Object-level Vertical Search

Zaiqing Nie Ji-Rong Wen Wei-Ying Ma
Web Search and Mining Group
Microsoft Research Asia
Beijing, China
{znie, jrwen, wyma} @microsoft.com

ABSTRACT

Current web search engines essentially conduct document-level ranking and retrieval. However, structured information about real-world objects embedded in static webpages and online databases exists in huge amounts. We explore a new paradigm to enable web search at the object level in this paper, extracting and integrating web information for objects relevant to a specific application domain. We then rank these objects in terms of their relevance and popularity in answering user queries. In this paper, we introduce the overview and core technologies of object-level vertical search engines that have been implemented in two working systems: Libra Academic Search (http://libra.msra.cn) and Windows Live Product Search (http://products.live.com).

Keywords

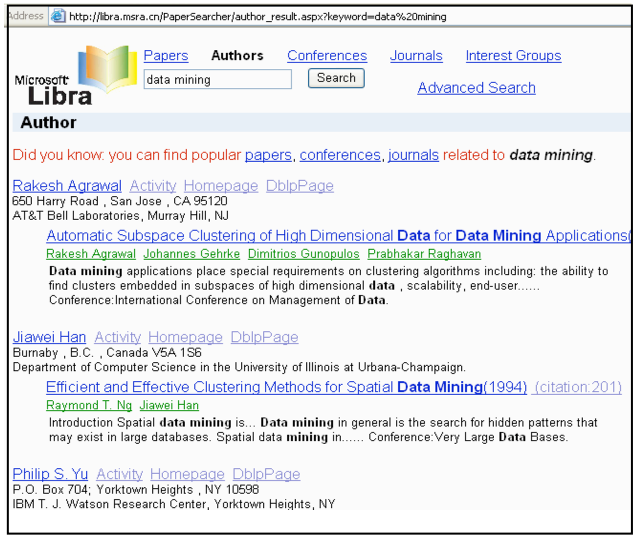
Web Information Extraction, Information Integration, Web Search, Object-level Ranking

1. INTRODUCTION

The primary function of current web search engines is essentially relevance ranking at the webpage level (i.e. Page-level Search), an information retrieval paradigm for more than 25 years. We believe that within a few years these types of general web search technologies will become commodity. In this paper, we are considering a paradigm shift to enable searching at the object level.

A large portion of the web is inherently (semi-)structured, and provides information for various kinds of real-world objects (i.e. entities). Typical objects are products, people, papers, organizations, and the like. We can imagine that if these objects can be extracted from the web and integrated in structured databases, powerful object-level search engines can be built to more precisely meet users' information needs. This is especially true for vertical search domains, such as academic search, product search, people search, and restaurant search. In these vertical domains, people are really interested in information about specific objects, not the pages themselves. For example, researchers always want to find information about conferences, journals and

3rd *Biennial Conference on Innovative Data Systems Research (CIDR)* January 7-10, 2007, Asilomar, California, USA.



http://libra.msra.cn/PaperSearcher/author_result.aspx?keyword=data%20mining

Author

Did you know: you can find popular papers, conferences, journals related to ***data mining***.

Rakesh Agrawal Activity Homepage DblpPage
650 Harry Road , San Jose , CA 95120
AT&T Bell Laboratories, Murray Hill, NJ
Automatic Subspace Clustering of High Dimensional **Data** for **Data Mining** Applications(
Rakesh Agrawal Johannes Gehrke Dimitrios Gunopulos Prabhakar Raghavan
Data mining applications place special requirements on clustering algorithms including: the ability to find clusters embedded in subspaces of high dimensional **data** , scalability, end-user......
Conference:International Conference on Management of **Data**.

Jiawei Han Activity Homepage DblpPage
Burnaby , B.C. , Canada V5A 1S6
Department of Computer Science in the University of Illinois at Urbana-Champaign.
Efficient and Effective Clustering Methods for Spatial **Data Mining**(1994) (citation:201)
Raymond T. Ng Jiawei Han
Introduction Spatial **data mining** is... **Data mining** in general is the search for hidden patterns that may exist in large databases. Spatial data **mining** in...... Conference:Very Large **Data** Bases.

Philip S. Yu Activity Homepage DblpPage
P.O. Box 704; Yorktown Heights , NY 10598
IBM T. J. Watson Research Center, Yorktown Heights, NY

Figure 1. Ranking Relevant Authors, Conferences, and Journals in Libra Academic Search

other researchers. If one wanted to find information about the top world researchers within a particular domain and tried a query in a basic page-level search engine, it could be very difficult to find popular researchers. Our object-level search engines, however, specifically provide lists of researchers, and extract and integrate the desired information (See Figure 1 for an example list of researchers for the query "data mining").

In Table 1, we compare object-level and page-level search. In page-level search, webpages are the basic retrieval units, and the information in a page is treated as a bag of words. Information retrieval technologies are used as core technologies to answer user queries, while object-level search uncovers structured information about real-world objects that are the retrieval units. One obvious advantage of object-level search is its capability of answering complex queries with direct and aggregate answers because of the availability of semantics defined by the object schema. Otherwise, it could take one several hours to sift through hundreds of webpages returned by a page-level search engine.

The challenge here is where and how to obtain high-quality structured data needed by an object-level search engine, and how to rank resulting objects to return the most relevant ones.

To illustrate the power of this new generation of web search, we have built a scientific web search engine called Libra (See Figure 1. http://libra.msra.cn) to evaluate various object-level web search

Table 1. Object-Level Search *vs.* Page-Level Search

	Page-Level Search	Object-Level Search
Technology	Information Retrieval ; Pages as Retrieval Units	Database; Machine Learning; Objects as Retrieval Units
Pros	Ease of Authoring; Ease of Use	Powerful Query Capability; Direct Answer; Aggregate Answer
Cons	Limited Query Capability	Where and How to Get the Objects?

techniques undertaken. We believe it is more advanced than the existing page-level academic search engines. In Libra, each researcher, scientific paper, conference, journal, and interest group is treated as a web object. We extract and integrate the information from multiple sources including ACM Digital Library, DBLP, and CiteSeer, and others. Objects can be retrieved and ranked according to their relevance to the query and their popularity. As a result of information extraction and integration, the system is essentially a web data warehouse with the capability of handling structured queries. We are currently working on applying Libra technologies to building Window Live Product Search (See Figure 2. http://products.live.com).

Until now, the Beta release of Window Live Product Search has incorporated our product page classification and extraction techniques. After the first month running, we have already indexed more than 100,000 sellers, 31,627,416 commercial pages, and 800 million automatically extracted product records. We believe we could make Window Live Product Search the largest product catalog in the world by using object-level vertical search technologies.

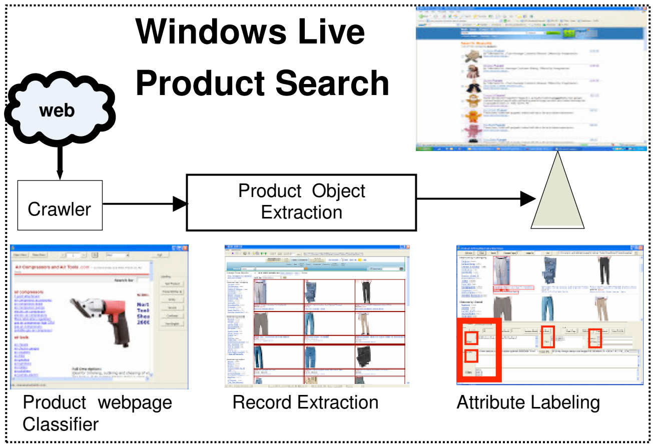


Figure 2. Window Live Product Search with our Product Page Classification and Extraction techniques.

The rest of the paper is organized as follows. In the next section, we outline the requirements of an object-level vertical search engine. Section 3 introduces the system architecture and core techniques. We then discuss subsystem integration. In Section 5, we emphasize the need for a new infrastructure to support object-level vertical search. Section 6 discusses related work. Section 7 presents our conclusion.

2. Requirements

The requirements for a large-scale object-level vertical search engine are as follows:

1. **Reliability**: High quality structured data is necessary to generate direct and aggregate answers. If the underlying data are not reliable, then the users may prefer sifting the webpages to find answers rather than trust the noisy direct answers returned by an object-level vertical search engine;
2. **Completeness**: We want our data to be as complete as possible to provide trustworthy answers;
3. **Ranking Accuracy**: With billions of potential answers to a query, an optimal ranking mechanism is critical for locating relevant object information.
4. **Scalability**: Since our object-level vertical search engines should include all the information within a vertical domain both on the web and in local databases, the object warehouse could be enormous. For example, according to our empirical study of 51,000 randomly crawled webpages, we found that more than ten percent of webpages are product pages. So, if we include both the product information extracted from crawled webpages and product data feeds from e-commerce databases in a search engine, this could entail dealing with billions of data records. We need to incorporate large-scale data processing technologies to make our structured data retrieval scalable.

In the following sections, we will introduce the system architecture and infrastructure design with these requirements in mind.

3. SYSTEM ARCHITECTURE & CORE TECHNIQUES

In this section, we discuss the system architecture and core techniques of object-level vertical search engines.

Figure 3 shows the brief architecture of an object-level vertical search engine. First, a crawler fetches web data related to the targeted objects within a specific vertical domain, and the crawled data is classified into different categories, such as papers, authors, products, and locations. For each category, a specific entity extractor is built to extract objects from the web data. At the same time, information about the same object is aggregated from multiple different data sources. Once objects are extracted and aggregated, they are put into the object warehouses, and vertical search engines can be constructed based on the object warehouses. Moreover, advanced object-level ranking and mining techniques can be applied to make search more accurate and intelligent.

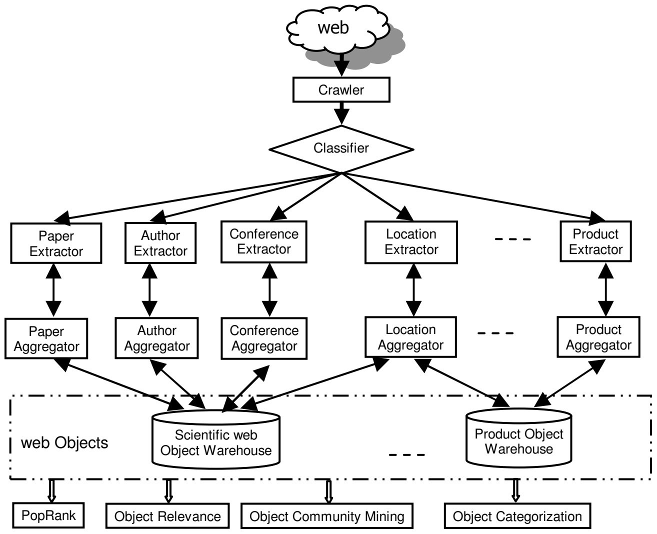
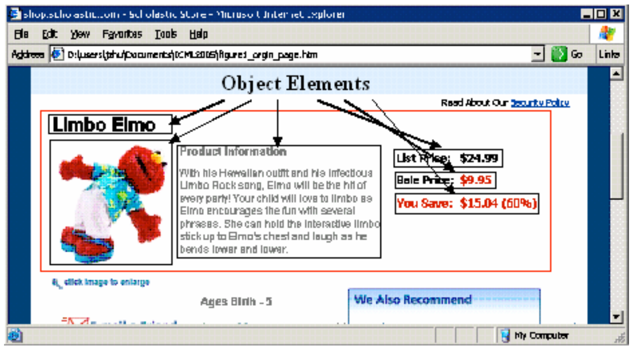


Figure 3. System Architecture


shop.scholastic.com - Scholastic Store - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address D:\users\jizhu\Documents\ICML2005\figure1_orgin_page.htm
Go
Links
Object Elements
Read About Our Security Policy
click image to enlarge
We Also Recommend
My Computer

Limbo Elmo

Product Information

With his Hawaiian outfit and his infectious Limbo Rock song, Elmo will be the hit of every party! Your child will love to limbo as Elmo encourages the fun with several phrases. She can hold the interactive limbo stick up to Elmo's chest and laugh as he bends lower and lower.

List Price: $24.99

Sale Price: $9.95

You Save: $15.04 (60%)

Ages Birth - 5

Figure 4. An Object Block with 6 Elements (contained in the red rectangle) in a Webpage.

3.1 Crawler and Classifier

The tasks of the crawler and classifier are to automatically collect all relevant webpages/documents that contain object information for a specific vertical domain. The crawled webpages/documents will be passed to the corresponding object extractor for extracting the structured object information and building the object warehouse.

3.1.1 Insight

If we use the nodes to denote the objects and edges to denote the relationship links between the objects, we can see that the objects information with a vertical domain forms an object relationship graph. For example, in Libra academic search, we have three different types of nodes to representing papers, authors, and conferences/journals, and three different types of edges (i.e. links) pointing to paper objects that represent three varying types of relationships. They are cited-by, authored-by, and published-by. The ultimate goal of the crawler is to effectively and efficiently collect relevant webpages and to build a complete object relationship graph with as many nodes and edges and as many attribute values for each node as possible (assuming we have perfect extractors and aggregators).

3.1.2 Our Approach

We build a "focused" crawler that uses the page classifier and the existing partial object relationship graph to guide the crawling process. Basically, in addition to the web graph which is used by most page-level crawlers, we employ an object relationship graph to guide our crawling algorithm.

Since the classifier is coupled with the crawler, it needs to be very fast to ensure efficient crawling. Based on our experience in building a classifier for Libra and Windows Live Product Search, we found that we could always use some strong heuristics to quickly prune most of irrelevant pages. For example, in our product pages classifier, we can use the price identifiers (such as dollar signs $) to efficiently prune most non-product pages. The average time of our product classifier is around 0.1 millisecond, and its precision is around 0.8, with recall around 0.9.

3.2 Object Extractor

Information (e.g. attributes) about a web object is usually distributed in many web sources and within small segments of webpages. The task of an object extractor is to extract metadata about a given type of objects from every web page containing this type of objects. For example, for each crawled product web page, we extract *name*, *image*, *price* and *description* of each product. If all of these product pages or just half of them are correctly extracted, we will have a huge collection of metadata about real-world products that could be used for further knowledge discovery and query answering. Our statistical study on 51,000 randomly crawled webpages shows that about 12.6 percent are product pages. That is, there are about 1 billion product pages within a search index containing 9 billion crawled webpages.

However, how to extract product information from webpages generated by many (maybe tens of thousands of) different templates is non-trivial. One possible solution is that we first distinguish webpages generated by different templates, and then build an extractor for each template. We say that this type of solution is *template-dependent*. However, accurately identifying webpages for each template is not a trivial task because even webpages from the same website may be generated by dozens of templates. Even if we can distinguish webpages, template-dependent methods are still impractical because learning and maintenance of so many different extractors for different templates will require substantial efforts.

3.2.1 Insight

By empirically studying webpages across websites about the same type of objects across web sites, we find many template-independent features.

- Information about an object in a web page is generally grouped together as an *object block*, as shown in Figure 4. Using existing web page segmentation [7] and data record extraction technologies [35], we can automatically detect these object blocks, which we further segment into atomic extraction entities called *object elements*. Each object element provides partial information about a single attribute of the web object.

Table 2. Statistical Results for Objects from Both Product Pages and Homepages. (We have used "DESC" instead of "DESC-RIPTION" and "TEL" instead of "TELEPHONE" for space.)

PRODUCT	BEFORE	HOMEPAGE	BEFORE
(NAME, DESC)	1.000	(NAME, TEL)	1.000
(NAME, PRICE)	0.987	(NAME, EMAIL)	1.000
(IMAGE, NAME)	0.941	(NAME, ADDRESS)	1.000
(IMAGE, PRICE)	0.964	(ADDRESS, EMAIL)	0.847
(IMAGE, DESC)	0.977	(ADDRESS, TEL)	0.906

- Our empirical study shows that strong sequencing characteristics exist for web objects of the same type across different web sites. To demonstrate that strong sequencing characteristics exist, we conduct our statistical study on two types of web objects, products and researchers' homepages. Specifically, we randomly collect 100 product pages (which contain 964 product blocks) and 120 homepages from different web sites. For product objects, the attributes "name", "image", "price" and "description" are surveyed. For researchers' homepages, the attributes "name", "telephone", "email" and "address" are considered. We decide the sequence order of the elements in a web page in a top-down and left-right manner based on their position information. Basically, the element in the top level will be ahead of all the elements below it and for the elements at the same level, the left elements will be ahead of their right elements. As we can see from Table 2, strong sequence characteristics exist for among most attribute pairs in both object types. For example, a product's name is always ahead of its description in all the pages.

3.2.2 Template-Independent Web Object Extraction

We propose *template-independent* metadata extraction techniques for the same type of objects. Specifically in [43][61][62], we extended the linear-chain Conditional Random Fields (CRFs) [31] which are the state of the art approaches in information extraction taking advantage of the sequencing characteristics to do better labeling.

3.2.2.1 2D Conditional Random Fields

In order to use the existing linear-chain CRFs for Web object extraction, we have to first convert a two-dimensional object block (*i.e.* an object block whose elements are two-dimensionally laid out) into a sequence of object elements. Given the two-dimensional nature of object blocks, how to sequentialize them in a meaningful way could be very challenging. Moreover, as shown by our empirical evaluation, using the two-dimensional neighborhood dependencies (*i.e.* interactions between labels of an element and its neighbors in both vertical and horizontal directions) in Web object extraction could significantly improve the extraction accuracy.

To better incorporate the two-dimensional neighborhood dependencies, a two-dimensional Conditional Random Field (*2D CRF*) model is proposed in [61]. We present the graphical representation of the 2D CRF model as a 2D grid (See Figure 5)

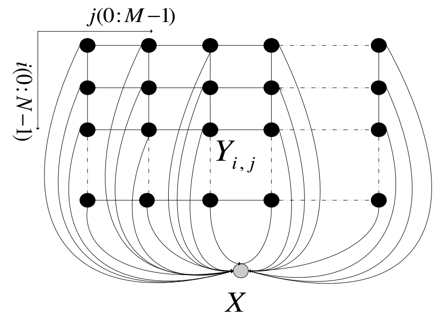


Figure 5. The Graphical Structure of 2D CRFs

and reformulate the conditional distribution by defining some matrix random variables. Then we deduce the forward-backward vectors based on the reformulated conditional distribution for efficient parameter estimation and labeling. Since the sizes of the elements in an object block can be arbitrary, we introduce the concept of *virtual states* to model an object block as a 2D grid. We compare our model with linear-chain CRF models for product information extraction and the experimental results show that our model significantly outperforms linear-chain CRF models in scenarios with two-dimensional neighborhood dependencies.

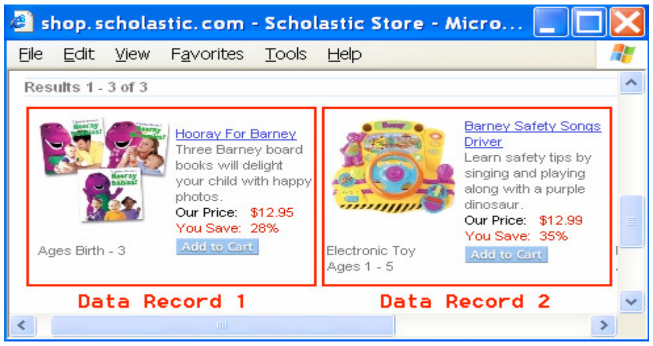
3.2.2.2 Hierarchical Conditional Random Fields

In traditional information extraction work, there is no clear definition of object extraction, and hence no discussion on how to link data record (i.e. object block) detection work with object attribute labeling. Although we can build a template-independent object extractor by first using the techniques in [7] or [35] to detect the data records and then use the techniques in [61] to label the data elements within the detected records. However, it is *highly ineffective to use decoupled strategies* – attempting to do data record detection and attribute labeling in two separate phases. This is because:

Error Propagation: Since record detection and attribute labeling are two separate phases, the errors in record detection will be propagated to attribute labeling. Thus, the overall performance is limited and up-bounded by that of record detection. Suppose the record detection and attribute labeling have precisions 0.8 and 0.9 respectively, then the overall precision will be no more than 0.8. And, if they also perform independently, the precision will be 0.72.

Lack of Semantics in Record Detection: Human readers always take into account the semantics of the text to understand Webpages. For instance, in Figure 4, when claiming a block is a data record, we use the evidence that it contains a product's name, image, price and description. Thus, a more effective record detection algorithm should take into account the semantics of the text, but existing methods [7][35][34] do not consider that.

Lack of Mutual Interactions in Attribute Labeling: The data records in the same page are related. They always share a common template and the elements at the same position of different records always have similar features and semantics. For example, in Figure 6(a) the button "Add to Cart" appears in the same position in all three data records and the element on the left-top of each record is an image. So, if we label the elements of the records

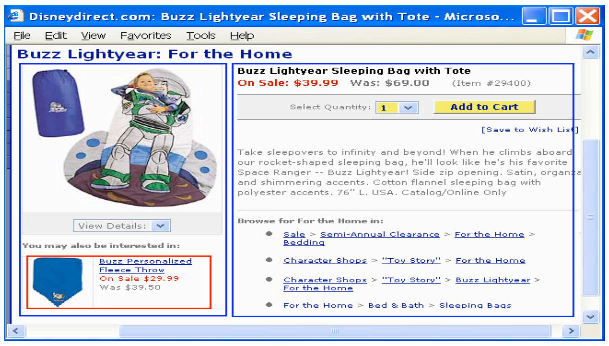

shop.scholastic.com - Scholastic Store - Micro...
File Edit View Favorites Tools Help
Data Record 1
Data Record 2

Results 1 - 3 of 3

Hooray For Barney
Three Barney board books will delight your child with happy photos.
Our Price: $12.95
You Save: 28%
Add to Cart
Ages Birth - 3

Barney Safety Songs Driver
Learn safety tips by singing and playing along with a purple dinosaur.
Our Price: $12.99
You Save: 35%
Add to Cart
Electronic Toy
Ages 1 - 5

Figure 6(a). A Sample Webpage with Two Similar Data Records Which are Contained in Red Boxes.


Disneydirect.com: Buzz Lightyear Sleeping Bag with Tote - Microso...
File Edit View Favorites Tools Help

Buzz Lightyear: For the Home

Buzz Lightyear Sleeping Bag with Tote
On Sale: $39.99 Was: $69.00 (Item #29400)
Select Quantity: 1 Add to Cart
[Save to Wish List]

Take sleepovers to infinity and beyond! When he climbs aboard our rocket-shaped sleeping bag, he'll look like he's his favorite Space Ranger -- Buzz Lightyear! Side zip opening. Satin, organza and shimmering accents. Cotton flannel sleeping bag with polyester accents. 76'' L. USA. Catalog/Online Only

View Details:

You may also be interested in:
Buzz Personalized Fleece Throw
On Sale $29.99
Was $39.50

Browse for For the Home in:
- Sale > Semi-Annual Clearance > For the Home > Bedding
- Character Shops > "Toy Story" > For the Home
- Character Shops > "Toy Story" > Buzz Lightyear > For the Home
- For the Home > Bed & Bath > Sleeping Bags

Figure 6(b). A Detail Page Containing Detailed Information about a Product.

within the same page together in a collective manner, it is easier for us to detect that the repeated elements "Add to Cart" are less informative and more likely to be noise. However, our 2D CRF approach [61] fails to consider that because the data records are independently labeled.

First-Order Markov Assumption: For Webpages and especially for detail pages (i.e. Webpages only containing detailed information about a single object), long distance dependencies always exist between different attribute elements. This is because there are always many irrelevant elements (i.e. noise) appearing between the attributes of an object. For example, in Figure 6(b) there is substantial noise, such as "Add to Cart" and "Select Quantity" between the price and description. However, 2D CRFs [61] cannot incorporate these long distance dependencies because of its first-order Markov assumption (i.e. only the dependencies of neighboring nodes are considered and represented.

In [62], we introduce a novel graphical model called Hierarchical Conditional Random Field (HCRF) model to jointly optimize record detection and attribute labeling.

By using the vision-based page segmentation (VIPS) technology [7], which makes use of page layout features such as font, color, and size to construct a *vision-tree* for a Web page, we can get a better representation of a page compared with commonly used tag-tree. Given a vision-tree, record detection can be considered as the task of locating the minimum set of elements that contain the content of a record. In this way, both record detection and attribute labeling become the task of assigning labels to the nodes on a vision-tree, so they can be integrated in one probabilistic model as in this paper. In contrast to existing decoupled strategies that perform record detection and attribute labeling as two separate phases, our approach leverages the labeling results of attribute labeling for record detection, and at the same time benefits from the incorporation of some global features based on tree-alignment for attribute labeling. As a conditional model [31], HCRF can efficiently incorporate any useful features for Web data extraction. By incorporating hierarchical interactions, HCRF could incorporate long distance dependencies and achieve promising results on detail Webpages.

Our Empirical studies show that mutual enhancement of record detection and attribute labeling could be achieved in our joint approach, and HCRFs could perform very promisingly on detail webpages.

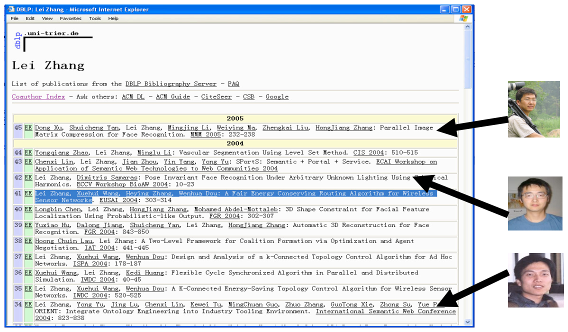
3.3 Object Aggregator

Each extracted web object need to be mapped to a real world object and stored into a web data warehouse. To do so, the object aggregator needs to integrate information about the same object and disambiguate different objects. This is a typical information/data integration problem and usually includes two subproblems:

- An object has multiple inconsistent attribute values, arising from the inconsistent formats, spelling mistakes, and so on. For instance, both "WWW" and "World Wide web Conference" could refer to the same conference;
- An object type has few other attributes other than its name, and targets to distinguish among two or more objects which share the same name. For example, when you search for the publications of "Lei Zhang", a very common Chinese name, in DBLP, different author objects are mixed in the same webpage (see Figure 7).

Although both of the subproblems need to be dealt with in practical applications, the latter one relies more on additional information in addition to the attribute values, and is challenging especially when there are few attributes. The former subproblem has attracted extensive attention, while relatively little work has been done in the latter one. Unfortunately, the latter is a common problem in our object-level vertical search scenarios. For example, in our Libra academic search scenario, we seem to have little information about author objects except their names, which is reliable indicator for use in disambiguating different real-world authors for two different papers. In our Product search scenario we face the same problem of product name disambiguation problem.

Some recent work [11] proposes the exploitation of connection via object relationships in the object-relationship graph, in addition to the available object attribute values, for name disambiguation. The assumption behind their approaches is that, if two identical names in different contexts refer to the same object, they are more likely to be strongly connected on the entity-relationship graph. For example, if two "Lei Zhangs" refer to the same person, it is very likely that they share some coauthors, references, or are indirectly related by a chain of relationships. Based on this assumption, two identical names are detected as referring to the same object only when the connection strength between them is stronger than a predefined threshold.


DBLP: Lei Zhang - Microsoft Internet Explorer
File Edit View Favorites Tools Help
dblp.uni-trier.de

Lei Zhang

List of publications from the DBLP Bibliography Server - FAQ

Coauthor Index - Ask others: ACM DL - ACM Guide - CiteSeer - CSB - Google

2005

45 EE Dong Xu, Shuicheng Yan, Lei Zhang, Mingjing Li, Weiying Ma, Zhengkai Liu, HongJiang Zhang: Parallel Image Matrix Compression for Face Recognition. MMM 2005: 232-238

2004

44 EE Yongqiang Zhao, Lei Zhang, Minglu Li: Vascular Segmentation Using Level Set Method. CIS 2004: 510-515

43 EE Chenxi Lin, Lei Zhang, Jian Zhou, Yin Yang, Yong Yu: SPortS: Semantic + Portal + Service. ECAI Workshop on Application of Semantic Web Technologies to Web Communities 2004

42 EE Lei Zhang, Dimitris Samaras: Pose Invariant Face Recognition Under Arbitrary Unknown Lighting Using [illegible]ical Harmonics. ECCV Workshop BioAW 2004: 10-23

41 EE Lei Zhang, Xuehui Wang, Heying Zhang, Wenhua Dou: A Fair Energy Conserving Routing Algorithm for Wireless Sensor Networks. EUSAI 2004: 303-314

40 EE Longbin Chen, Lei Zhang, HongJiang Zhang, Mohamed Abdel-Mottaleb: 3D Shape Constraint for Facial Feature Localization Using Probabilistic-like Output. FGR 2004: 302-307

39 EE Yuxiao Hu, Dalong Jiang, Shuicheng Yan, Lei Zhang, HongJiang Zhang: Automatic 3D Reconstruction for Face Recognition. FGR 2004: 843-850

38 EE Hoong Chuin Lau, Lei Zhang: A Two-Level Framework for Coalition Formation via Optimization and Agent Negotiation. IAT 2004: 441-445

37 EE Lei Zhang, Xuehui Wang, Wenhua Dou: Design and Analysis of a k-Connected Topology Control Algorithm for Ad Hoc Networks. ISPA 2004: 178-187

36 EE Xuehui Wang, Lei Zhang, Kedi Huang: Flexible Cycle Synchronized Algorithm in Parallel and Distributed Simulation. IWDC 2004: 40-45

35 EE Lei Zhang, Xuehui Wang, Wenhua Dou: A K-Connected Energy-Saving Topology Control Algorithm for Wireless Sensor Networks. IWDC 2004: 520-525

34 EE Lei Zhang, Yong Yu, Jing Lu, Chenxi Lin, Kewei Tu, MingChuan Guo, Zhuo Zhang, GuoTong Xie, Zhong Su, Yue P[illegible] ORIENT: Integrate Ontology Engineering into Industry Tooling Environment. International Semantic Web Conference 2004: 823-838

Figure 7. Three Persons Found in the DBLP Page for "Lei Zhang".

3.3.1 *Insight*

In real applications, many relationships may be missing in local datasets. For example, some citation links between papers are missing in Libra because of the limitation of reference extraction techniques. In such cases, for two author name appearances that refer to the same author, the connection strength may not be strong enough and the existing connection-based approaches will fail to detect them as a match. Therefore, an approach that leverages more information besides the connection evidence in the local dataset (i.e. local connection) is desired.

Based on our experience with author name disambiguation in Libra academic search and product name disambiguation in Window Live Product Search, we found that, for different object appearances with identical names referring to the same object, their contexts should be strongly connected with each other on the web. We call such connections *web Connections*. For example, in the Libra scenarios, we find that authors may list their papers on their homepages. So if two papers are listed together on the same web page, their authors with the same name are more likely to be the same real-world author. In product search scenarios, we find that different offers from multiple ecommerce websites about the same product are generally listed together by various shopping aggregator websites for comparison.

3.3.2 *Object Identification using Web Connections*

We measure the web connection between two object appearances based on the co-occurrences of their contexts in a website (or webpage). The co-occurrence information could be easily obtained by sending the context information as queries to a search engine (e.g., Google, MSN Search). However it is non-trivial to measure the web connection strength based on the co-occurrence information, since co-occurrences in different websites usually indicate quite different connection strengths. For example, to determine whether two identical author names of two papers refer to the same person, the co-occurrences of their papers in the website of a small research lab usually provides stronger connection than the co-occurrences in DBLP. This is because the former co-occurrence most likely indicates that the papers are related to the small lab, while the latter one only shows that these papers are in the computer science domain. Moreover, even different co-occurrences in the same website need to be discriminated (e.g., co-occurrences in the same webpage usually indicate stronger connection than those in different webpages.). To handle these issues, our measure of web connection not only discriminates the relative importance of different websites, but also considers the URL distance between webpages inside a website.

Practical Considerations: There are still some issues in computing the Web connection. First, when only a search engine is given, all the Web sites are "hidden" behind the search interface, and cannot be obtained beforehand. Second, even if all the Web sites are known beforehand (e.g., we have already indexed the Web corpora locally), there are usually billions of Web sites in Web corpora, and to train a connection function handling such a high dimension is infeasible. We introduce a practical solution to compute web connections based on our observations on the Web corpora according to a study in Libra:

First of all, the coverage distribution of a given type of objects among different websites usually obeys a power-law distribution. By the study on the distribution of around 1 million papers of Libra in the Web corpora (using the paper titles to get the Web occurrences of paper objects), as illustrated in Figure 8, only a few sites are the ones with high coverage (i.e. large number of academic papers); while the "massive many" are the sites which rarely contain papers. Most of the websites with relatively higher coverage are the ones which provide paper search service or the websites of prestigious research organizations.

Second, although co-occurrences of two context-objects in different sites are of different importance for name disambiguation and are not necessarily proportional to the inverse coverage score, we find that co-occurrences of context-objects in sites with extremely low coverage score (i.e., the sites whose coverage scores are below a certain threshold) always provide strong enough evidence for a correct match. For example, we found that the following two papers "Emergent Semantics from Folksonomies: A Quantitative Study" and "Integrating Web Services into Ontology-based Web Portal" could both be found in http://apex.sjtu.edu.cn/, a site of a small research group focusing on knowledge management, and the authors "Lei Zhang" of those two papers are actually the same person. For the purpose of convenience, we call these sites as small hubs, while the ones with high coverage score are called as big hubs.

Based on the prior observations, we can make the following assumption for name disambiguation:

For two name appearances A_1 and A_2 with the same name, if their context-objects are found to co-occur in one of the small hubs (whose coverage scores are below a certain threshold), A_1 and A_2 most likely refer to the same object.

The assumption above is more likely to be valid when a conservative threshold is set.

Third, it is feasible to discover the big hubs for a certain type of objects with a few times of probing. Figure 9 shows the growth of the number of newly discovered big hubs (i.e. Websites with coverage scores greater than 1%) when we use paper titles to probe the Web corpora. We observe that the newly discovered big hubs tends to converge after only around 600 probes of papers, while there are around 1 million papers in our dataset. Note that the sample papers for probing are randomly selected, and there is no bias in it.

Based on the above three observations, our name disambiguation approach considers two object appearances with the same name as

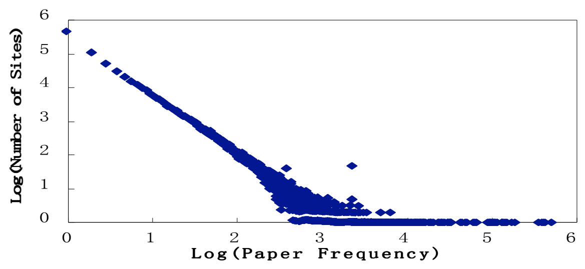


Figure 8. Power Law Distribution of Paper Frequencies of Websites.

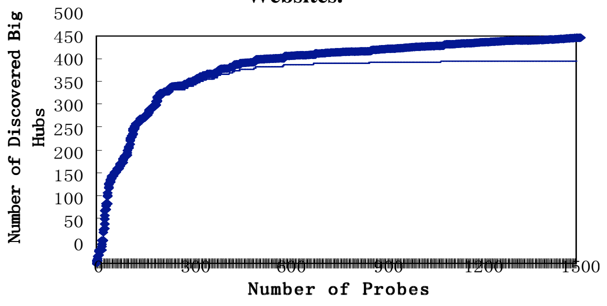


Figure 9. The Growth of Big Hub (>1%) Numbers by Randomly Probing

the same object once their context-object information is found in a small hub. For the appearance pairs where there is no co-occurrence in any small hub, we need to compute the Web connection strength for all their co-occurrences in big hubs. As shown in the above observation, the number of big hubs is relatively limited, so it becomes feasible to train an adaptive connection function which gives suitable weights to the co-occurrences in these big hubs.

As we mentioned before, we use a coverage threshold to determine which websites/webpages are small hubs. We use the labeled dataset (i.e. training data) to empirically select a good coverage threshold. Specifically, we try several different coverage thresholds, and for each threshold, we simply use the co-occurrences in small hubs for name disambiguation in the labeled dataset without considering the co-occurrences in the big hubs. For each tried coverage threshold setting, we observe the precision result. Generally, the higher the threshold is, the lower the precision will be. We usually set a relatively conservative threshold to ensure the precision.

3.4 Object-level Ranking and Analysis

After information extraction and integration we construct the relationship graph between web objects. By performing link analysis on this object relationship graph, we can compute the importance of a web object or discover other interesting knowledge or patterns that are impossible to obtain in the traditional web graph.

On the traditional web graph, different pages have different popularity according to their in-links. Technologies such as PageRank [46] and HITS [29] have been successfully applied to distinguish the popularity of different webpages through analyzing the link structure in the web graph. It is obvious that, in

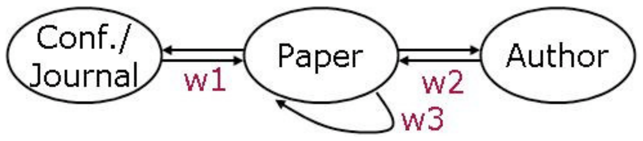


Figure 10. Paper Object Relationship Graph

the object graph, objects are also not equally popular. Take the research domain as an example. Only several top conferences within a research field can attract high quality papers, and their papers are more likely to be read. To help users quickly locate their interested objects, we should calculate the popularity of collected objects. Because it is clear that the more popular the objects are, the more likely they will be interested by a user. So a natural question is: could the popularity of web objects be effectively computed by also applying link analysis techniques? According to our experience in building Libra academic search, the answer to the question is yes, but quite different technologies are required because of the unique characteristics of object graph.

For link analysis, the most unique characteristics of the object graph is the heterogeneity of links, i.e., objects are related to each other through different types of relationships. For example, a paper object may be cited by a set of other paper objects, written by a set of author objects, and published in a conference/journal object (See Figure 10). So there are three kinds of different links in the graph: cited-by, authored-by and published-by and they have quite different semantics. The traditional link analysis methods including PageRank and HITS assume that all the links are with the same "endorsement" semantics and equally important, directly applying these methods would result in unreasonable popularity ranking. For example, the popularity of a paper should not be affected too much by the number of authors, and the number of citations does have a large impact on it. We add a *popularity propagation factor (PPF)* to each link of the object relationship graph pointing to an object, and uses different propagation factors for links of different types of relationships [42]. For example, for the links pointing to a paper object, we need three propagation factors for the three different types of relationships: cited-by, authored-by, and published-by, respectively. However manually assigning these factors to make the popularity ranking reasonable is extremely challenging. With a huge link graph, it is difficult for us to tell which types of links are more important and even harder to quantify their exact importance.

3.4.1 Insight

Base on our experience in building Libra academic search, it is always easier for us to collect some partial ranking of the objects from domain experts. For example, as researchers, we know the order of the top conferences or journals within our research field, and we may also know which papers are more popular.

3.4.2 PopRank Model

We propose *PopRank*, a method to measure the popularity of web objects in an object graph. It extends the PageRank model by using different propagation factors for links of different types of relationships. We propose a learning based approach to automatically learn the popularity propagation factors for different types of links using the partial ranking of the objects given by domain experts. The simulated annealing algorithm is used to explore the search space of all possible combinations of propagation factors and to iteratively reduce the difference between the partial ranking from the domain experts and that from our learned model.

One major challenging problem facing our learning approach is that it is prohibitively expensive to try hundreds of combinations of feasible factors which are normally needed for us to get a reasonable assignment of the propagation factors. It may take hours to compute the *PopRank* of the objects to test the optimality of a PPF factor assignment. In order to make the learning time manageable, we propose the use of a subgraph of the entire object link graph in the learning process. As soon as we have most of the related objects and their links surrounding the training objects, we should be able to calculate a close approximation of the *PopRank* of these training objects. Since we are not interested in finding the exact rank scores but the relative rank of these training objects, little reduction of accuracy will not affect the optimality of the assignment too much. However, in cases where the object link graph is prohibitively large, one might have to trade optimality for efficiency. The *PopRank* link analysis model has been evaluated in the context of Libra academic search. Our experimental results on *Libra* show that *PopRank* can achieve significantly better ranking results than naively applying PageRank on the object graph.

4. Putting Together: Integrating Subsystems

In this section, we mainly discuss how we integrate all the subsystems together to achieve overall high performance. Since data quality is the most important factor to evaluate the overall performance of object-level vertical search engine, we will focus on how to improve the overall data quality, mainly on improving the extraction accuracy.

4.1 Validation across Multiple Sources and Subsystems

Because of the arbitrary nature of the web data, it's challenging to achieve perfect extraction, especially when we want to only train a single extractor for all webpages containing the same type of objects. Fortunately, the information on the web is always redundant, and could be used for validation across these redundant sources. For example, the paper meta-data could be extracted from author homepages and from PDF files. From PDF files, we could extract the paper metadata both from their header and from their references. At the same time these metadata are also available from web databases such as ACM Digital Library, IEEE Digital Library.

If we could find the same metadata twice we could be ensured that our extraction is correct. Of course, here we have to use the object integration subsystem to match difference occurrences about the same paper to enable the validation function.

4.2 Object Retrieval Model Insensitive to Extraction Error

In traditional IR models, documents are taken as the retrieval units and the content of documents is considered reliable. However, the reliability assumption is no longer valid in the object retrieval context. There are several possible routes to introduce errors in object contents during the process of object extraction:

Source-level error: Since the quality of web sources can vary significantly, some information about an object in some sources may be simply wrong.

Record-level error: Due to the huge number of web sources, automatic approaches are commonly used to locate and extract the data records from webpages or web databases. It is inevitable that the record extraction (i.e. detection) process will introduce additional errors. The extracted records may miss some key information or include some irrelevant information, or both.

Attribute-level error: Even if the web source is reliable and the object contents are correctly detected, the description of an object (i.e. object element labeling) may be still wrong because of incorrect attribute value extraction. For example, it is very common to label a product name by brand, or vice versa. In Citeseer, we usually find that author names are concatenated to paper titles, or some names are missing.

We focus on this unreliability problem in web object retrieval. Our basic ideas are based on two principles. First, as described above, errors can be introduced in both the record level and attribute level. Moreover, as errors will be propagated along the extraction process in decoupled object extraction techniques, the accuracy of attribute extraction is surely lower than that of record extraction. However, separating record contents into multiple attributes will bring more information than just treating all contents in a record as a unit. Therefore, it is desirable to combine both record-level representation and attribute-level representation. We hope that by combing representations of multiple levels our method is insensitive to extraction error. Second, multiple copies of information about the same object usually exist. These copies may be inconsistent because of diverse web site qualities and the limited performance of current information extraction techniques. If we simply combine the noisy and inaccurate object information extracted from different sources, we will not be able to achieve satisfactory ranking results. Therefore, we need to distinguish the quality of the records and attributes from different sources and trust data of high reliability more and data of low reliability less. We hope that even when data from some sites have low reliability, we can still get good retrieval performance if some copies of the objects have higher reliability. In other words, our method should also take advantage of multiple copies of one object to achieve stable performance despite varying qualities of the copies.

Based on the above arguments, our goal is to design retrieval models insensitive to data errors and that can achieve stable performance for data with varying extraction accuracies. Specifically, we propose several language models for Web object retrieval, namely a record-level representation model, an attribute-level representation model, and a model balancing record-level and attribute-level representations. We test these models on our Libra Academic Search and compare their performance. We conclude that the best model is the one combining both record-level and attribute-level evidence and taking into account of the errors at different levels.

5. Infrastructure

Once we crawl, extract, and integrate objects from the web, we need an effective infrastructure to store, index, query, and analyze them. In our object-level vertical search scenarios, a good infrastructure should provide support to the following requirements:

1. **It can accommodate large amount of (semi-)structured data with flexible schemas. It should also support sparse columns and on-line schema changes.** To provide a

uniform infrastructure to support different vertical domains, a flexible scheme definition mechanism is needed. In addition, we usually need to change the schemas in real applications. For example, when we first built the product search engine, the initial schema contains attributes like name, price, description, and image. Then, for objects about digital cameras, we want to add more attributes, such as pixel, memory, resolution, etc., and for some other objects about cars, we want to add attributes like color, engine, seats, safety, etc. So, the schemas may change frequently and at the same time some columns are unavoidably sparse since they are only meaningful for subsets of data.

2. **It needs to support rich queries that exploit both the structure and content information. In other words, both (limited) SQL-style structured query and IR-style keyword query should be supported and combined**. In search scenarios, all of the data will be directly retrieved by end users. So, query syntax needs to be user-friendly and simple. Keyword query will be commonly used to search across the objects. On the other hand, advanced users would like to leverage structure information to get more accurate and focused answers. Therefore, structured query capability, such as range query and aggregation, is also necessary.

3. **The infrastructure should run on clusters of commodity machines. Distributed data processing capabilities like data partition, load balancing, parallel execution, replication, fail-over, and recovery should be supported.** The huge data volume and huge query volume make distributed system the best choice for building a search engine. Google File System [22] demonstrates that a robust and efficient storage system built on thousands of commodity machines is the key to the success of a modern search engine. For object-level vertical search, such a kind of large-scale distributed system is also crucial.

We argue that both the existing search engine infrastructures and relational database systems cannot fully meet the above requirements. We are working on building a new infrastructure from scratch to manage large-scale web objects. This infrastructure can be either viewed as a light and scalable distributed database system, or a search engine supporting structured data processing. Moreover, we are also exploiting some new technologies (e.g. C-Store [51]) to see if they are compatible with our requirements.

Currently, there is a strong tendency to combine or merge database system and information retrieval system. The infrastructure we are building can be taken as one case of the DB+IR movement, in a more concrete application scenario.

6. Related Work

We classify the related work into the following categories align with the core techniques in object-level vertical search: Web object extraction, name disambiguation, object-level link analysis, and object relevance ranking.

6.1 Web Object Extraction

Wrapper learning approaches like [40][30] are template-dependent. They take in some manually labeled Webpages and learn some extraction rules (i.e. wrappers). Since the learned wrappers can only be used to extract data from similar pages, maintaining the wrappers as Web sites change will require great efforts. Furthermore, in wrapper learning, a user must provide explicit information about each template. Even if the wrapper learning is efficient, a system that extracts data from many Web sites, as in our application, will be expensive to train. [60][18] [6][10] are also template-dependent, but they do not need manually labeled training samples. They automatically produce wrappers from a collection of similar Webpages. [1][14] take a collection of pages which are assumed to be generated by a common template to deduce the unknown template. Then, the deduced template is used to extract data from similar Webpages.

[59][34] are two template-independent methods. [34] segments data on list pages using the information contained in their detail pages. The need of detail pages is a limitation because automatically identifying links that point to detail pages is non-trivial and there are also many pages that do not have detail pages behind them. [59] mines data records by string matching and also incorporates some visual features to achieve better performance. However, [59] detects data records only using tree regularities and not consider semantics. Furthermore, the data extracted by [34][59] have no semantic labels.

[21] treats Web information extraction as a classification problem. It uses support vector machine to identify the start and end tags for a single attribute. For the task of extracting multiple attributes, this single-attribute extraction method loses the dependencies between different attributes.

The idea of exploring the mutual benefits by integrating two tasks has been attempted in previous work. [41] attempts a mutually beneficial integration of information extraction and data mining. Information extraction makes possible the text mining which needs to handle unstructured text documents, and data mining could provide additional clues to improve the recall of an IE system. [53] proposes an integrated model to do information extraction and coreference. Incorporating extraction uncertainty could help coreference, and leveraging the identified coreference could improve extraction accuracy. However, [53] is not a fully closed integration. As the model could be very complex, separate learning and inference for different substructures is employed in [53]. [52] proposes a factorial CRF to jointly solve two NLP tasks (noun phrase chunking and part of speech tagging) on the same observation sequence. The difference is that our data are hierarchical trees and the data in [52] are sequences.

[49] uses Hierarchical Hidden Markov models (HHMMs) [20] to extract relation instances from biomedical articles. Our work differs from this in two key aspects. First, HHMMs are generative models and HCRFs are discriminative models. Discriminative models could result in better performance in information extraction for their flexibility to incorporate arbitrary and non-independent features of the observations, but generative models must make some strong independence assumption of the observations to achieve inference tractability. This is the key idea underlying the Conditional Random Fields [31]. Second, the data in [49] are two-level representations of sentences, but our data are arbitrary vision-trees. Other work on hierarchical and multi-scale models can be found in Computer Vision and Image Processing [27][58]. Unlike [27], the proposed HCRF model is not a simple multi-scale model because it has inter-level interactions. Our model is also different from [58] in both the graph representation and inference algorithm.

Other work, such as collective information extraction [5] and Semi-Markov extraction models [12][48], could achieve higher performance in named entity extraction problems on flat text documents. However, it's not easy or impossible to adapt them for the integrated Web data extraction, where the data are hierarchically represented. In contrast, the proposed hierarchical HCRF model is the natural and efficient method for it.

6.2 Name Disambiguation

The traditional work on name disambiguation is usually based on the string similarity of the attribute value, such as [24][28][37]. These approaches couldn't work for scenarios where there are few attribute values available for disambiguation.

Recently, the relationship information among different types of objects in a local dataset has started to be exploited for name disambiguation, such as [3][11][17][36][45]. The limitation of these approaches is that they depend too much on the completeness of relationship information, and probably result in low recall.

Some previous research on name disambiguation exploits some specific additional information according to the characteristic of the application. Among them, [33] works based on a concept hierarchy structure; [15] explores the profile for some constraints to improve matching accuracy; [16] uses personal homepages in the reference reconciliation of a PIM system; [39] uses the predefined Website providing zip code for the unsupervised record linkage of restaurant record. Besides, [26] explores the features specific to the citation dataset, such as similarity of paper title words. In contrast, *WebNaD* exploits the local connection and the Web connection, both of which are general information.

Web appearance disambiguation [3] is another work related to ours. It is different from our work in that [3] uses the local structure information for name disambiguation in Web corpora, while our methods can be seen as using the Web corpora for name disambiguation in local structure data. Another kind of Web connection can be derived from the link structure model in [3], and we believe the framework of using web connections can result in higher recall if we could leverage these kinds of evidence. However, for name disambiguation in a local dataset, especially when only a general search engine is given, a more practical Web connection such as Website co-occurrence should be used as an alternative.

6.3 Object-level Link Analysis

Brin and Page first introduce the PageRank technique [46] to calculate the importance of a Web page based on the scores of the pages pointing to the page. Hence, Webpages pointed by many high quality pages become important as well. Alternatively, the importance score of a Web page is equal to the probability that a random surfer, starting from a random page, will reach the Web page at a specific time. Since the PageRank model considers that all the links have the same authority propagation factors, it could not be directly applied to our object-level ranking problem.

The PageRank model has also been adapted to structure databases. Guo et al. [25] introduce XRANK to rank XML elements using the link structure of the database. Balmin et al. propose the ObjectRank system [2] which applies the random walk model to keyword search in databases modelled as labelled graphs. A similar notion of our popularity propagation factors called authority transfer rates is introduced. In their relevance feedback survey study, the authors find out that using different transfer rates for different edge types is effective. However the papers did not discuss how these authority transfer rates could be assigned.

Xi et al. [56] propose a unified link analysis framework called ``link fusion" to consider both the inter- and intra- type link structure among multi-type inter-related datan objects. The PageRank and HITS algorithms [29] are shown to be special cases of the unified link analysis framework. Although the paper mentioned some similar notion of our popularity propagation factor, however how to assign these factors is considered as the most important future research work in the paper. Furthermore, our *PopRank* model itself is also significantly different from the link fusion framework. In our *PopRank* model we take both the Web popularity of an object and the popularity from the object relationship graph into account. Combining both types of popularity is important, especially for application domains where objects are widely available on Web databases and Webpages. For example, if we want to build a product search engine to rank the product related objects, the Web popularity of these objects could be very useful to calculate the popularity of these products, only using the object relationship graph could lead to unreasonable ranking.

6.4 Object Relevance Ranking

In recent years, researchers began to segment webpages into blocks [8][35] to promote retrieval precision in web search. In passage retrieval or block retrieval works, researchers primarily care about the way of segmenting documents or webpages, and usually use the highest relevance score of a passage or block as the score of whole document or page. There are also many studies on structured document retrieval [32][55] and utilizing multiple fields of webpages for web page retrieval [44][54]. These methods linearly combine the relevance score of each field to solve the problem of scoring structured documents with multiple weighted fields. In [47], the authors show that the type of score linear combination methods is not as effective as the linear combination of term frequencies. In our work, we follow this way of handling the multiple attributes problem.

However, our work focuses on object level retrieval, which is much closer to users' requirements and considers the quality of each data source and the accuracy of the extracted object information during retrieval. This is a completely new perspective, and differs significantly from the structured document retrieval and passage/block retrieval work we discussed above.

We noticed that a need exists for document-level Web page retrieval to handle the anchor text field of a page, which is extracted from multiple Webpages [13][19]. Researchers in this area often treat all of the anchor texts as a bag of words for retrieval. There is little work which considers the quality of extracted anchor text. Moreover, since anchor text is a single field independently extracted from multiple Webpages, there is no need for unstructured retrieval. Because ignoring the structure information will not help improving the quality of the anchor text, there is no need for balancing structured and unstructured retrieval models.

The work on distributed information retrieval [9][23][38][57] is related to our work in the sense that it combines information from multiple sources to answer user queries. However, other researchers focus on selecting the most relevant search engines for queries and rank query results instead of integrating object information.

7. Conclusion

In this paper, we propose a new paradigm called object-level vertical search to enable web search at the object level. Specifically, we introduce the system architecture of such an object-level search engine and its core techniques. More importantly, we share our experience in building two real vertical search engines: Libra academic search and Window Live Product Search. Some of the techniques described in the paper including name disambiguation are still in the process of transferring to these two search engines.

We are currently working on evaluating the model in a more general way and in other application domains. We believe that our approach is generally applicable for most vertical search domains, such as Yellow Page Search, Blog Search, People Search, Job Search, and Restaurant Search.

8. Reference

[1] Arasu, A., and Garcia-Molina, H. Extracting Structured Data from Webpages. In *Proc. of ACM SIGMOD*, 2003.

[2] Balmin, A., Hristidis, V., and Papakonstantinou, Y. ObjectRank: Authority-based keyword search in databases. In *Proc. of VLDB*, 2004.

[3] Bekkerman, R., and McCallum, A. *Disambiguating Web Appearances of People in a Social Network.* In Proc. of the WWW, 2005.

[4] Bhattacharya, I., and Getoor, L. *Iterative record linkage for cleaning and integration*. In DMKD, 2004.

[5] Bunescu, R. C., and Mooney, R. J. Collective information extraction with relational Markov networks. *In Proc. of ACL*, 2004.

[6] Buttler, D., Liu, L., and Pu, C. A Fully Automated Object Extraction System for the World Wide Web. In *Proc. of IEEE ICDCS*, 2001.

[7] Cai, D., Yu, S., Wen, J.-R. and Ma, W.-Y. VIPS: a Vision-based Page Segmentation Algorithm, Microsoft Technical Report, MSR-TR-2003-79, 2003.

[8] Cai, D., Yu, S., Wen, J. R., and Ma, W. Y. Block-based web search. *In ACM SIGIR Conference*, 2004.

[9] Callan, J. Distributed information retrieval. In Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval, edited by W. Bruce Croft. Kluwer Academic Publisher, pp. 127-150, 2000.

[10] Chang, C.-H., and Liu, S.-L. IEPAD: Information Extraction Based on Pattern Discovery. In *Proc. of WWW*, 2001.

[11] Chen, Z., Kalashnikov, D.V., and Mehrotra, S. *Exploiting relationships for object consolidation*. In ACM IQIS, 2005.

[12] Cohen, W. W., and Sarawagi, S. Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. In *Proc. of SIGKDD*, 2004.

[13] Craswell, N., Hawking, D., and Roberson, S. Effective Site Finding using Link Anchor Information. In *Proceedings of SIGIR*, 2001.

[14] Crescenzi, V., Mecca, G., and Merialdo, P. ROADRUNNER: Towards Automatic Data Extraction from Large Web Sites. In *Proc. of VLDB*, 2001.

[15] Doan, A., Lu, Y., Lee, Y., and Han, J. *Object matching for information integration: a profiler-based approach.* In IIWeb, 2003.

[16] Dong, X., and Halevy, A. *A Platform for Personal Information Management and Integration*. In CIDR, 2004.

[17] Dong, X., Halevy, A., and Madhavan, J.. *Reference econciliation in Complex Information Spaces*. In Proc. Of SIGMOD, 2005.

[18] Embley, D. W., Jiang, Y., and Ng, Y.-K. Record-Boundary Discovery in Web Documents. In *Proc. of SIGMOD*, 1999.

[19] Fagin, R., Kumar, R., McCurley, K., Novak, J., Sivakumar, D., Tomlin, J., and Williamson, D. Searching the Workplace Web. In Proceedings of the Twelfth International World Wide Web Conference, 2003.

[20] Fine, S., Singer Y., and Tishby, N. The hierarchical hidden Markov model: Analysis and applications. Machine Learning, 32:41-62, 1998.

[21] Finn, A., and Kushmerick, N. Multi-level boundary classification for information extraction. In *Proc. ECML*, 2004.

[22] Ghemawat, S., Gobioff, H., and Leung, S.-T., The Google File System, In *Proc. of SOSP*, 2003.

[23] Gravano, L., and Garcia-Molina, H. Generalizing gloss to vector-space databases and broker hierarchies. In Proceeding of the International Conference on Very Large Data Bases (VLDB), 1995.

[24] Gravano, L., Lpeirotism, P., Koudas, N., Srivastava, D. *Text Joins in an RDBMS for Web Data Integration*. In Proc. Of WWW, 2003.

[25] Guo, L., Shao, F., Botev, C., and Shanmugasundaram, J. XRANK: Ranked keyword search over XML documents. In *ACM SIGMOD*, 2003.

[26] Han, H., Giles, L., Zha, H., Li, C. and Tsioutsiouliklis, K. *Two supervised learning approaches for name disambiguation in author citations.* In *JCDL* 2004.

[27] He, X., Zemel, R. S., and Carreira-Perpiñán, M. Á. Multi-scale Conditional Random Fields for Image Labeling. In *Proc. of CVPR*, 2004.

[28] Hernandez, M., and Stolfo, S. *The merge/purge problem for large datasets*. In Proc. Of the SIGMOD, 1995.

[29] Kleinberg, J. Authoritative Sources in a Hyperlinked Environment, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 1998

[30] Kushmerick, N. Wrapper induction: efficiency and expressiveness. Artificial Intelligence, 118:15-68, 2000.

[31] Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labelling sequence data. In *Proc. of ICML*, 2001.

[32] Lalmas, M. Dempster-Shafer's Theory of Evidence Applied to Structured Documents: Modeling Uncertainty. In Proceedings of SIGIR, 1997

[33] Lee, M., Hsu, W., and Kothari, V. *Cleaning the spurious links in data*. IEEE Intelligent Systems, Mar-Apr, 2004.

[34] Lerman, K., Getoor, L., Minton, S., and Knoblock, C. Using the Structure of Web Sites for Automatic Segmentation of Tables. In *Proc. of ACM SIGMOD*, 2004.

[35] Liu, B., Grossman, R. and Zhai, Y. Mining data records in webpages. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

[36] Malin, B. *Unsupervised name disambiguation via social network similarity*. In Proc. Of the Workshop on Link Analysis, Counterterrorism, and Security, in conjunction with SDM, 2005.

[37] McCallum, A., Nigam, K., and Ungar, L. *Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching*. In Proc. Of SIGKDD, 2000.

[38] Meng, M., Liu, K., Yu, C., Wu, W., and Rishe, N. Estimating the usefulness of search engines. In ICDE Conference, 1999.

[39] Michalowski, M., Thakkar, S., and Knoblock, C. *Exploiting secondary sources for unsupervised record linkage*. In IIWeb, 2004.

[40] Muslea, I., Minton, S., and Knoblock C. A. Hierarchical Wrapper Induction for Semi-structured Information Sources. Autonomous Agents and Multi-Agent 4, 1/2 (2001), 2001.

[41] Nahm, U. Y., and Mooney, R. J. A Mutually Beneficial Integration of Data Mining and Information Extraction. In *Proc. of AAAI*, 2001.

[42] Nie, Z., Zhang, Y., Wen, J.-R., and Ma, W.-Y. Object-level Ranking: Bringing Order to web Objects. In Proc. WWW, 2005.

[43] Nie, Z., Wu, F., Wen, J.-R., and Ma, W.-Y. Extracting Objects from the Web. In *Proc. of ICDE*. 2006.

[44] Ogilvie, P., and Callan, J. Combining Document Representations for known item search. In Proceedings of SIGIR, 2003.

[45] On, B., Elmacioglu, E., Lee, D., Kang, J., and Pei, J. *An Effective Approach to Entity Resolution Problem Using QuasiClique and its Application to Digital Libraries*. In JCDL 2006.

[46] Page, L., Brin, S., Motwani, R.,Winograd, T. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford Digital Library Technologies Project, 1998

[47] Robertson, S., Zaragoza, H., and Taylor, M. Simple BM25 Extension to Multiple Weighted Fields. ACM CIKM, 2004.

[48] Sarawagi, S., and Cohen, W. W. Semi-Markov Conditional Random Fields for Information Extraction. In *Proc. of NIPS*, 2004.

[49] Skounakis, M., Craven, M., and Ray S. Hierarchical Hidden Markov Models for Information Extraction. In *Proc. of IJCAI*, 2003.

[50] Song, R., Liu, H., Wen, J. R., and Ma, W. Y. Learning Block Importance Models for Webpages. In *Proc. of WWW*, 2004.

[51] Stonebraker, M., et al., C-Store: A Column Oriented DBMS. In *Proc. of VLDB*, pages 553-564, 2005.

[52] Sutton, C., Rohanimanesh, K., and McCallum, A. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. In *Proc. ICML*, 2004.

[53] Wellner, B., McCallum, A., Peng, F., and Hay, M. An Integrated, Conditional Model of Information Extraction and Coreference with Application to Citation Matching. In *Proc. of UAI*, 2004.

[54] Westerveld, T., Kraaij, W., and Hiemstra, D. Retrieving Webpages using Content, Links, URLs and Anchors. In The Tenth Text REtrieval Conference (TREC2001), 2001.

[55] Wilkinson, R. Effective Retrieval of Structured Documents. In Proceedings of SIGIR, 1994.

[56] Xi, W., Zhang, B., Chen, Z., Lu, Y., Yan, S., Ma, W.-Y. Link Fusion: A Unified Link Analysis Framework for Multi-type Inter-related Data Objects. In *Proc. of WWW* 2004.

[57] Xu, J., and Callan, J. Effective retrieval with distributed collections. In Proceedings of SIGIR, 1998.

[58] Yu, S. X., Lee, T., and Kanade, T. A Hierarchical Markov Random Field Model for Figure-Ground Segregation. *Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, September, 2001.

[59] Zhai, Y., and Liu, B. Web Data Extraction Based on Partial Tree Alignment. In *Proc. of WWW,* 2005.

[60] Zhao, H., Meng, W., Wu, Z., Raghavan, V., and Yu, C. Fully Automatic Wrapper Generation for Search Engines. In *Proc. of WWW*, 2005.

[61] Zhu, J., Nie, Z., Wen, J.-R., Zhang, B., and Ma, W.-Y. 2D Conditional Random Fields for web Information Extraction. In *Proc. of ICML*, 2005.

[62] Zhu, J., Nie, Z., Wen, J.-R., Zhang, B., and Ma, W.-Y. Simultaneous Record Detection and Attribute Labeling in web Data Extraction. In *Proc. of SIGKDD*, 2006.