol. 13, pp. 572-577. 2014.
2. C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "An event data extraction method based on HTML structure analysis and machine learning," IEEE 39th Annual Computer Software and Applications Conference Workshop (COMPSACW), vol. 3, pp. 217-222. 2015.

Domestic Conferences

1. C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "Development of event information summarization system based on spatio-temporal-information [in Japanese]," The 2014 DICOMO Multimedia, Distributed, Cooperative, and Mobile Symposium (DICOMO), pp. 657-665. 2014.
2. C. Liao, K. Hiroi, K. Kaji, and N. Kawaguchi, "A proportion of event data extraction based on HTML structure analysis and machine learning [in Japanese]," IPSJ SIG on UBI, vol. 13, pp. 1-7. 2015.
3. C. Liao, K. Sakurada, and N. Kawaguchi, "A proposal of POI potential information search engine based on web contents analysis [in Japanese]," The 79th National Convention of IPSJ, vol. 3, pp. 45-46. 2017.