Unsupervised Extraction and Normalization of Product Attributes from Web Pages

XIONG, Jiani

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Systems Engineering and Engineering Management

The Chinese University of Hong Kong
July 2010

摘要

網絡用户一般會在考慮購買產品前瀏覽不同的零售網站以作比較。但由於相關的零售網站以及產品型號眾多，網絡用户往往要花很多時間去蒐集詳細資料。對於一般用户來說，這絕對是高成本、低效能的工作。有見及此，本論文提出一個嶄新的框架，可以從不同的零售網站的原始網頁執行無監督式學習(unsupervised learning)的產品屬性的提取與規範化(product attribute extraction and normalization)。因此，我們的做法可以處理大量不同佈局格式的網頁的產品屬性的提取與規範化。我們還開發了一個生成模型，可以模型在網頁裡的文本片段的生成。我們更爲該生成模型開發了一個推算方法。我們利用來自49個零售網站158個關於數碼相機、音樂播放機和液晶電視的網頁進行大規模的實驗。實驗證明我們的方法比現今最先進的方法取得更好的表現。

Abstract

We investigate the problem of jointly extracting and normalizing product attributes from different product description web sites without any labeled training example. One challenge of this problem is that it needs to handle a huge number of Web pages in diverse layout formats not known in advance. To tackle this problem, we consider the clues embodied in the text content and the layout of web pages. A generative model is developed to model the generation of text fragments in web pages taking into consideration of the relationship among the text content and layout formats of text fragments. The attribute name and value contained in a text fragment are differentiated providing finer-grained information of product attributes. We employ Dirichlet process prior in our framework leading to another characteristic that it allows unlimited number of product attributes. An unsupervised inference algorithm based on MCMC is derived. We evaluate our framework by conducting experiments on three different domains consisting of 158 Web pages from 49 different web sites. The experimental results are promising and show that our framework is effective.

Acknowledgments

I would like to thank many people who have contributed to this thesis.

Firstly, I would like to thank my supervisor Prof. Wai Lam, for his guidance and support in my two-year study. On the other hand, I would like to thank Dr. Tak-Lam Wong, Gatien for his assistance to improve my research. In addition, I would like to thank Prof. Hong Cheng and Prof. Chun-Hung Cheng as my examination committee for their comments given on this thesis.

Next, I would like to thank my friends Cecia, Bo, Lee, Annie, Catherine and Ivier, who contribute with great support and encouragement in my life.

Finally, I would thank for my parents, who always understand my choice and stand behind me given no condition.

Thesis/Assessment Committee

Professor CHENG Hong (Chair)
Professor LAM Wai (Thesis Supervisor)
Professor CHENG Chun Hung (Committee Member)
Professor YAN Zhang (External Examiner)

Contents

List of Figures

List of Tables

Chapter 1

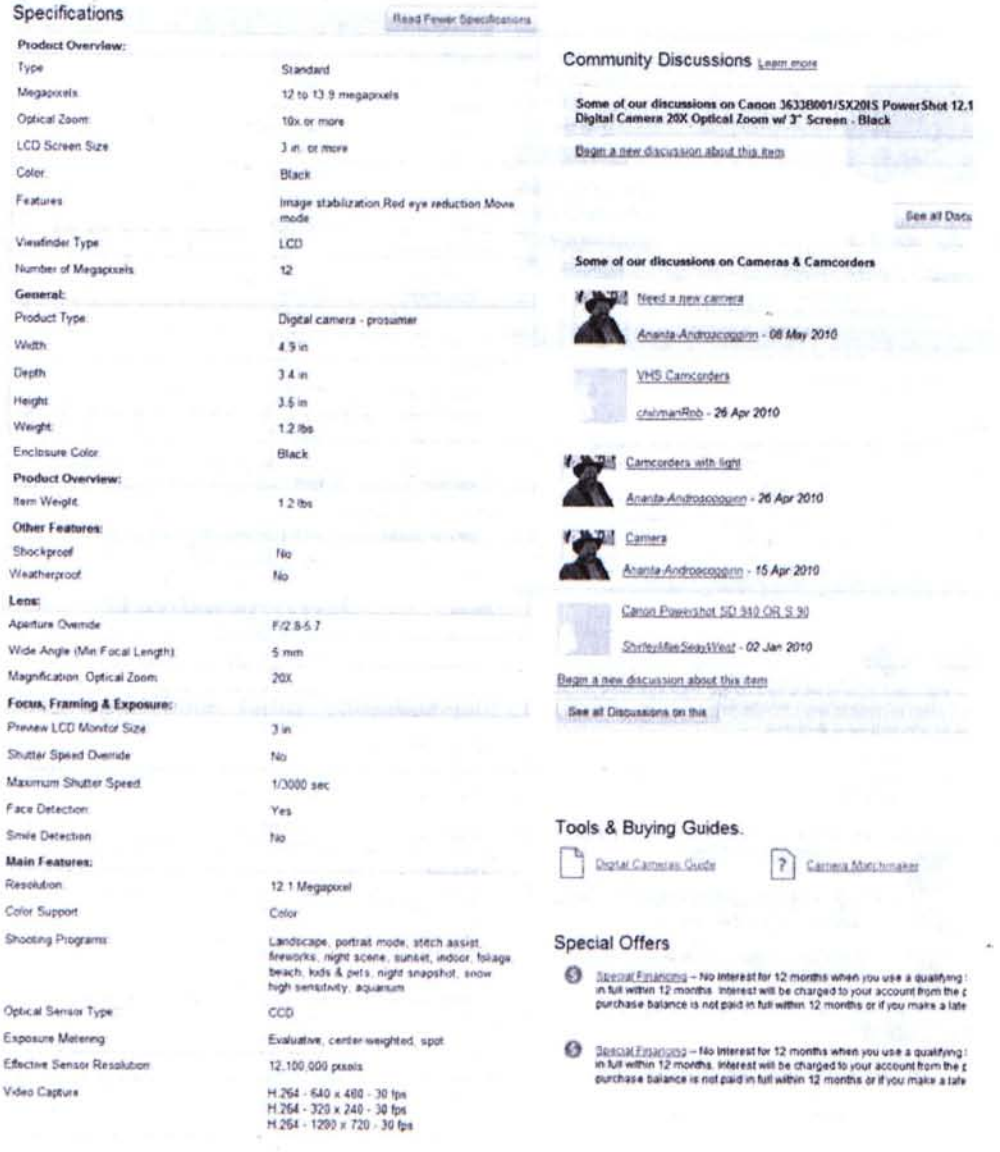
Introduction

1.1 Background

Dated from 1990s, online shopping [36] created a path whereby consumers can directly buy products from a seller interactively in real-time without an intermediary service over the Internet. Its rapid development can be seen from the largely and steadily increasing amount of existing online store web sites providing information of millions of kinds of products. Since information on the web is always distributed, ambiguous, and unstructured, it becomes a difficult task for a consumer to retrieve, analyze, and compare products.

Different web sites tend to organize information in their own fashion. Figures 1.1 and 1.2 show two sample web pages about digital cameras collected from two different online store web sites. They organize and display the product attributes, such as "color", "weight", etc. of the same product in a tabular format. To acquire information about products, say digital

Figure 1.1: A sample of a portion of a web page showing some product information of a digital camera collected from a web site. (Web site URL: http://www.newegg.com)



Specifications

Read Fewer Specifications

Product Overview:	
Type	Standard
Megapixels	12 to 13.9 megapixels
Optical Zoom	10x or more
LCD Screen Size	3 in. or more
Color	Black
Features	Image stabilization, Red eye reduction, Movie mode
Viewfinder Type	LCD
Number of Megapixels	12
General:	
Product Type	Digital camera - prosumer
Width	4.9 in
Depth	3.4 in
Height	3.6 in
Weight	1.2 lbs
Enclosure Color	Black
Product Overview:	
Item Weight	1.2 lbs
Other Features:	
Shockproof	No
Weatherproof	No
Lens:	
Aperture Override	F/2.8-5.7
Wide Angle (Min Focal Length)	5 mm
Magnification, Optical Zoom	20X
Focus, Framing & Exposure:	
Preview LCD Monitor Size	3 in
Shutter Speed Override	No
Maximum Shutter Speed	1/3000 sec
Face Detection	Yes
Smile Detection	No
Main Features:	
Resolution	12.1 Megapixel
Color Support	Color
Shooting Programs	Landscape, portrait mode, stitch assist, fireworks, night scene, sunset, indoor, foliage, beach, kids & pets, night snapshot, snow, high sensitivity, aquarium
Optical Sensor Type	CCD
Exposure Metering	Evaluative, center-weighted, spot
Effective Sensor Resolution	12,100,000 pixels
Video Capture	H.264 - 640 x 480 - 30 fps H.264 - 320 x 240 - 30 fps H.264 - 1280 x 720 - 30 fps

Community Discussions Learn more

Some of our discussions on Canon 3633B001/SX20IS PowerShot 12.1 Digital Camera 20X Optical Zoom w/ 3" Screen - Black

Begin a new discussion about this item

See all Docs

Some of our discussions on Cameras & Camcorders

Need a new camera

Ananta-Androscoggin - *08 May 2010*

VHS Camcorders

chairmanRob - *26 Apr 2010*

Camcorders with light

Ananta-Androscoggin - *26 Apr 2010*

Camera

Ananta-Androscoggin - *15 Apr 2010*

Canon Powershot SD 940 IS 90

ShirleyMaeSeasWest - *02 Jan 2010*

Begin a new discussion about this item

See all Discussions on this

Tools & Buying Guides.

Digital Cameras Guide

Camera Matchmaker

Special Offers

Special Financing – No interest for 12 months when you use a qualifying in full within 12 months. Interest will be charged to your account from the purchase balance is not paid in full within 12 months or if you make a late

Special Financing – No interest for 12 months when you use a qualifying in full within 12 months. Interest will be charged to your account from the purchase balance is not paid in full within 12 months or if you make a late

Figure 1.2: A sample of a portion of a web page showing some product information of a camera collected from a web site different from the one depicted in Figure 1. (Web site URL: http://www.sears.com)

cameras from the Internet, one third of people [36] who shop online usually query search engines, which are in fact information retrieval systems, with keywords hoping that the returned results contain relevant web sites or web pages. Since the basic unit of the results returned by a search engine is an entire web document, the user is required to manually identify the precise text fragments about the product attributes from web pages, resulting in ineffective extraction and analysis of information. This drawback inspires the idea of precise information extraction from web content. Layout information of web pages are taken in consideration to achieve this task. Moreover, search engines usually have limitations in term matching, web documents containing terms with the same semantic meaning as query terms may not be returned. This raises the need of a framework that can achieve precise information extraction as well as product attribute normalization. In addition, the large amount of search results requires the framework to be unsupervised, which can save human efforts in preparing training data.

1.2 Motivation

Databases normally organize data in a structured way. Unlike this manner, the data contained in web pages, which are usually formatted by human, does not have a well-defined structure. Semi-structured text documents containing a mix of ungrammatical texts and HTML tags can be one example. It poses additional difficulty to automatically extract the desired information. To solve this problem, several works have been developed to extract particular contents for specific tasks from web pages [34]. In particular, wrappers for

information extraction [20] is a promising approach to addressing this problem. Information extraction wrappers are trainable methods that analyze semi-structured texts and learn patterns for extracting inforamtion. They are often developed for specific tasks, for example, to extract the values of certain product attributes of digital cameras from a particular web site. In wrapper induction, the purpose is to automatically construct a wrapper from a set of training examples collected from a web site. The learned wrapper, normally composed of a set of extraction rules, can be applied to the remaining web pages of the *same* site to extract information. For instance, one can prepare a set of training examples of digital cameras with values of the attribute "Effective Pixels" of each record annotated to learn a wrapper. The learned wrapper can then be applied to the remaining web pages of the *same* web site to extract the values of "Effective Pixels" of different digital cameras.

However, existing wrapper learning methods have several major limitations. One of the limitations is that they are supervised approaches demanding for vast amount of human efforts in preparing training examples. Meanwhile, no attribute other than the pre-defined ones can be extracted since the learned wrapper can only recognize attributes that are annotated in advance without discovering other attributes. Referring to the previous example, a wrapper trained for the attribute "Effective Pixels" of digital cameras cannot extract other attributes such as "Weight". Another limitation is that the learned wrapper can only be applied to the web site where the training examples come from. For instance, the learned wrapper for Figure 1.1 cannot be applied to the web site shown in Figure 1.2 due to their

different layout formats. Layout formats of web sites are always different from one to another. Hence preparing a particular wrapper for each web site makes it ineffective to extract desired information from vast amount of web sites.

Several approaches have been developed to address the above wrapper adaptation problem. For example, unsupervised wrapper learning aims at reducing the human effort by learning extraction rules without any training examples. Some unsupervised wrappers have been proposed by making use of layout information of web pages which are generated by templates [11]. The major idea of these approaches is to align the structures of different web documents generated from the same template. Text fragments that are located in the same position after the alignment but different in content are considered to be useful fields and they are extracted. However, since the extraction is template dependent, the fields extracted from different web sites, even in the same domain, may not be synchronized. For instance, a field extracted from a particular site about digital cameras may contain values of both product attributes "weight" and "dimension", whereas in another site, the values of "weight" and "dimension" are extracted in two different fields. Chuang et al. proposed an unsupervised wrapper learning technique which can construct wrappers to extract synchronized data from multiple sources [10]. For example, the field containing both "weight" and "dimension" can be further segmented into two separate fields synchronizing the two separate fields about "weight" and "dimension" extracted in another web site. The rational of their approach is to identify the optimal segmentation of the text in web

pages from different sites. However, their method requires to train a field model for each field capturing the field's characteristics. Suppose there are two different field models for "weight" and "dimension". These field models are required to be trained from manually prepared training examples, or developed by human experts in advance hence resulting in high labor cost. Moreover, it cannot handle previously unseen fields. They proposed a heuristic method for training the field models for previously unseen fields in an unsupervised manner. The idea is to consider each group of aligned segments created by an unsupervised wrapper as a single field, and train a field model for each group using HMM with a pre-defined labeling rule. However, such method can only apply to a web page that contains multiple records. For web pages containing a single record, such as the ones in Figures 1.1 and 1.2, there exists neither group of aligned segments, nor a single group in which the aligned segments refer to different fields.

Another shortcoming of existing unsupervised wrapper learning methods is that they cannot resolve the extracted information from different web sites. For example, suppose "portrait" and "landscape" are two text fragments extracted from web pages about digital cameras. Product attribute normalization aims at clustering extracted text fragments into the same underlying attribute based on their content. For instance, it enables grouping the text fragments "portrait" and "landscape" into the same cluster because they refer to the same attribute. For another example, both "effective pixels" in Figure 1.1 and "effective sensor resolution" in Figure 1.2 refer to the same attribute "effective-pixel", hence they also should be grouped into the

same cluster. Production attribute normalization is beneficial for users to search and compare different product records, as well as software agents to conduct intelligent tasks. Chuang et al. proposed a clustering method to match the extracted data based on the tokens of the data in a separate step [10]. Unfortunately, since their method mainly considers simple overlapping of tokens, it is not able to resolve the text fragments "portrait" and "landscape" to the same attribute. Moreover, the clustering algorithm requires to fix the number of clusters in advance. However, the number of attributes in a domain is unknown in practice.

In summary, existing approaches suffer from one or more of the following problems: (i) the learned extraction rules cannot handle web pages from different web sites with layout formats not known in advance, (ii) human effort is needed to prepare training examples, (iii) they are unable to discover previously unseen attributes, and (iv) the extracted attribute values are not normalized according to the product attributes they refer. In this thesis, we aim at addressing these problems by developing an unsupervised learning framework for jointly extracting and normalizing product attributes from different web sites.

1.3 Our Approach

We propose to develop a framework which can automatically extract product attribute information from a large number of different web sites with layout formats not known in advance, normalize the extracted attribute informa-

tion, and organize the information in a structured manner. This framework aims at jointly extracting and normalizing product attributes in an unsupervised way. One advantage of solving the two problems in a single framework is that the resulting solution can optimize the performance of the two tasks and reduce possible conflicts. Our mathematical formulation formally illustrates that the two tasks can be tackled in a coherent manner. We illustrate the idea of our framework using a running example.

Referring to the web pages about two different digital cameras shown in Figures 1.1 and 1.2. These two web pages are collected from two different web sites and therefore they have different layout formats. The task of extraction is to extract product attribute relevant information from the web page and discard the irrelevant information. For instance, the tables under the tab "specification" in Figures 1.1 and 1.2 will be identified as attribute-relevant information in the extraction task, while skipping other information like advertisements and user reviews by treating them as attribute-irrelevant. Meanwhile, the normalization task will resolve the reference attribute that each attribute item refers to. For instance, both "Focal length f=5.0 - 100 mm (35mm equivalent: 28-560mm)" in Figure 1.1 and "Wide Angle (Min.Focal Length): 5 mm " are referring to the reference attribute "focal-length", so they are likely to be grouped to the same cluster.

We define an attribute as a field of a product; an attribute value as the text representing the value of a particular attribute for a record; and an attribute name as the text displayed on the web page to show the at-

tribute to which an attribute value refers. For instance, suppose there is a product attribute about "effective-pixels". An example of its attribute name and attribute value are "Effective Pixels" and "12.1 MP" respectively for the digital camera shown in Figure 1.1. Another example of the attribute name and attribute value of the same attribute are "Megapixels" and "12 to 13.9 megapixels" respectively for the digital camera shown in Figure 1.2. Meanwhile, other information besides product attributes, for example, the columns under "Special offers" in Figure 1.1 and user comments in Figure 1.2, is defined as "irrelevant" data with product attribute information. Very often, users may know a few terms related to the content of some attributes of interest in the domain. Such information can be easily obtained, for example, by scanning one web page about digital cameras and collecting a few terms such as "image" and "stabilizer" in a list. Based on such information, one can infer from the content of the text fragments in the web page that the text fragments "Image Stabilization" and "Optical Image Stabilizer System" in Figure 1.1 likely refer to an attribute name and the corresponding attribute value respectively. In addition, there commonly exists some previously unseen attributes. For example, from the layout format of the web page in Figure 1.1, it can be inferred that the text fragments "Metering System" and "Evaluative*, Center-weighted average, Spot**" should be another pair of attribute name and attribute value because the layout format of these text fragments is similar to that of the extracted text fragments "Image Stabilization" and "Optical Image Stabilizer System" in Figure 1.1. Meanwhile, some annotation attached in the field, i.e. "*Facial brightness is also evaluated in Face Detect. ** Metering frame is fixed to the center/linked

to AF frame", is treated as attribute value as well. Similarly, more pairs of attribute name and attribute value such as "Exposure Metering" and "Evaluative, center-weighted, spot", can be discovered from Figure 1.2. As the terms appeared within the text fragment "Evaluative, center-weighted, spot" in Figure 1.2 and that of the text fragment "Evaluative*, Center-weighted average, Spot**" in Figure 1.1 share certain similarity, one can infer that these two extracted pairs of attribute name and attribute value refer to the same attribute. Additionally, this also increases the degree of confidence in extraction. This scenario shows that the content and the layout format of attribute names and attribute values in web pages can be cooperative for extraction and discovery of previously unseen attributes. In our framework, the *page-independent* information, which refers to the sematic meaning of the contents, as well as the *page-dependent* information, which refers to the layout format, jointly affect the result of extraction and normalization of product attributes.

Sometimes, an extracted attribute value from a particular web page may not be associated with any attribute name. For instance, there is an extracted attribute value "portrait" not associated with any attribute name in a web page. It is not trivial for a user to understand the attribute that this attribute value refers to. Suppose the attribute value "Landscape, portrait mode, stitch assist, fireworks, night scene, sunset, indoor, foliage, beach, kids & pets, night snapshot, snow, high sensitivity, aquarium" is extracted in Figure 1.2 representing the attribute "Shooting Programs" of the digital camera. If the text fragments can be automatically clustered to the same

group representing an attribute, one can easily observe that they refer to the same attribute corresponding to "Shooting Programs". This allows better understanding and interpretation of the semantic meaning of the extracted text fragments in the same group.

1.4 Potential Applications

Our framework can summarize information from different data sources and help users analyze and compare products. For example, shopping search engines, one of the most popular Internet application that has been widely used by online consumers, can potentially adopt it to enhance service. The interface of existing shopping search engines normally provides an HTML form field into which a user can type product queries to return lists of vendors selling a particular product, as well as pricing information. The incorporation of our approach enables product search and comparison based on product attributes other than price. Moreover, the extracted and normalized information can be applied to other intelligent tasks. The organized database of product information generated by our approach can be applied to business users for conducting further data mining. Customers interests on product types can be inferred by conducting analysis on the data of product attributes, therefore products with similar properties can be recommended to users.

Another potential application is to construct the product attribute taxonomy which can capture the relationship among attributes in a hierarchi-

cal structure. For example, "max-focal-length" and "min-focal-length" are two different normalized product attributes in our framework. In essence, they can be grouped together and become a more abstract reference attribute "focal-length". As a consequence, if we can organize the reference attributes in a hierarchical structure in which the lower level nodes represent finer grained reference attributes and the higher level nodes represent more abstract reference attributes, users can compare products in different level of abstraction.

1.5 Research Contributions

Our approach to unsupervised product attribute extraction and normalization can effectively acquire and organize the product information from a large number of web pages with different layout formats. We have developed an unsupervised learning framework for jointly extracting and normalizing product attributes from multiple web sites. For example, the text fragments "fireworks" are "portrait" are samples of extracted and normalized text fragments in the digital camera domain using our method. These two fragments do not have words in common, but actually they refer to the product attribute "shooting mode" in the digital camera domain. Unlike existing methods which conduct the extraction and normalization tasks in separate steps unavoidably leading to the accumulation of errors, we propose a single framework which can conduct extraction and normalization tasks simultaneously in an unsupervised manner.

Our framework considers the *page-independent* content information and the *page-dependent* layout information in a single framework. As illustrated in the above motivating example, the mutual influence between the content and the layout format of text fragments provides useful clues for attribute extraction and normalization tasks. We design a probabilistic graphical model to model the relationship between the content and layout information for solving the two tasks simultaneously.

In practice, the number of attributes is not known in advance. We employ Dirichlet process prior leading to another characteristic that the number of attributes to be discovered needs not to be fixed and can be unlimited, different from existing works which need to fix the number of attributes. This can handle product attributes not known in advance and new attributes can be discovered. Theoretically, it can handle an infinite number of attributes. We also incorporate Hidden Markov Models (HMM) to achieve labeling for tokens of each attribute field. Since words with similar sematic meaning are likely to refer to the same attribute, each single attribute is applied with a single HMM, with higher probabilities that related words would be generated. Therefore, labeling of attribute name and attribute value can be executed more precisely.

The semantic meaning of the extracted and normalized attributes can be visualized by a set of weighted terms in the model. This can significantly help users understand and interpret the attributes. We have conducted extensive experiments from three different domains consisting of 158 web pages

from 49 web sites. The experimental results show that our framework is robust and effective.

1.6 Thesis Organization

Chapter 2 presents a literature survey of existing works addressing issues related to attribute extraction and normalization. Chapter 3 provides the problem definition and gives a preliminary of our framework as well as the graphical models. Chapter 4 describes our generative models for the generation of text fragments in web pages and presents our unsupervised inference method for solving the problem. In Chapter 5, experiments are conducted based on our framework. We draw the conclusions and present several possible directions of future work in Chapter 6.

Chapter 2

Literature Survey

Our approach to unsupervised product attribute extraction and normalization draws on techniques from different research areas. In the following sections we will give a literature survey of information extraction techniques. Then, works on attribute normalization as well as approaches aiming at integration of these two tasks will be discussed.

2.1 Supervised Extraction Approaches

In the survey on information extraction systems conducted by Chang et al. [8], they described various supervised information extraction techniques that have been proposed to extract attributes from semi-structured documents including web pages. They compared the information extraction systems in three dimensions: the techniques used, the degree of automation, and the task domain explaining why an IE system fails to handle some web sites of particular structures. The criteria are believed to be capable of providing

qualitatively measures to evaluate various IE approaches.

Meng et al. proposed a novel schema-guided approach to wrapper generation [25]. They provided a user-friendly interface that allows users to define the schema of the data to be extracted, and specifies mappings from an HTML page to the target schema. The system can automatically generate an extraction rule to extract data from the page, based on the mappings. Since the user never has to deal with the internal extraction rule, or even familiarity with the details of HTML, the approach to wrapper generation can significantly reduce the human work.

Lafferty presented conditional random fields [21] as a framework for building probabilistic models to segment and label sequence data, offering several advantages over hidden Markov models and stochastic grammars for such tasks, including the ability to relax strong independence assumptions made in those models. This framework had been applied to extract information from web documents achieving the state-of-the-art performance.

Sarawagi and Cohen developed a semi-Markov CRF model [29] which can assign labels to segments of a sequence rather than individual elements of a sequence. They presented Semi-CRFs as a tractable extension of CRFs that offer much of the power of higher-order models without the associated computational cost. Also they conducted experiments on five named entity recognition problems and proved that semi-CRFs generally outperformed conventional CRFs.

Sutton et al. proposed Dynamic CRF models (DCRFs) [32] for labeling and segmenting sequence data. The model was presented as an integration of the best of both conditional random fields and the widely successful dynamic Bayesian networks (DBNs). Therefore, it addressed difficulties both of DBNs, by incorporating arbitrary overlapping input features, and of previous conditional models, by allowing more complex dependence between labels. In the paper, inference in DCRFs were performed using approximate methods, and training by maximum a posteriori estimation. Experiment results showed that a DCRF performed better than a series of linear-chain CRFs, achieving comparable performance using only half the training data.

Zhu et al. developed an approach to segmenting web pages and labeling the elements within the web pages from different sources [39] achieving promising performance. Their approach was template-independent and hence it could handle web pages in different layout formats. The main idea of their approach was to integrate Hierarchical Conditional Random Fields (HCRF), which is used for learning the web page structure, and Semi-Conditional Random Fields (Semi-CRF), which is used for labeling the text fragments, into one model. Extraction was accomplished by jointly solving the two problems under a single framework.

However, the supervised methods can only partially solve the problems in wrapper learning. Since the learning requires training examples, and the learned extraction rules can only extract those product attributes pre-

specified in the training examples in advance, there are still limitations that are not addressed.

2.2 Unsupervised Extraction Approaches

Several methods have been developed to extract data from web pages without supervision. IEPAD [7] is a system aiming at extracting information by recognizing the repeated patterns using PAT trees inside the web pages. The core technique of this work is unsupervised discovery of extraction rules. The system can automatically identify record boundary by repeated pattern mining and multiple sequence alignment. The discovery of repeated patterns are realized through a data structure call PAT trees. Additionally, repeated patterns are further extended by pattern alignment to comprehend all record instances. No human intervention and training example was involved in this work.

Liu et al. [22] proposed a system known as MDR (Mining Data Records in web pages) to discover the data region in a web page by making use of the repeated pattern in HTML tag trees. This system firstly builds a HTML tag tree of the page, then conducts mining algorithms in data regions in the page using the tag tree and string comparison, and heuristics are then applied to extract useful information from the data region. Both IEPAD and MDR assume that the input web pages contain multiple records and repeated patterns. It exploits such evidence and recognizes the repeated patterns appeared in the web pages. However, the web pages are required to have similar

layout format and this may not be true in web pages collected from different sources.

Grenager et al. [15] applied hidden Markov model and exploited prior knowledge to extract information in an unsupervised manner. They demonstrated that for certain field structured extraction tasks, such as classified advertisements and bibliographic citations, small amounts of prior knowledge could be used to learn effective models in a primarily unsupervised fashion, which could dramatically improve the quality of the learned structure. However, the quality of the extracted data was unlikely suitable for subsequent data mining tasks.

Golgher et al. [14] proposed a web data extraction method by applying a bootstrapping technique and a query-like approach. The idea was to exploit the existing repositories which can be generated from legacy databases, or the data extracted by an existing wrapper for a web site. The system searched the text fragments exactly matched with the elements in the repository in the unseen web page. A revised wrapper for the unseen site was then generated by bootstrapping using the matched item. This approach assumed that the seed words, which refer to the elements in the source repository in their framework, must appear in the unseen web page. However, exact matching of items from different web sites was generally not feasible.

Wong and Lam [37] aimed at reducing the human work of preparing training examples by automatically adapting extraction knowledge learned

from a source web site to new unseen sites. Two kinds of features related to the text fragments from the web documents, site-invariant features and site-dependent features, were investigated in this work. The site-invariant features, which derived from previously learned extraction knowledge and the items previously collected or extracted from the source web site, would be exploited to automatically seek a new set of training examples in the new unseen target site. Both the site-dependent features and the site-invariant features of these automatically discovered training examples would be considered in the learning of new information extraction knowledge for the target site, and no human effort was needed.

Besides, Probst et al. [27] proposed a semi-supervised algorithm to extract attribute value pairs from text description. Their approach aimed at handling free text descriptions by making use of natural language processing techniques, that is, using unlabeled data to extract an initial seed list that served as training data for the supervised and semi-supervised classification algorithms. Hence, it required very little initial user supervision, but could not be applied to web documents which were composed of mixing HTML tags and free texts.

2.3 Attribute Normalization

The product attribute normalization problem is related to the task of record resolution. Record resolution is the problem of determining which records in a database refer to the same entities, and is a crucial and expensive step in

the data mining process. Singla and Domingos [31] developed an approach to record resolution based on Markov Logic Network. Their approach was to formulate first-order logic and probabilistic graphical models and combine them in Markov logic by attaching weights to first-order formulas, and viewing them as templates for features of Markov networks. Experiments on two citation databases showed that the resulting learning and inference problems can be solved efficiently. Bhattacharya and Getoor [3] proposed an unsupervised approach for record resolution based on Latent Dirichlet Allocation (LDA). A probabilistic generative model was developed for collectively resolving entities in relational data, which did not make pairwise decisions and introduced a group variable to capture relationships between entities. One limitation of these approaches is that the entities are required to be extracted in advance and cannot be applied to raw data.

2.4 Integrated Approaches

A common drawback of existing methods is that the extraction and normalization tasks are conducted in two separate steps, leading to conflict solutions and degrading overall performance. Approaches based on CRF have been proposed to collaboratively conduct information extraction and mining. McCallum and Jensen [24] proposed the use of unified, relational, undirected graphical models for information extraction and data mining, in which extraction decisions and data-mining decisions were made in the same probabilistic "currency" with a common inference procedure. In this case, each of the two components were able make up for the weaknesses of the

other and therefore improving the performance of both. Wellner [35] et al described an approach to integrated inference for extraction and coreference based on conditionally-trained undirected graphical models. They advocated conditional-probability training to allow free use of arbitrary non-independent features of the input, and adapted undirected graphical models to represent autocorrelation and arbitrary possibly cyclic dependencies. Also approximate inference and parameter estimation were performed in these large graphical models by structured approximations. However, the attributes to be extracted have to be known in advance in these approaches and previously unseen attributes cannot be handled.

To address the problems mentioned above, we apply the use of Dirichlet process prior. Dirichlet process prior has been studied and applied in image analysis [5] and language modeling [33]. Hall et al. [17] have employed Dirichlet process prior to model the relationship between two fields of data, which was called cross-field dependence in this paper, and applied to the research area of de-duplication that explicitly models cross-field dependence. The model used a single set of latent variables to control two disparate clustering models: a Dirichlet-multinomial model over titles, and a non-exchangeable string-edit model over venues. Our framework extends the Dirichlet process mixture model by integrating HMM model, and shows that the content and layout information of text fragments can be considered jointly to achieve a better solution in product attribute extraction and normalization.

Chapter 3

Problem Definition and Preliminaries

This chapter aims at defining the main problem to be tackled in this thesis. Pre-processing of raw data is mentioned in this chapter. We also give a brief introduction of our framework, as well as the background knowledge of the graphical models, since our framework is developed using the graphical models for representation.

3.1 Problem Definition

In a product domain $\mathcal{D}$, let $\mathcal{A}$ denote a set of reference attributes and a_i be the i-th attribute in $\mathcal{A}$. For example, in the digital camera domain, reference attributes of digital cameras may include "lcd-screen-size", "effective-pixels", "focal-length", etc. We design a special element denoted as $\bar{a}$ representing "not-an-attribute". Since the number of attributes is unknown and hence

the size of $\mathcal{A}$ denoted by $|\mathcal{A}|$ is between 0 and ∞.

Given a collection of product record web pages $\mathcal{W}$ collected from a set of web sites $\mathcal{S}$. Let $w_i(s)$ be i-th page collected from the site s. Within the web page $w_i(s)$, we can collect a set of text fragments $X(w_i(s))$. For example, "Optical Zoom 20X" and "Focal Length f=5.0 - 100 mm (35mm equivalent: 28-560mm)" are samples of text fragments collected from the page shown in Figure 1.1. Let $x_j(w_i(s))$ be the j-th text fragment in the web page $w_i(s)$. Essentially, each x in $X(w_i(s))$ can be represented by a five-field tuple (U, Q, L, T, A). U refers to the tokens of each text fragment, and Q refers to the label information of the tokens, i.e. q_1 represents the attribute name information, labeled as "attribute-name", and q_2 represents the attribute value information contained in the text fragment, labeled as "attribute-value", respectively. In particular, $\bar{q}$ represents that the token is a "attribute-irrelevant" token. Take the fragment "Focal Length f=5.0 - 100 mm (35mm equivalent: 28-560mm)" as an example. The tokens "Focal Length" refer to the attribute name, while the remaining tokens correspond to the attribute value. For another example, the text fragment "Community Discussions" corresponds to neither the attribute name nor the attribute value, so it refers to attribute-irrelevant information. L refers to the layout information of the text fragment. For example, the text fragment "General" is in boldface and in larger font size in Figure 1.1. T, defined as the target information, is a binary variable which is equal to 1 if the underlying text fragment is related to an attribute in $\mathcal{A}$, and 0 otherwise. For example, the values of T for the text fragments "General" and "Focal Length f=5.0 - 100

mm (35mm equivalent: 28-560mm)" are 0 and 1 respectively. A, defined as the attribute information, refers to the reference attribute that the underlying text fragment belongs to. It is a realization of $\mathcal{A}$ and hence it must be equal to one of the elements in $\mathcal{A}$. For example, the values of A for the text fragments "Focal Length f=5.0 - 100 mm (35mm equivalent: 28-560mm)" and ※ "Wide Angle (Min.Focal Length): 5 mm " collected from Figures 1.1 and 1.2 respectively should be equal to the reference attributes "focal-length" included in $\mathcal{A}$.

In practice, the layout information L and the token information U of a text fragment can be observed from web pages. However, the target information T, the attribute information A and the label information of tokens Q cannot be observed. As a result, given the observation of L and U, the task of product attribute extraction can be formulated as the prediction for the value of T for each text fragment in web pages aiming at discovering all text fragments corresponding to certain attributes A. Formally, for each text fragment, we aim at finding $T = t^*$, such that

$$t^* = \arg\max_t P(T = t|L, U) \tag{3.1}$$

The task of attribute normalization can be defined as the prediction of the value of A for each text fragment, so that one can obtain the reference attribute to which the underlying text fragment refers. Formally, for each text fragment, we aim at finding $A = a^*$, such that

$$a^* = \arg\max_a P(A = a|L, U) \tag{3.2}$$

Meanwhile, our framework predicts the label information of tokens Q for each text fragment, and the information can help with the task of extraction as well as the task of normalization. Formally, for each text fragment, we aim at finding $Q = q^*$, such that

$$q^* = \arg\max_q P(Q = q|L, U) \tag{3.3}$$

When $T = 1$, we have $P(A = a|L, U) > 0$, $P(Q = q, q \in q_1, q_2|L, U) > 0$ for some $a \in \mathcal{A}\backslash\bar{a}$ and $P(A = \bar{a}|L, U) = 0$. When $T = 0$, we have $P(A = \bar{a}|L, U) = 1$, $P(Q = \bar{q}|L, U) = 1$. As a result, conducting product attribute extraction and normalization separately may lead to conflict solutions degrading the performance of both tasks. In our framework, we aim at predicting the values of T, A and Q such that the joint probability $P(T, A, Q|L, U)$ can be maximized leading to a solution satisfying both tasks.

3.2 Preliminaries

3.2.1 Web Pre-processing

Our framework can automatically extract and normalize product attributes collected from web pages. A web page is an HTML document mixed with ungrammatical text fragments and HTML tags. The web pages are first pre-processed to automatically identify a set of text fragments. Some of the identified text fragments are related to product attributes, while some of them are irrelevant. Each web page can be represented by a Document Object Model (DOM) structure [16]. DOM structure is an ordered tree representing the layout format of a web page. There are two kinds of nodes in

a DOM structure. The first kind of nodes are the HTML nodes which are responsible for the layout format of the web page. These nodes are labeled with the corresponding HTML tags. The second kind of nodes are the text nodes, which are responsible for the text displayed in browsers. These nodes are simply labeled with the associated texts. Figures 3.1 and 3.2 depict the excerpt of the HTML document and a portion of the DOM structure corresponding to the web page in Figure 1.2 respectively.

We define a text fragment as the text within a block of information such as a line, a paragraph, a row of table, etc., conveying a single idea or message. To identify the text fragments, we select some HTML tags such as TR, BR, P, etc. We call these HTML tags and the corresponding HTML nodes in the DOM structure separators and separator nodes respectively. Consider a separator node, namely, $node_{seq}$, in a DOM structure. The texts contained in the text nodes that are offspring of $node_{seq}$ but do not have other separator nodes between $node_{seq}$ and the underlying text nodes are concatenated to form a text fragment. Each single word within a text fragment is defined as a "token". The tokens denote the content information of the web page to our framework. For example, "Display Information Battery status, focus, fader, menu, exposure" is an identified text fragment because the texts "Display Information" and "Battery status, focus, fader, menu, exposure" are the offspring of the separator TR and there is no other separator node between them in the DOM structure.

Additionally, we get the layout information from the DOM structure.

```
                <div id="pcraTabContent3" style="display:block;">
<div id="pcraSpecs">
  <table cellpadding="0" cellspacing="0" class="specification">
    <tr>
      <td colspan="2" class="title">General</td>
    </tr>
    <tr>
      <td class="name">Brand</td>
      <td class="desc">Canon</td>
    </tr>
    <tr>
      <td class="name">Series</td>
      <td class="desc">PowerShot SX Series</td>
    </tr>
    <tr>
      <td class="name">Model</td>
      <td class="desc">PowerShot SX20 IS</td>
    </tr>
    <tr>
      <td class="name">Color</td>
      <td class="desc">Black</td>
    </tr>
    <tr>
      <td class="name">Dimensions (WxHxD)</td>
      <td class="desc">4.88" x 3.48" x 3.42"</td>
    </tr>
    <tr>
      <td class="name">Weight</td>
      <td class="desc">Approx. 19.8 oz./560g (camera body only)</td>
    </tr>
    <tr>
      <td class="name">Type</td>
      <td class="desc">SLR-Style</td>
    </tr>
    <tr>
      <td colspan="2" class="title">Image Sensor</td>
    </tr>
```

Figure 3.1: An excerpt of the HTML texts for the web page shown in Figure 1.2

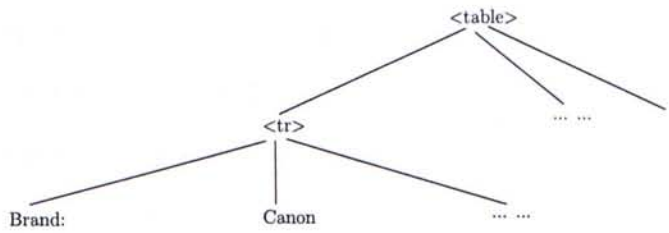


Figure 3.2: A portion of the DOM structure for the web page shown in Figure 1.2

Besides tokens about content, each text fragment may also contain one or more special tokens featuring layout characteristics of the related text fragment. The set of layout tokens includes "bold", "center", "capitalized", "single_word", "double_word", "triple_word", and "mutiple_word" etc. A set of layout functions are designed to represent the characteristics. For example,

$$f^L_{bold}(x_n) = \begin{cases} 1, & x_n \text{ is bold,} \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

captures the characteristic of "bold" of the text fragment x_n. Take Figure 1.1 as an example. "General" and "Image Sensor" are both bold in layout format, therefore $f^L_{bold}(\text{"General"}) = 1$. Consider another example, "Color Black" has two words within the text fragment, so we have $f^L_{double\_word}(\text{"Color Black"}) = 1$ while $f^L_{single\_word}(\text{"Color Black"}) = 0$, $f^L_{triple\_word}(\text{"Color Black"}) = 0$ and $f^L_{multiple\_word}(\text{"Color Black"}) = 0$. In our framework, the layout information will collaborate with content information to tackle product attribute extraction and normalization problems.

We construct a dictionary of tokens based on a small quantity of text fragments. Since the raw data consists of attribute-relevant data as well as large amount of attribute-irrelevant data, we set up a simple filtering mechanism that only tokens appear more than once will be taken into consideration, while those show up only once will be treated as attribute-irrelevant and skipped. Our framework employ a stemmer [23] for reducing inflected words to their stem, base or root form. For example, "interfaces" and "included" should be reduced to the root words "interface" and "include" respectively. Our framework is then applied to these identified text fragments for attribute extraction and normalization.

3.2.2 Overview of Our Framework

We employ graphical models to develop our framework. Generally, probabilistic graphical models [18] use a graph-based representation as the foundation for encoding a complete distribution over a multi-dimensional space. The nodes in the graph models correspond to random variables, and joint probability distributions are defined by taking products over functions defined on connected subsets of nodes. By exploiting the graph-theoretic representation, the formalism provides general algorithms for computing marginal and conditional probabilities of interest. Therefore, it is effective for graphical models to formulate probabilistic models of complex interactions among the elements in the problem. These characteristics make graphical models widely used in probability theory, Bayesian statistics, and machine learning. More detailed background of graphical models is described in Section 3.2.3.

Dirichlet process [5] is employed in our framework. Dirichlet process can provide an explicit construction of the non-parametric models and support discrete samples. Another important characteristic of Dirichlet process is that the number of mixture components is not required to provide in advance [38]. All the properties make it suitable for tackling our problem in hands among various existing clustering approaches, such as K-Means, linkage metrics, and density-based connectivity [2]. The background of Dirichlet process is described in the Appendix A.

Our framework also employs Hidden Markov Models(HMM). HMM can label the tokens of each text fragment as an attribute field. We set up a unique HMM model for each cluster based on the distribution of tokens within each reference attribute, so that tokens with higher probability belonging to one cluster would also share higher probability to be generated for the same text fragment. The background of HMM is described in the Appendix B.

3.2.3 Background of Graphical Models

A graphical model is a probabilistic model for which a graph denotes the conditional independence structure between random variables. Two branches of graphical representations of distributions are commonly used, namely, directed graphical models and undirected graphical model. These two branches are based on directed acylic graphs and undirected graphs, respectively. Since

our thesis uses the directed graphical models, the following introduction will only focus on this pattern.

Let $\mathcal{G}$ be a directed acyclic graph, then the model represents a factorization of the joint probability of all random variables. Let $Z_1, Z_2, \ldots, Z_n$ be a random variable indexed by the nodes of the graph. Also, let θ denote the set of random variables indexed by the parents of Z. Given that $P(Z_1|\theta_1), P(Z_1|\theta_1), \ldots, P(Z_n|\theta_n)$ sum or integrate to one, a joint probability distribution can be defined as

$$P(Z_1, Z_2, \ldots, Z_n) = \prod_{i=1}^{n} P(Z_i|\theta_i) \tag{3.5}$$

In other words, the joint distribution factors into a product of conditional distributions. Any two nodes are conditionally independent given the values of their parents.

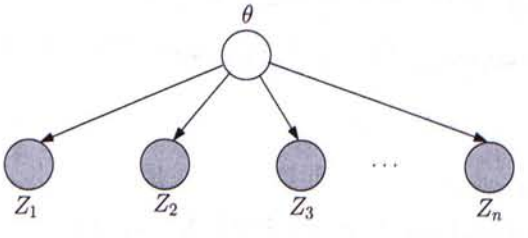


Figure 3.3: A simple example of a graphical model

Figure 3.3 illustrates a graphical model asserting that the variables Z_n are conditionally independent and identically distributed given θ. Generally, the graphical models use a *plate* to capture replication. Figure 3.4 is a shorthand

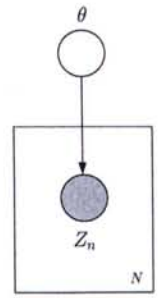


Figure 3.4: A shorthand for the graphical model in Figure 3.3

for the graphical model of Figure 3.3 using plate representation.

The graph provides an appealing visual representation of a joint probability distribution. Regardless of the form of the probability functions, the factorization in Equation 3.5 implies a set of conditional independence statements among the variables. The entire set of conditional independence statements can be obtained from a polynomial time reachability algorithm based on the graph.

The graphical structure can be exploited by algorithms for probabilistic inference [19]. Let (O, U) be a partitioning of the node indices of a graphical model into disjoint subsets, such that (X_O, X_U) is a partitioning of the random variables. Let O and U be the set of observable variables and the set of unobservable variables, respectively. In this thesis, inference tries to solve the problem of maximum a posteriori (MAP) probabilities:

$$p^*(x_O) = \max_{x_U} p(x_O, x_U) \tag{3.6}$$

From these basic computations we can obtain other quantities of interest. The graphical structure can make this computations efficient.

Chapter 4

Our Proposed Framework

4.1 Our Proposed Graphical Model

Our proposed framework is based on a specially designed graphical model as depicted in Figure 4.1. Shaded nodes and unshaded nodes represent the observable and unobservable variables respectively. The edges represent the dependence between variables and the plates represent the repetition of variables. Figure 4.2 illustrates the meaning of each notation used in our framework.

We employ Dirichlet process prior to tackle our problem. Each mixture component refers to a reference attribute in our framework. As a result, our framework can handle unlimited number of reference attributes. Essentially, our framework can be viewed as a mixture model containing unlimited number of components with different proportion. Each component refers to a reference attribute in the domain. Suppose we have a collection of N differ-

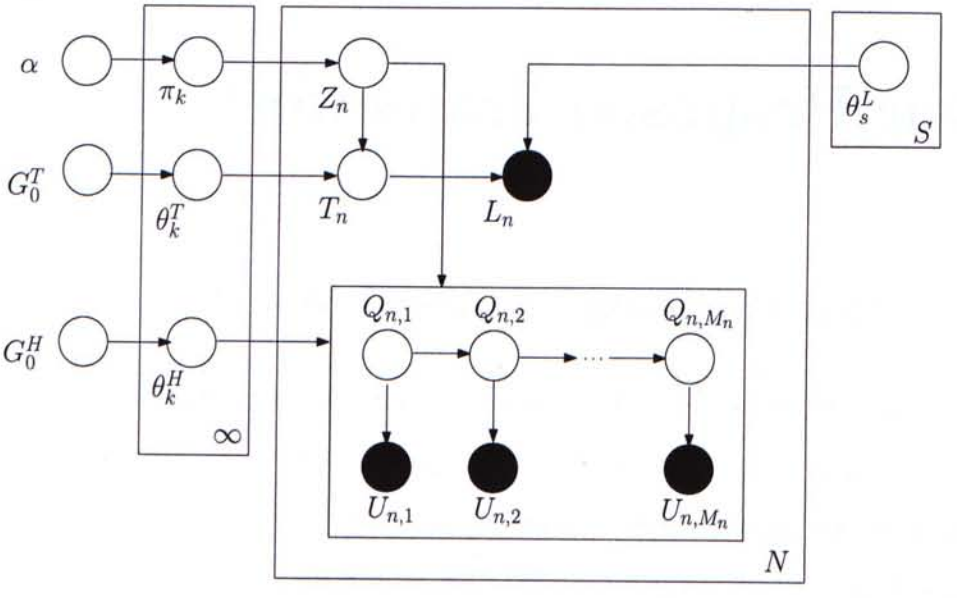


Figure 4.1: The graphical model for the generation of text fragments in web pages

ent text fragments collected from S different web sites. Each generation of a text fragment is modeled as an independent and identical event. The n-th text fragment x_n consists of an unobservable variable Z_n representing the index of the mixture component from which the underlying text fragment is generated. Essentially, A_n is replaced with Z_n for clarity and $A_n = a_{z_n}$ where $a_i \in \mathcal{A}$. We also employ Hidden Markov Models (HMM) to predict the label of each token of the N text fagments. As mentioned in Section 3.1, we use three kinds of labels in this thesis, i.e. "attribute-name", "attribute-value" and "attribute-irrelevant". Suppose there are M_n tokens of the n-th text fragment. We assume that each mixture component consists of an individual HMM. Hence through the variable Z_n we can find the corresponding HMM of the n-th text fragment for labeling its tokens. The token information U_n, also known as the page-independent content information, is then generated according to $P^H(U_n|Q_n, Z_n, \theta_k^H)$, where $P^H(\cdot|Q_n, Z_n, \theta_k^H)$ is the probability distribution about the token information U_n given the variables Q_n, Z_n and θ_k^H. U_n represents the sequence of tokens $U_{n,1}, U_{n,2}, \ldots, U_{n,M_n}$, while Q_n represents the label information of tokens $Q_{n,1}, Q_{n,2}, \ldots, Q_{n,M_n}$. θ_k^H refers to the set of parameters of the k-th HMM model.

Next, the target information T_n is generated by $P^T(T_n|\theta_{z_n}^T)$, where $P^T(\cdot|\theta_k^T)$ is the probability distribution about the target information T given the variable θ_k^T. Since the layout format of the text fragments in a web page is page-dependent, we have a set of layout distributions, namely, θ_s^L, for generating the page-dependent layout format of the text fragments in the page s. As shown in the running example in Section 1.3, there is mutual coop-

N	The number of text fragments
S	The number of web sites
x_n	The n-th text fragment
M_n	The number of tokens of the n-th text fragment
Z_n	The "index" of the parameters, stating the cluster from which the text fragment comes
U_n	The tokens information $U_{n,1}, U_{n,2}, \ldots, U_{n,M_n}$ of the n-th text fragment
Q_n	The label information of tokens $Q_{n,1}, Q_{n,2}, \ldots, Q_{n,M_n}$
W_n	The set of tokens $U_{n,1}, U_{n,2}, \ldots, U_{n,M_n}$ and labels $Q_{n,1}, Q_{n,2}, \ldots, Q_{n,M_n}$ of the n-th text fragment
T_n	The target variable illustrating whether the text fragment is related to product attribute
L_n	The layout of the n-th text fragment, indicating whether it has or has not some particular layout
π_k	The proportion of the kth component in the mixture
θ_k^T	A set of binomial distribution parameters for generating T_n
θ_k^H	The set of parameters of the k-th HMM model
θ_s^L	A set of site-dependent parameters controlling the layout format of each text fragment on the page
α	The parameter denoted in the stick breaking of Dirichlet process
G_0^T	The hyper parameter, or prior process to generate θ_k^T
G_0^H	The hyper parameter, or prior process to generate θ_k^H

Figure 4.2: Notations Used in Our Framework

eration between the layout information and the target information of a text fragment. T_n together with θ_s^L will generate the page-dependent layout information L_n of the n-th text fragment according to $P^L(L_n|T_n, \theta_{s(x_n)}^L)$, where $P^L(\cdot|T_n, \theta_s^L)$ is the probability distribution about the layout information L given the variables T_n and θ_s^L and $s(x_n)$ denotes the web page from which x_n is collected.

In ordinary Dirichlet mixture models, each mixture component consists of a distribution to characterize the data. Instead, our framework consists

of two different distributions parameterized by θ_k^T and θ_k^H for the k-th component. θ_k^T and θ_k^H are in turn generated from the base distributions G_0^T and G_0^H respectively in the Dirichlet process. G_0^T and G_0^H act as the prior distributions of the target information and the component-relevant HMM information respectively. For example, suppose we model the target information of the text fragments. Since T is a binary variable, it can be modeled as a Bernoulli trial. Therefore, $P^T(\cdot|\theta_k^T)$ can be a binomial distribution with parameter θ_k^T and G_0^T can be a Beta distribution, which is the conjugate prior of a binomial distribution. Similarly, G_0^H can be can be a Dirichlet distribution which is the conjugate prior of a mixture model, $P^H(\cdot|\theta_k^H)$ is a multinomial distribution, and θ_k^H is the set of parameters of multinomial distribution in component k.

We adopt the stick breaking construction representation of Dirichlet process prior in the graphical model depicted in Figure 4.1. In summary, we can break a one-unit length stick for an infinite number of times. Each time, we break a π_k portion from the remaining portion of the stick according to $Beta(1, \alpha)$ in the k-th break, where $Beta(\alpha_1, \alpha_2)$ is the Beta distribution, with parameters α_1 and α_2. Therefore, the k-th piece of the broken sticks can represent the proportion of k-th component in the mixture. Dirichlet process prior can support an infinite number of mixture components, which refer to the reference attributes in our framework. Z_n is then drawn from the distribution π. In summary, the generation process can be described as follows:

$$
\begin{aligned}
&\tilde{\pi_k}|\alpha \sim Beta(1,\alpha) \qquad \pi_k = \tilde{\pi_k}\prod_{i=1}^{k-1}(1-\tilde{\pi_k}) \\
&\theta_k^T|G_0^T \sim G_0^T \qquad \theta_k^H|G_0^H \sim G_0^H \\
&Z_n|\pi \sim \pi \\
&T_n|\theta_k^T \sim P^T(\theta_{Z_n}^T) \\
&U_n|Q_n, Z_n, \theta_k^H \sim P^H(U_n|Q_n, Z_n, \theta_k^H) \\
&L_n|T_n, \theta_s^L \sim P^L(L_n|T_n, \theta_{s(x_n)}^L)
\end{aligned}
\tag{4.1}
$$

The joint probability for generating a particular text fragment x_n given the parameters α, G_0^T , G_0^H, and θ_s^L can then be expressed as follows:

$$
\begin{aligned}
&P(U_n, Q_n, Z_n, L_n, T_n, \pi_1, \pi_2, \ldots, \theta_1^T, \theta_2^T, \ldots, \theta_1^H, \theta_2^H, \ldots|\alpha, G_0^T, G_0^H, \theta_s^L) \\
&= \prod_{i=1}^{\infty}\{P^L(L_n|T_n, \theta_{s(x_n)}^L)[P^T(T_n|Z_n, \theta_i^T)P^H(U_n|Q_n, Z_n, \theta_i^H)]^{\chi_{\{Z_n=i\}}} \\
&P(Z_n = i|\pi_1, \pi_2, \ldots)P(\theta_i^T|G_0^T)P(\theta_i^H|G_0^H)\}\prod_{i=1}^{\infty}P(\pi_i|\alpha, \pi_1, \ldots, \pi_{i-1})
\end{aligned}
\tag{4.2}
$$

where

$$
\begin{aligned}
&P^H(U_n|Q_n, Z_n, \theta_k^H) \\
&= \prod_{m=1}^{M_n}[P(u_{n,m}|q_{n,m}, Z_n, \theta_k^H)P(q_{n,m}|q_{n,m-1}, Z_n, \theta_k^H)]
\end{aligned}
\tag{4.3}
$$

and $\chi_{\{Z_n=i\}} = 1$ if $Z_n = i$ and 0 otherwise.

4.2 Inference

As described above, Equation 4.3 provides the basic formulation of the graphical model. For simplicity, we let $\boldsymbol{O}$, $\boldsymbol{U}$, and $\boldsymbol{\varphi}$ be the set of observable variables, which include all L_n and U_n, where $1 \leq n \leq N$, the set of unobservable variables, which include all Z_n, T_n, θ_k^T, θ_k^H and π_k, where $1 \leq n \leq N$ and $1 \leq k \leq \infty$, and the set of model parameters, which include α, G_0^T, G_0^H,

and θ_s^L respectively. Given a set of text fragment and the parameters $\boldsymbol{\varphi}$, the unsupervised learning problem can be viewed as an inference problem defined as follows:

$$\begin{aligned} U^* &= \arg\max_u \{P(\boldsymbol{U} = u|\boldsymbol{O}, \boldsymbol{\varphi})\} \\ &= \arg\max_u \{\log P(\boldsymbol{U} = u|\boldsymbol{O}, \boldsymbol{\varphi})\} \end{aligned} \tag{4.4}$$

Since the computation of $\log P(\boldsymbol{U}|\boldsymbol{O}, \boldsymbol{\varphi}) = \log \frac{\int P(\boldsymbol{U}, \boldsymbol{O}|\boldsymbol{\varphi})_d \boldsymbol{O}}{P(\boldsymbol{O}|\boldsymbol{\varphi})}$ involves the marginalization of $P(\boldsymbol{U}, \boldsymbol{O}|\boldsymbol{\varphi})$, over the unobservable variables, exactly solving Equation 4.4 is intractable. As a result, approximation methods are required. In this thesis, we make use of Markov Chain Monte Carlo (MCMC) techniques [26] to solve this problem in a principled and efficient manner.

Due to the difficulty of direct sampling, Metropolis-Hastings algorithm [9] may be considered. The Metropolis-Hastings algorithm can draw samples from any probability distribution $P(x)$, requiring only that a function proportional to the density can be calculated. The algorithm uses a proposal density $Q(x'; x^t)$, which depends on the current state x^t, to generate a new proposed sample x'. This proposal is "accepted" as the next value ($x^{t+1} = x'$) if α drawn from satisfies

$$\alpha < \min\{\tfrac{P(x')Q(x^t;x')}{P(x^t)Q(x';x^t)}, 1\} \tag{4.5}$$

If the proposal is not accepted, the current value of x is retained: $x^{t+1} = x^t$. For example, the proposal density could be a Gaussian function centered on the current state x^t: $Q(x'; x^t) \sim N(x^t, \sigma^2 I)$.

However, in our graphical model, new density forms are available when

the components Z_n, T_n, L_n, U_n and Q_n are used seperatedly. In this case, Metropolis-Hastings algorithm needs to handle multi-component situation. It requires repeatedly sampling different components and would be very time consuming. This inspires us looking for another algorithm that cost less time on computation. The goal of our framework is to generate a Markov chain with stationary distribution $f(\boldsymbol{U}, \boldsymbol{O}, \boldsymbol{\varphi})$. Since the conjugate priors are used in our model, we adopt Gibbs sampling [13] to sample from the posterior distribution $P(\boldsymbol{U}|\boldsymbol{O}, \boldsymbol{\varphi})$. Based on the sampling process, we can determine how many distinct components are likely contributing to our data and what the parameters are for each component. It avoids sampling from all the relevant variables in the framework as that mentioned in Section 4.1, hence saves time on computation.

Figure 4.3 depicts the high-level outline of our inference algorithm. We sample for the component indicator Z_n for the n-th text fragment as well as the component parameters θ_k^T and θ_k^H, for all $1 \leq k \leq \infty$. Assuming current state of the Markov chain in MCMC algorithm consists of $Z_1, Z_2, \ldots$, and the component parameters θ_k^T and θ_k^H, for all $1 \leq k \leq \infty$. For convenience, we use a variable W_n to represent the set of tokens $U_{n,1}, U_{n,2}, \ldots, U_{n,M_n}$ and labels $Q_{n,1}, Q_{n,2}, \ldots, Q_{n,M_n}$ of the n-th text fragment. Samples can be generated by repeating the following steps:

1. For $i = 1, \ldots, N$:

   - If Z_i is currently a singleton, remove $\theta_{Z_i}^T$ and $\theta_{Z_i}^H$ from the state.

   - Draw a new value for Z_i from the conditional distribution:

 # Unsupervised inference algorithm

INPUT: $\boldsymbol{X}$: The set of text fragments from different Web pages

OUTPUT: Z_n, T_n and U_n for all $x_n \in \mathcal{X}$

INIT:

0 set all model parameters as uninformative prior
1 until convergence
2 foreach $x_n \in \mathcal{X}$
3 sample Z_n according to Equations 4.6 and 4.7
4 update θ_k^T and θ_k^H for all k according to Equation 4.8
5 update T_n according to Equation 4.9
6 learn the HMM model corresponding to x_n and update θ_k^H using Baum-Welch algorithm
7 use the learned HMM model to label the text fragments using Viterbi algorithm
8 end foreach
9 end until

Figure 4.3: A high-level outline of our unsupervised inference algorithm.

$$P(Z_i = z | Z_{-i}, T_i, W_i, \theta^T_{Z_i}, \theta^H_{Z_i}) = \begin{cases} \frac{N_{-i,c}}{N-1+\alpha} F(\theta_i^T, T_i) F(\theta_i^H, W_i), & \text{for existing } z, \\ \frac{\alpha}{N-1+\alpha} \int F(\theta^T, T_i) dG_0^T F(\theta^H, W_i) dG_0^H, & \text{for a new } z \end{cases} \tag{4.6}$$

- If the new Z_i is not associated with any other observation, draw a value for $\theta^T_{Z_i}$ and $\theta^H_{Z_i}$ from:

$$\begin{aligned} P(\theta^T | T_i) &\propto F(\theta_i^T, T_i) G_0^T(\theta^T) \\ P(\theta^H | W_i) &\propto F(\theta_i^H, W_i) G_0^H(\theta^H) \end{aligned} \tag{4.7}$$

2. For all $1 \leq k \leq \infty$:

- Draw a new value for θ_k^T and θ_k^H from the posterior distribution based on the prior G_0^T and G_0^H and all the data points currently associated with component k:

$$\begin{aligned} P(\theta^T|T_k) &\propto \prod_{i:Z_i=k} P(T_i|\theta^T)P(\theta^T) \\ &= \prod_{i:Z_i=k} F(\theta^T, T_i)G_0^T(\theta^T) \\ P(\theta^H|W_k) &\propto \prod_{i:Z_i=k} P(W_i|\theta^H)P(\theta^H) \\ &= \prod_{i:Z_i=k} F(\theta^H, W_i)G_0^H(\theta^H) \end{aligned} \tag{4.8}$$

As mentioned in Section 4.1, T_i can be modeled as a Bernoulli trial, let G_0^T be a Beta distribution, which is the conjugate prior of a binomial distribution, then $P(\cdot|\theta_k^T)$ can be modeled as a binomial distribution with the parameter θ_k^T. So the posterior probability $P(\theta^T|T_k)$ is also a Beta distribution. Similarly, let G_0^H be can be a Dirichlet distribution, which is the conjugate prior of a mixture model then $P(\cdot|\theta^H)$ can be modeled as a multinomial distribution with parameter θ^H and its posterior probability $P(\theta^H|W_k)$ is a Dirichlet distribution.

As exemplified in Section 1.3, our framework can consider the page-dependent layout format of text fragments to enhance extraction. Considering the fact that web pages within one web site usually share the same set of layout information, we use a set of parameters θ_s^L to represent the layout information. Therefore, $P(\cdot|\theta_s^L, T_n)$ can be modeled as a multinomial distribution. Given θ_s^L, we can update T_n based on the $P(T_n|\theta_s^L, T_n)$.

$$P(T_n|L_n) \propto P(L_n|\theta_s^L, T_n)P(T_n)P(\theta_s^L) \tag{4.9}$$

After updating θ^H for all the components, the n-th text fragment will be labeled by the corresponding HMM generated from the k-th component, where $Z_n = k$. θ_k^H contains a set of HMM parameters: the start probability representing at which label the HMM starts, the transition probability representing the change of labels in the underlying Markov chain, and the emission probability representing which token would be generated by each label. We conduct Baum-Welch algorithm to derive the maximum likelihood estimate and update the set of probabilities. And the text fragment, also known as a token sequence, is labeled using Viterbi algorithm based on the updated parameters of the model.

To initialize this algorithm, we need to provide the parameters α, G_0^T, G_0^H, and θ_s^L. For the model parameters, α is the scaling parameter in the Dirichlet process, which essentially affects the number of normalized attributes in the normalization process. Since we apply our framework to the domains, for example, digital cameras, in which each product contains a number of attributes, we set α to a value that favors a large number of extracted attributes. G_0^T refers to the prior knowledge about how likely a text fragment will be a product attribute. We treat it as an uninformative prior by letting $\alpha = 1, \beta = 1$ of a Beta distribution. Similarly, G_0^H is treated as uninformative and all α's of a Dirichlet distribution are set to 1. θ_s^L can also be initialized in this way.

4.3 Product Attribute Information Determination

As mentioned in Section 4.1, the integration of HMM enables our framework to label tokens of text fragments as "attribute-name", "attribute-value" or "attribute-irrelevant". To determine whether a text fragment should be extracted or not, in other words, whether it is relevant to a reference attribute or not, we design a procedure to achieve this task. The threshold e_1 decides whether a text fragment is "attribute-relevant" or "attribute-irrelevant". Let p_1 represent the proportion of tokens labeled as "attribute-irrelevant" within a sequence of a text fragment. When $p_1 < e_1$ within a sequence, the whole sequence will be considered as "attribute-relevant" for subsequent processing. When $p_1 > e_1$, the sequence will be considered as "attribute-irrelevant". The threshold e_2 decides whether a cluster is "attribute-relevant" or "attribute-irrelevant". Let p_2 represent the threshold of the proportion of predicted "attribute-irrelevant" text fragments within a cluster. When $p_2 > e_2$ within a cluster, the whole cluster will be considered as "attribute-irrelevant". In this case, all the text fragments that exist in this cluster, regardless "attribute-relevant" or "attribute-irrelevant", are treated as "attribute-irrelevant".

 # Product Attribute Information Determination

INPUT: $\boldsymbol{X}$: The set of text fragments from different web pages

OUTPUT: Labels ("attribute-relevant" or "attribute-irrelevant") for all $x_n \in \mathcal{X}$

INIT:

0 execute unsupervised inference algorithm for the set of text fragments and get HMM lables
1 set the threshold e_1 and e_2
2 foreach cluster k
3 foreach $x_{n,k} \in \mathcal{X}_k$
4 calculate p_1 of x_n
5 if $p_1 < e_1$
6 label $x_{n,k}$ as "attribute-relevant"
7 else
8 label $x_{n,k}$ as "attribute-irrelevant"
9 end foreach
10 calculate p_2 for cluster k
11 if $p_2 > e_2$
12 label all $x_{n,k}$ as "attribute-relevant"
13 end foreach

Figure 4.4: An outline of product attribute information determination.

Chapter 5

Experiments and Results

We have conducted several sets of experiments to evaluate our framework. The dataset used in our experiments is composed of data from three different domains, namely, digital camera, MP3 player, and LCD TV domains. For these domains, a set of web pages were collected from different web sites, which were randomly selected by making use of product search engines. One human accessor were invited to prepare the ground truth of the data for evaluation. Each Web page was first pre-processed to generate a set of text fragments as described in Section 3.2.1. The attribute name and the attribute value, if any, of the text fragment will be identified. Note that such annotation is only used for evaluation purpose. Table 5.1 summarizes the information of the dataset. The first and second column of the table shows the total number of web pages and the total number of web sites from which the data is collected. The third column shows the total number of text fragments in all the web pages after pre-processing. The fourth column shows the total number of text fragments about product attributes in all the pages.

Domain	No. of pages	No. of sites	No. of text fragments	No. of text fragments related to attributes
Digital Camera	50	21	5696	690
MP3 Player	59	13	5040	572
LCD TV	61	15	3014	270

Table 5.1: A summary of the data used in the experiments collected from the digital camera, MP3 player, and LCD TV domains.

In each domain, we conducted two sets of experiments. In the first set of experiments, we applied our framework to jointly extract and normalize the product attributes from all the web pages in the domain. We call this set of experiments "Our Approach". The second set of experiment employs latent Dirichlet allocation (LDA) [6] to cluster text fragments instead of Dirichlet process. Since the number of components has to be given to LDA model in advance, here we set $K = 50$. Then HMM models are applied to labeling the text fragments. This set of experiments can be considered as a comparison and called "LDA Approach". Note that this approach asks for knowledge about the number of product attributes in advance, while the "Our Approach" can handle unlimited number of product attributes. We conducted several runs in each set of experiments using different parameters of the model. The performance of both extraction and normalization in each run were recorded for evaluation.

We first evaluate the performance of product attribute extraction. The

attribute extraction results from each set of experiment are compared with the ground truth prepared by the human accessor. Extraction precision and recall are adopted as the evaluation metrics. Extraction recall is defined as the number of text fragments correctly extracted text fragments corresponding to a product attribute divided by the actual number of text fragments corresponding to a product attribute. Extraction precision is defined as the number of correctly extracted text fragments corresponding to a product attribute divided by the total number of text fragments extracted by the system. Extraction F_1-measure is defined as the harmonic mean of equal weighting of extraction recall and precision. Similarly, we use precision and recall to evaluate labeling results. For each label, i.e. "attribute-name" and "attribute-value", precision is defined as the number of tokens correctly predicted as "attribute-name" or "attribute-value", divided by the total number of tokens which are predicted as "attribute-name" or "attribute-value". Recall is defined as the number of tokens correctly predicted as "attribute-name" or "attribute-value", divided by the actual number of tokens which are judged by the human accessor as "attribute-name" or "attribute-value". Also F_1-measure is defined as the harmonic mean of equal weighting of extraction recall and precision.

Table 5.2 shows the the product attribute extraction performance of "Our Approach" and "LDA Approach" respectively. Each column refers to the extraction performance in a domain depicted in the first row. Each row of the tables corresponds to an set of experiment result using "Our Approach" and "LDA Approach". "Our Approach" obtains better results compared with the

		DC			MP3			LCD TV		
		P	R	F	P	R	F	P	R	F
Our Approach	Extraction	0.476	0.610	0.535	0.294	0.575	0.393	1.000	0.376	0.547
	Label "attribute-name"	0.599	0.554	0.557	0.630	0.619	0.625	0.549	0.407	0.450
	Label "attribute-value"	0.804	0.580	0.629	0.837	0.533	0.630	0.893	0.716	0.784
LDA Approach	Extraction	0.476	0.610	0.535	0.294	0.575	0.393	1.000	0.330	0.497
	Label "attribute-name"	0.628	0.545	0.558	0.515	0.545	0.529	0.519	0.373	0.409
	Label "attribute-value"	0.476	0.610	0.535	0.789	0.471	0.557	0.883	0.712	0.777

Table 5.2: The attribute extraction performance of "Our Approach" and "LDA Approach" on the digital camera (DC), MP3 player (MP3), and LCD TV domains. P, R, and F refer to the recall, precision, and F_1-measure respectively.

"LDA Approach". In particular, the F_1-measure of label "attribute-value" are 0.629, 0.625 and 0.784 in the digital camera, MP3 players and LCD TV domains respectively. "LDA Approach" obtains a relatively low recall and precision compared with "Our Approach". Since the text fragments contains more tokens that are judged to be "attribute-value", e.g. there are several text fragments contains only "attribute-value" tokens, "Our Approach" appears more adaptable in handling the text fragments.

We have investigated the effect of the thresholds e_1 and e_2 of our framework. The results can achieve a stable performance when $e_1 = 0.1 - 0.15$ and $e_2 = 0.5 - 0.6$. In particular, we keep e_1 in a relatively low value. The major reason is to keep a certain level of filtering clusters. If e_1 was set to a higher value, i.e. above 0.2, the framework would treat almost all the clusters as attribute-relevant. This also explains why the difference of the extraction F_1 of "Our Approach" and "LDA Approach" is not obvious. Although the

clusters generated by these two methods are different, the presence of most clusters for HMM labeling and evaluation leads to an indifferent extraction F_1.

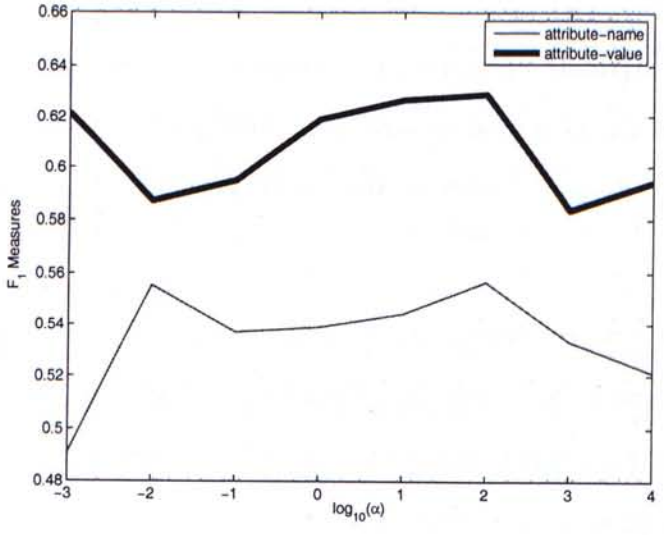


Figure 5.1: The effect of α in Dirichlet process on the extraction performance

Figure 5.1 illustrates the effect on F_1 measures for label "attribute-name" and "attribute-value" under different α of Dirichlet process. α affects changes of the clustering results as well as the labeling performance. From Figure 5.1 we can see that the performance is quite stable while α changes within a relatively wide range. It means that we can choose the value of α from a wide range instead of sticking to a certain value. In our experiment we pick $\alpha = 100$, with which the performance gets a relatively high value and the

Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
memory	iso	battery	dimension	lcd
card	resolution	rechargeable	height	color
optical	effective	include	width	tft
build	image	alkaline	weight	screen
expandable	megapixel	lithium	length	display
Att. 6	Att. 7	Att. 8	Att. 9	Att. 10
white	iso	box	red	scene
balance	speed	open	eye	auto
flash	additional	usb	fix	mode
daylight	burst	connect	reduce	select
cloudy	high	cable	flash	portrait

Table 5.3: The visualization of the top five weighted terms in the ten largest normalized attributes in the digital camera domain.

difference between F_1 measures of "attribute-name" and "attribute-value" is relatively small.

We conduct a qualitative analysis on the attribute normalization by comparing the tokens that are generated in each cluster. Table 5.3 shows the top 5 weighted terms in 10 normalized attributes in the digital camera domain. It can be observed that the semantic meaning of the attributes can be easily interpreted from the terms. The output of attribute normalization can be very useful for supporting other intelligent applications such as product attribute indexing and product retrieval.

We also look into how α affects the number of clusters. Figure 5.2 illus-

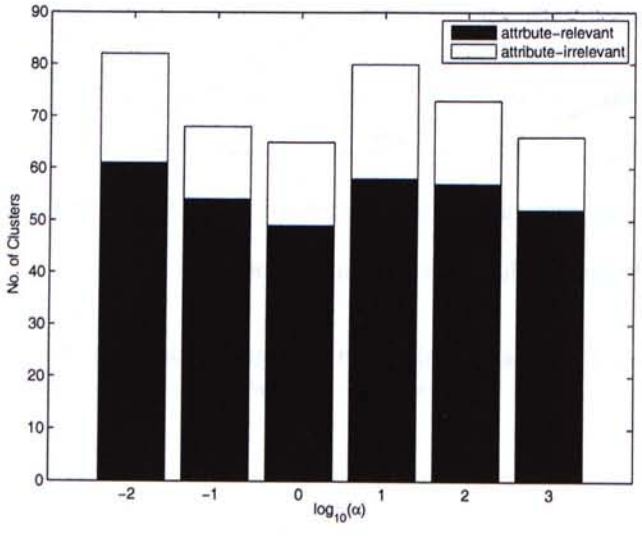


Figure 5.2: The effect of α in Dirichlet process on the number of "attribute-relevant" and "attribute-irrelevant" clusters

trates the effect of α on the number of clusters. The number of "attribute-relevant" clusters as well as "attribute-irrelevant" clusters fluctuate within a reasonably small range when α changes. This also strengthens the conclusion that the value of α can be chosen within a wide range. Notice that we do not need to provide the exact number of clusters as the "LDA Approach" did. Dirichlet process will automatically adapt the size of clusters and the number of clusters according to the data set. Our framework performs more adaptable than "LDA Approach" since it is difficult to choose the number of cluster in advance.

Chapter 6

Conclusion

We have developed an unsupervised framework which aims at simultaneously extracting and normalizing product attributes from a number of web pages collected from different sites. We have proposed a graphical model, which employs Dirichlet process prior, to model the generation of text fragments in Web pages. This leads to a property that our framework can handle an unlimited number of product attributes. An unsupervised inference algorithm based on MCMC is derived. We show that the page-independent content information and the page-dependent layout information can collaborate and improve both extraction and normalization performance. An HMM model is integrated in our framework to label tokens for further uses. Extensive experiments on real-world data have been conducted to show the robustness and effectiveness of our approach. We also proposed potential applications that can make use of the results of attribute extraction and normalization.

We plan to extend our framework to several directions. One possible

direction is to develop a method to automatically construct the ontology of a domain based on the normalized attributes. Since each normalized attribute can be represented by a set of terms in our framework, we plan to develop an automatic method to organize the attributes into a hierarchical structure, enriching the expressiveness of the extracted information. Another direction is to utilize the extracted and normalized attributes for automatic product record matching. There are numerous different products being sold in online stores. Automatic product matching can compute a score between products considering attribute-wise information. This is useful for users to analyze the products make decision.

Bibliography

[1] C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152 – 1174, 1974.

[2] P. Berkhin. A survey of clustering data mining techniques. *Grouping Multi-dimensional Data: Recent Advances in Clustering*, pages 25 – 71, 2006.

[3] I. Bhattacharya and L. Getoor. A latent Dirichlet model for unsupervised entity resolution. In *In Proceedings of the 2006 SIAM International Conference on Data Mining (SDM)*, 2006.

[4] D. Blackwell and J. MacQueen. Ferguson distributions via Pólya Urn schemes. *Annals of Statistics*, pages 353 – 355, 1973.

[5] D. Blei and M. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121 – 144, 2006.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993 – 1022, 2003.

[7] C. Chang and S. C. Lui. IEPAD: information extraction based on pattern discovery. In *Proceedings of the Tenth International Conference on World Wide Web (WWW)*, pages 681–688, 2001.

[8] C.-H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan. A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428, 2006.

[9] S. Chib and E. Greenberg. Understanding the MetropolisHastings algorithm. *American Statistician*, 49(4):327 – 335, 1995.

[10] S. L. Chuang, K. Chang, and C. Zhai. Context-aware wrapping: Synchronized data extraction. In *Proceedings of the Thirty-Third Very Large Databases Conference*, pages 699–710, 2007.

[11] V. Crescenzi, G. Mecca, and P. Merialdo. Towards automatic data extraction from large web sites. In *Proceedings of the Twenty-Seventh Very Large Databases Conference*, pages 109–118, 2001.

[12] T. Ferguson. Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209 – 230, 1973.

[13] C. George and G. Edward I. Explaining the Gibbs sampler. *American Statistician*, 46(3):167 – 174, 1992.

[14] P. Golgher and A. da Silva. Bootstrapping for example-based data extraction. In *Proceedings of the Tenth ACM International Conference on Information and Knowledge Management (CIKM)*, pages 371–378, 2001.

[15] T. Grenager, D. Klein, and C. Manning. Unsupervised learning of field segmentation models for information extraction. In *Proceedings of the Forty-Third Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 371–378, 2005.

[16] DOM Interest Group. Document Object Model (DOM).

[17] R. Hall, C. Sutton, and A. McCallum. Unsupervised deduplication using cross-field dependencies. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2008.

[18] M. Jordan. Graphical models. *Statistical Science, Special Issue on Bayesian Statistics*, 19:140 – 155, 2004.

[19] M. I. Jordan. Probabilistic inference in graphical models. *The Handbook of Brain Theory and Neural Networks*, 2002.

[20] N. Kushmerick. Wrapper induction for information extraction. In *PhD Thesis. University of Washinton*, 1997.

[21] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

[22] B. Liu, R. Grossman, and Y. Zhai. Mining data records in web pages. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 601–606, 2003.

[23] J. B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22 – 31, 1968.

[24] A. McCallum and D. Jensen. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *Proceedings of the IJCAI Workshop on Learning Statistical Models from Relational Data*, 2003.

[25] X. Meng, H. Wang, D. Hu, and C. Li. A supervised visual wrapper generator for web-data extraction. pages 657–662, 2003.

[26] R. M. Neal. Probabilistic inference using Markov Chain Monte Carlo methods. Technical report, Dept. of Computer Science, University of Toronto, 1993.

[27] K. Probst, M. K. R. Ghai, A. Fano, and Y. Liu. Semi-supervised learning of attribute-value pairs from product descriptions. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2838–2843, 2007.

[28] Lawrence R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257 – 286, 1989.

[29] S. Sarawagi and W Cohen. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 17, Neural Information Processing Systems (NIPS)*, 2004.

[30] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, pages 639 – 650, 1994.

[31] P. Singla and P. Domingos. Entity resolution with Markov logic. In *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM)*, pages 572 – 582, 2006.

[32] C. Sutton, K. Rohanimanesh, and A. McCallum. Dynamic conditional random fileds: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of Twenty-First International Conference on Machine Learning (ICML)*, pages 783–790, 2007.

[33] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, pages 1566 – 1581, 2005.

[34] J. Turmo, A. Ageno, and N. Catala. Adaptive information extraction. *ACM Computing Surveys*, 38(2), 2006.

[35] B. Wellner, A. McCallum, F. Peng, and M. Hay. An integrated, conditional model of information extraction and coreference with application to citation matching. In *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 593 – 601, 2004.

[36] the free encyclopedia Wikipedia. Online shopping.

[37] T.-L. Wong and W. Lam. Adapting web information extraction knowledge via mining site invariant and site depdent features. *ACM Transactions on Internet Technolog*, 7(1), 2007.

[38] T.-L. Wong, W. Lam, and T.-S. Wong. An unsupervised framework for extracting and normalizing product attributes from multiple web sites. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR)*, 2008.

[39] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and H.-W. Hon. Webpage understanding: an integrated approach. In *Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 903–912, 2007.

Appendix A

Dirichlet Process

The following description of the Dirichlet process is extracted from (Y. Teh, 2005) [33]. The Dirichlet process was formally introduced by Thomas Ferguson in 1973 [12]. Let (Θ, β) be a measurable space, with G_0 a probability measure on the space. Let α_0 be a positive real number. A Dirichlet process $DP(\alpha_0, G_0)$ is defined to be the distribution of a random probability measure G over (Θ, β) such that, for any finite measurable partition $(A_1, A_2, \ldots, A_r)$ of β, the random vector $(G(A_1), \ldots, G(A_r))$ is distributed as a finite-dimensional Dirichlet distribution with parameters $(\alpha_0 G_0(A_1), \ldots, \alpha_0 G_0(A_r))$:

$$(G(A_1), \ldots, G(A_r)) \sim Dir(\alpha_0 G_0(A_1), \ldots, \alpha_0 G_0(A_r)) \tag{A.1}$$

We write $G \sim DP(\alpha_0, G_0)$ if G is a random probability measure with distribution given by the Dirichlet process.

Measures drawn from a Dirichlet process are discrete with probability one [12]. This property is made explicit in the stick-breaking construction due to Sethuraman [30]. The stick-breaking construction is based on independent sequences of independent random variables $(\pi'_k)_{k=1}^{\infty}$ and $(\theta_k)_{k=1}^{\infty}$:

$$\pi'_k | \alpha_0, G_0 \sim Beta(1, \alpha_0) \qquad \theta_k | \alpha_0, G_0 \sim G_0, \tag{A.2}$$

Now define a random measure G as

$$\pi_k = \pi'_k \prod_{l=1}^{k-1} (1 - \pi'_l) G = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k}, \tag{A.3}$$

where δ_θ is a probability measure concentrated at θ. Sethuraman [30] showed that G as defined in this way is a random probability measure distributed

according to $DP(\alpha_0, G_0)$.

A second perspective on the Dirichlet process is provided by the Pólya urn scheme [4]. The Pólya urn scheme shows that draws from the Dirichlet process are both discrete and exhibit a clustering property. The Pólya urn scheme does not refer to G directly; it refers to draws from G. Thus, let $\theta_1, \theta_2, \ldots$ be a sequence of i.i.d. random variables distributed according to G. That is, the variables $\theta_1, \theta_2, \ldots$ are conditionally independent given G, and hence exchangeable. Let us consider the successive conditional distributions of θ_i given $\theta_1, \ldots, \theta_{i-1}$, where G has been integrated out. Blackwell and MacQueen [4] showed that these conditional distributions have the following form:

$$\theta_i|\theta_1, \ldots, \theta_{i-1}, \alpha_0, G_0 \sim \sum_{l=1}^{i-1} \frac{1}{i-1+\alpha_0}\delta_{\theta_l} + \frac{\alpha_0}{i-1+\alpha_0}G_0 \qquad \text{(A.4)}$$

We can interpret the conditional distributions in terms of a simple urn model in which a ball of a distinct color is associated with each atom. The balls are drawn equiprobably; when a ball is drawn it is placed back in the urn together with another ball of the same color. In addition, with probability proportional to α_0 a new atom is created by drawing from G_0 and a ball of a new color is added to the urn.

Expression A.4 shows that θ_i has positive probability of being equal to one of the previous draws. Moreover, there is a positive reinforcement effect; the more often a point is drawn, the more likely it is to be drawn in the future. To make the clustering property explicit, it is helpful to introduce a new set of variables that represent distinct values of the atoms. Define $\phi_1, \ldots, \phi_K$ to be the distinct values taken on by $\theta_1, \ldots, \theta_{i-1}$, and let m_k be the number of values θ_i' that are equal to ϕ_k for $1 \leq i' < i$. We can re-express A.4 as

$$\theta_i|\theta_1, \ldots, \theta_{i-1}, \alpha_0, G_0 \sim \sum_{k=1}^{K} \frac{m_k}{i-1+\phi_k}\delta_{\theta_l} + \frac{\alpha_0}{i-1+\alpha_0}G_0 \qquad \text{(A.5)}$$

The Dirichlet process mixture model [1] adds a level to the hierarchy by treating η_n as the parameter of the distribution of the nth observation. Given the discreteness of G, the DP mixture has an interpretation as a mixture

model with an unbounded number of mixture components. In particular, suppose that observations x_i arise as follows:

$$\begin{aligned} \theta_i|G &\sim G \\ x_i|\theta_i &\sim F(\theta_i), \end{aligned} \tag{A.6}$$

where $F(\theta_i)$ denotes the distribution of the observation x_i given θ_i. The factors θ_i are conditionally independent given G, and the observation x_i is conditionally independent of the other observations given the factor θ_i. When G is distributed according to a Dirichlet process, this model is referred to as a Dirichlet process mixture model. A graphical model representation of a Dirichlet process mixture model is shown in Figure A.1.

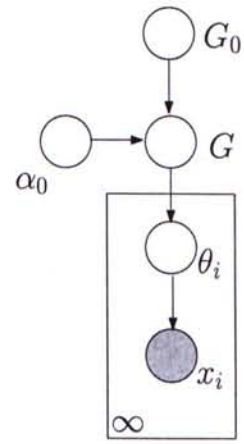


Figure A.1: A representation of a Dirichlet process mixture model as a graphical model. In the graphical model formalism, each node in the graph is associated with a random variable, where shading denotes an observed variable. Rectangles denote replication of the model within the rectangle.

Since G can be represented using a stick-breaking construction Equation A.3, the factors θ_i take on values ϕ_k with probability π_k. We may denote this using an indicator variable z_i which takes on positive integral values and is distributed according to π (interpreting π as a random probability measure on the positive integers). Hence an equivalent representation of a Dirichlet process mixture is given by the following conditional distributions:

$$\begin{aligned} \pi|\alpha_0 &\sim Stick(\alpha_0) \qquad z_i|\pi \sim \pi \\ \phi_k|G_0 &\sim G_0 \qquad x_i|z_i, (\phi_k)_{k=1}^{\infty} \sim F(\phi_{z_i}), \end{aligned} \tag{A.7}$$

Moreover, $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$ and $\theta_i = \phi_{z_i}$.

Appendix B

Hidden Markov Models

The following description of the hidden Markov model (HMM) is extracted from (R. Rabiner, 1989) [28].A hidden Markov model is a statistical model in which the system being modeled is assumed to be a Markov process with unobserved state. In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. Lawrence R. Rabiner [28] extended the concept of Markov models to include the case where the observation probabilistic function of the state, and the resulting model is called Hidden Markov Models. In a hidden Markov model, the state is not directly visible, but output, dependent on the state, is visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. Note that the adjective "hidden" refers to the state sequence through which the model passes, not to the parameters of the model; even if the model parameters are known exactly, the model is still "hidden".

Suppose there is a sequential data $u = u_1, u_2, \ldots, u_t, \ldots, u_T$, $u_t \in \Re^d$. As in the mixture model, every $u_t, t = 1, \ldots, T$, is generated by a hidden state, q_t. The diagram below shows the general architecture of an instantiated HMM. Each oval shape represents a random variable that can adopt any of a number of values. The random variable q_t is the hidden state at time t (with the model from the above diagram, $q = q_1, q_2, \ldots, q_t, \ldots, u_T$, $q_t \in \Re^d$). The random variable u_t is the observation at time t. The arrows in the diagram denote conditional dependencies. The underlying states follow a Markov chain. From the diagram, it is clear that the conditional probability distribution of the hidden variable q_t at time t, given the values of the hidden variable s at all times, depends only on the value of the hidden variable q_{t-1}: the values at time q_{t-2} and before have no influence. In other words, the

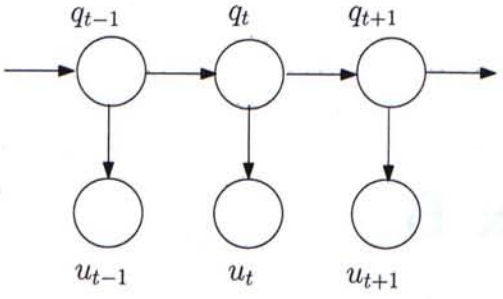


Figure B.1: The general architecture of an HMM

future is independent of the past:

$$P(q_{t+1}|q_t, q_{t-1}, \ldots, q_0) = P(q_{t+1}|q_t) \tag{B.1}$$

Therefore, the transition probability of HMM is defined as

$$a_{k,l} = P(q_{t+1} = l|q_t = k), \tag{B.2}$$

$k, l = 1, 2, \ldots, M$, where M is the total number of states. π_k represents the initial probabilities of states, so we have

$$\sum_{l=1}^{M} a_{k,l} = 1 \quad \text{for any } k, \sum_{k=1}^{M} \pi_k = 1 \tag{B.3}$$

Similarly, the value of the observed variable u_t only depends on the value of the hidden variable q_t (both at time t). For a fixed state, the observation u_t is generated according to a fixed probability law. Given state k, the probability law of U is specified by $b_k(u)$. In Discrete Markov Processes, suppose U takes finitely many possible values, $b_k(u)$ is specified by the probability mass function. In continuous cases, most often the Gaussian distribution is assumed like

$$b_k(u) = \frac{1}{\sqrt{(2\pi)^d |\sum_k|}} exp(-\frac{1}{2}(u - \mu_k)^t \sum_k^{-1}(u - \mu_k)) \tag{B.4}$$

In summary,

$$\begin{aligned} P(u, q) &= P(q)P(u|q) = \pi_{q_1} b_{q_1}(u_1) a_{q_1,q_2} b_{q_2}(u_2) \cdots a_{q_{T-1},q_T} b_{q_T}(u_T) \\ P(u) &= \textstyle\sum_q P(q)P(u|q) = \sum_q \pi_{q_1} b_{q_1}(u_1) a_{q_1,q_2} b_{q_2}(u_2) \cdots a_{q_{T-1},q_T} b_{q_T}(u_T) \end{aligned} \tag{B.5}$$

There are three canonical problems associated with HMM:

- Given the parameters of the model, compute the probability of a particular output sequence. This requires summation over all possible state sequences, but can be done efficiently using the forward algorithm, which is a form of dynamic programming.

- Given the parameters of the model and a particular output sequence, find the state sequence that is most likely to have generated that output sequence. This requires finding a maximum over all possible state sequences, but can similarly be solved efficiently by the Viterbi algorithm.

- Given an output sequence or a set of such sequences, find the most likely set of state transition and output probabilities. In other words, derive the maximum likelihood estimate of the parameters of the HMM given a dataset of output sequences. No tractable algorithm is known for solving this problem exactly, but a local maximum likelihood can be derived efficiently using the Baum-Welch algorithm or the Baldi-Chauvin algorithm. The Baum-Welch algorithm is an example of a forward-backward algorithm, and is a special case of the Expectation-maximization algorithm.