Structured Positional Entity Language Model for Enterprise Entity Retrieval[*]

Chunliang Lu, Lidong Bing, Wai Lam
Dept. of Systems Engineering & Engineering Management
The Chinese University of Hong Kong
{cllu, ldbing, wlam}@se.cuhk.edu.hk

ABSTRACT

We investigate the problem of general entity retrieval for enterprise websites. Our framework transforms the webpage content into a structured content representation, which captures hierarchical information blocks and semi-structured data records information. To facilitate entity retrieval given a user query, we develop a structured positional entity language model suitable for ranking entities extracted from the webpage content incorporating the structured content representation. Different from existing language models for retrieval, our proposed model considers both the proximity and the structured webpage content in a unified manner. Extensive experiments on the benchmark datasets demonstrate the effectiveness of our proposed framework.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

Keywords

Information Retrieval; Entity Search; Semi-structured Information

1. INTRODUCTION

When we visit a website, we usually want to find some information about that organization/enterprise, such as what products they provide or who the current board members are. Much of the information need can be answered by one or more named entities, like person or organization names. An entity retrieval system can return a list of entities, instead of just documents, that can directly answer the query. It has been shown that more that 40% of Web search queries are targeted on entities [23]. Thus such capability can save lots of users' time of manual exploration. In this paper, we investigate the problem of entity retrieval for an enterprise website.

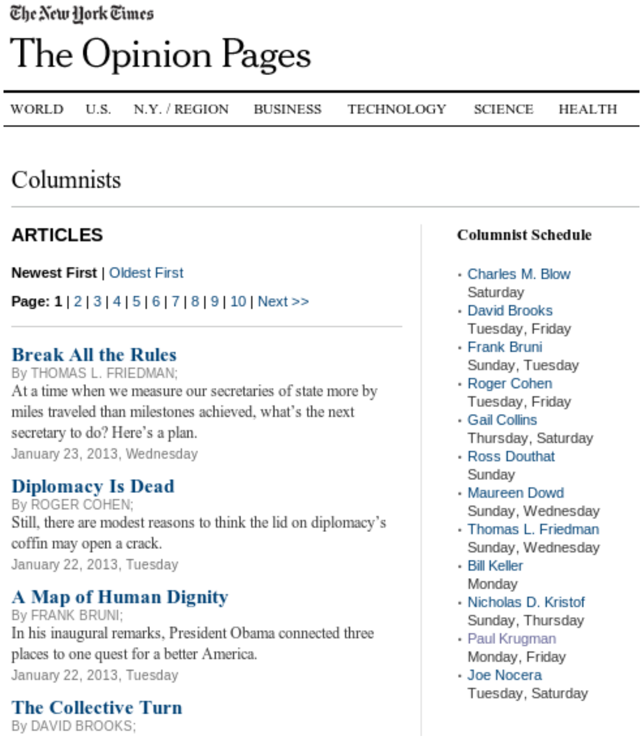
There are two related research areas, namely, enterprise search and entity retrieval. Both areas have been studied previously, but our goal is not exactly the same. For enterprise search, TREC introduced the expert finding task in the Enterprise track, together with the document retrieval task [1]. The task of document retrieval can be regarded as Web search on a single website with the goal of returning relevant webpages or documents. The task of expert finding aims at locating suitable person names for an area of expertise within an enterprise website. Expert finding can be regarded as a special form of entity retrieval, requiring the retrieval of only person entities. TREC Entity track succeeded the Enterprise track in 2009, where the expert finding task is extended to general entity retrieval in the Related Entity Finding (REF) task [3]. For each query, given the source entity name, together with its URL and the target entity type, we need to retrieve a set of entities satisfying the query narrative. Compared to the Enterprise track, answer entities do not need to be confined to an enterprise website. Some existing methods for entity retrieval follow a typical question answering approach [26, 27]. The problem we investigate in this paper shares some characteristics with entity retrieval, but we focus on finding general entities as answers in a given enterprise site. We locate the answer entities mainly from the corresponding enterprise website since the information on the enterprise website is more reliable. Thus the objective can also be regarded as an extension to the traditional enterprise search.

To retrieve relevant entities for a query, we make use of the structured content information of webpages. We observe that people tend to organize webpages into a hierarchical structure for clear presentation and easy navigation. Similar to the fact that an ordinary document is organized into sections and subsections, a single webpage also typically possesses hierarchical content structure. For example, the page segment shown in Figure 1 has three levels of headings, with the terms "The Opinion Pages" as the top-level heading, the left block with the heading "ARTICLES" and the right block with the heading "Columnist Schedule" sharing the second-level heading "Columnist". Besides, in most websites, multiple webpages are organized in a hierarchical structure, which

[*]The work described in this paper is substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: CUHK413510) and the Direct Grant of the Faculty of Engineering, CUHK (Project Code: 2050522). This work is also affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.
Copyright 2013 ACM 978-1-4503-2263-8/13/10 ...$15.00.
http://dx.doi.org/10.1145/2505515.2505702.

is usually indicated by the navigation menu and commonly referred as the logical sitemap [27]. When finding information from webpages, we humans often make use of clues or evidences from the structure and heading information. For example, consider the query "Find the regular opinion columnists of the New York Times"[1]. In the webpage from The New York Times website shown in Figure 1, the answers, such as "Charles M. Blow", are located in the right block with the heading "Columnists Schedule". Normally we will focus on that block of information to find answers after examining the webpage structure, ignoring names mentioned in other sections.



Break All the Rules
By THOMAS L. FRIEDMAN;
At a time when we measure our secretaries of state more by miles traveled than milestones achieved, what's the next secretary to do? Here's a plan.
January 23, 2013, Wednesday

Diplomacy Is Dead
By ROGER COHEN;
Still, there are modest reasons to think the lid on diplomacy's coffin may open a crack.
January 22, 2013, Tuesday

A Map of Human Dignity
By FRANK BRUNI;
In his inaugural remarks, President Obama connected three places to one quest for a better America.
January 22, 2013, Tuesday

The Collective Turn
By DAVID BROOKS;

- Charles M. Blow
  Saturday
- David Brooks
  Tuesday, Friday
- Frank Bruni
  Sunday, Tuesday
- Roger Cohen
  Tuesday, Friday
- Gail Collins
  Thursday, Saturday
- Ross Douthat
  Sunday
- Maureen Dowd
  Sunday, Wednesday
- Thomas L. Friedman
  Sunday, Wednesday
- Bill Keller
  Monday
- Nicholas D. Kristof
  Sunday, Thursday
- Paul Krugman
  Monday, Friday
- Joe Nocera
  Tuesday, Saturday

Figure 1: Page segment from The New York Times website

Aiming at utilizing and exploiting the structured content of webpages, we develop a framework for tackling the entity retrieval problem in a rigorous manner by proposing a structured positional entity language model. We first analyze and process webpages on an enterprise website to extract the site structure and transform the webpage content into a structured content representation. The resulting representation captures the webpage content using hierarchical information blocks. Named entity detection is conducted to locate all entities in the content. To facilitate entity retrieval given a query, we develop a structured positional entity language model incorporating the structured content representation of the webpage. The proposed entity language model is used to measure the relevance of each entity in a webpage to the given query and facilitate the entity ranking. Different from existing language models for retrieval, our proposed model considers both the proximity and the structured page content in a unified manner. The proximity principle prefers the entities near the occurrence of query terms, and at the same time, the structured content representation facilitates the consideration of the heading and the semi-structured information. Returning to the entity retrieval example described above regarding The New York Times columnists, the person entity "Charles M. Blow" can be found as an answer based on the evidence that the query term "columnist" appears near the entity in the heading of the same block. Furthermore, despite the fact that the text "The Opinion Pages" appears quite a distance away from the answer text in the raw webpage content, such text can be treated as a top-level heading text of the block containing the person names. This structured view of the webpage content facilitates a desirable increase in the confidence that the person names are the answers since the term "opinion" appeared in the heading text matches with a similar term in the query.

Proximity-based language models have been employed in information retrieval, as well as expert finding. For example, the models proposed in [20] performs expert finding by employing proximity-based document representation. The positional language model in [18] makes use of proximity to tackle the document retrieval by constructing a language model for each position in the document. Another related research area is XML retrieval which has been extensively investigated in various INEX[2] tracks, such as the Linked Data Track [25]. Our proposed model differs from the above mentioned methods, as we consider both the structure of the webpage content and the proximity in an entity language model in a unified manner. Moreover, the entity retrieval problem investigated in this paper has a more challenging problem setting compared with XML retrieval. More details are presented in the next section.

2. RELATED WORK

Two closely related research areas are enterprise search and entity retrieval, both of which have been investigated previously. For enterprise search, TREC introduced the Enterprise track in 2005, featuring two tasks: document retrieval and expert finding. The task of document retrieval can be regarded as Web search on a single website, with the goal of returning relevant webpages or documents. The task of expert finding can be regarded as a special form of entity retrieval, requiring the retrieval of only person entities for an area of expertise within an enterprise website. The language model approach for expert finding was first proposed by Balog et al. [2]. Petkova and Croft [20] used proximity-based document representation for expert finding. TREC Entity track succeeded Enterprise track in 2009, extending expert finding to general entity retrieval in the Related Entity Finding (REF) task. Compared to the Enterprise track, answer entities do not need to be confined to an enterprise website. Most existing works follow a question answering approach. In 2011, Wang et al. [26] developed a method that combines their Document-Centered Model with affinity score between candidate entities and keywords to rank the entities. In 2010, Yang et al. [27] used the reconstructed logical hierarchical sitemap to enhance retrieval. In 2009, Fang et al. [10] proposed a hierarchical relevance retrieval model, considering document, passage, and entity for entity retrieval. Bron et al. [7] performed a detailed analysis on four core components, namely, co-occurrence models, type

[1] It is derived from Query 41 of TREC Entity track in 2010.

[2] `https://inex.mmci.uni-saarland.de/`

filtering, context modeling, and homepage finding. Kaptein et al. [14] exploited Wikipedia as a pivot to perform entity ranking. Our enterprise entity retrieval problem can be regarded as an extension to the traditional enterprise search.

The entity search using Semantic Web has been proposed and investigated recently [23, 6, 24, 13]. The semantic search makes use of existing Linked Data as sources to perform retrieval. In our problem setting, the involved enterprises usually do not have well-formatted semantic data, which makes it difficult to apply semantic search for an enterprise. In contrast, our framework makes use of the existing webpages on the enterprise website to conduct entity retrieval. The research area of XML retrieval heavily relies on the structure of the XML file, and the problem has been extensively investigated in various INEX tracks such as the Linked Data Track [25]. However, usually the goal of XML retrieval is to return a list of relevant XML nodes. In some XML corpora, each major entity is already associated with a XML node in a well-structured context. The entity retrieval problem investigated in this paper has a more challenging problem setting. With the objective of returning a list of relevant entities that can answer the user's given query, we need to extract entities from webpages and construct a structured context for entities, since webpages do not have a well-defined structure contrasting to XML files.

Proximity-based models have been used in information retrieval and expert finding. For example, Petkova and Croft [20] developed a proximity-based document representation model to rank person entities for tackling the problem of expert finding. Lv and Zhai [18] proposed the positional language model (PLM) for information retrieval, and demonstrated the effectiveness over other proximity-based models. However, PLM is mainly used for document ranking. Our proposed entity language model differs from the above mentioned models, as we explicitly construct an entity language model for each entity by considering both the proximity and the structure of the webpage in a unified manner.

Entity retrieval is closely related to question answering, and there are some previous works investigating question answering using webpages. Pinto et al. [21] proposed to tackle question answering using semi-structured data found on webpages. Yin et al. [28] studied the problem of structured knowledge extraction from attribute-value Web tables. Question answering systems usually take a three-step approach, namely, query analysis, finding relevant documents, and answer extraction. They usually do not distinguish and take advantage when focusing on a domain or an enterprise on the Web.

There are various approaches to extract structure from webpages. Webpage segmentation aims at dividing webpage content into coherent groups. Kohlschutter and Nejdl [16] made use of text-density as a measure to identify the individual text segments. Chakrabarti et al. [8] formulated the segmentation as a graph optimization problem. Beside the segmentation, semi-structured record extraction aims to extract structured content information presenting similar entities as well as their attributes, which is useful for various applications such as knowledge base population [5]. Miao et al. [19] investigated a method based on the tag paths to preform record detection. Bing et al. [4] proposed RST structure to facilitate the record set detection. Our structured content representation of a webpage considers the semi-structured data as well as logical relationship of the page content.

3. STRUCTURED CONTENT GENERATION

As mentioned before, we develop a framework for tackling the entity retrieval problem by developing a structured positional entity language model. One core aspect of our proposed entity language model is the structured content generated from the webpage where the entity is located. A webpage has an inherent structure called Document Object Model (DOM). However, the DOM structure is not suitable for constructing entity language model, since it is mainly used to describe the page layout rather than semantic relationship among terms in the document. There are some existing work to represent structured document [17], however they are mainly used to enhance document retrieval by content or by structure. In our framework, the structure of the document is represented as a tree whose nodes, referred to as *information blocks*, correspond to some segment of the webpage, such as a heading or a section containing several paragraphs. The leaf nodes comprise the minimal page segment, such as a single paragraph, and any non-leaf nodes constitute a larger semantically relevant information block.

We start by crawling all the webpages under the enterprise domain, and performing site-level analysis such as webpage template detection and link extraction. The Wikipedia articles about the enterprise, if any, are also collected. All the collected webpages are transformed to a hierarchical structured representation. Entity detection and resolution are then performed to find named entities.

3.1 Site-level Structure Extraction

A webpage template is defined as a segment that appears on more than two webpages on the same domain, such as the webpage footer. They are widely used in websites to provide a uniform appearance. We remove all the webpage templates using the boilerpipe library[3] [15] from all webpages since they normally do not contain useful information for retrieval. A webpage may have multiple URLs, and we may get duplicated pages with exactly the same content in the crawling process. These pages are merged in this step.

Internal links between webpages within the same website play an important role. The anchor text usually gives a good description about the target page, thus it provides a site-level page context. For all the pages, we find the incoming links and the corresponding anchor texts from other pages in the same domain. The navigation menu in a website provides a logical organization of webpages. We locate the navigation menu and extract the website hierarchical logical structure following the method proposed in [27]. Take the official website of CIKM 2013 as an example. The "Call for Papers" page is linked in the "Call For Papers" item of the "Participants" menu. We extract the higher level menu text, such as "Participants" in this example, and attach it in the meta-heading of the page.

3.2 Page-level Structured Content Generation

We transform each webpage into a structured representation, named as *information block*. An information block is defined as a hierarchical structure, which either contains a list of children blocks, or corresponds to a page segment.

[3] Available at `https://code.google.com/p/boilerpipe/`

Each block contains page content with relatively coherent format, and is optionally associated with a heading block. Two or more sibling blocks exhibiting similar layout format form a set of semi-structured data records.

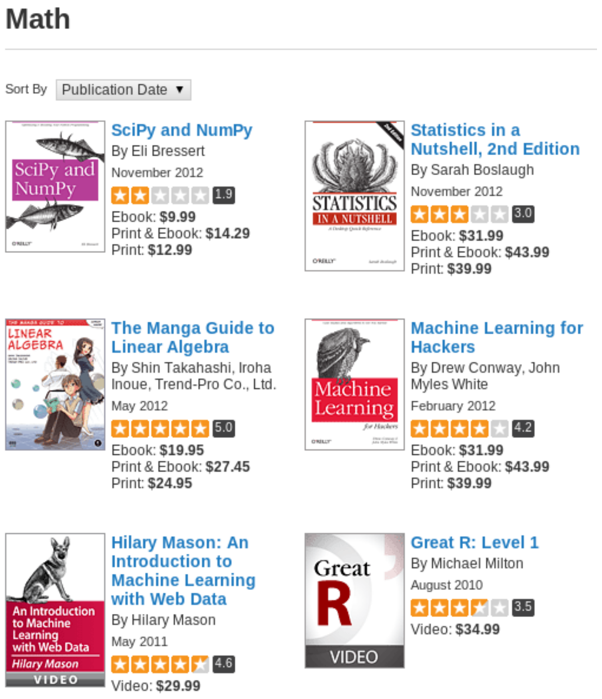
```
   <infoblock> ::= [<heading>] (<block>|<recordsblock>)+
<recordsblock> ::= <recordblock> <recordblock>+
 <recordblock> ::= <field>+
       <field> ::= [<heading>] <fieldvalue>+
  <fieldvalue> ::= <block>
     <heading> ::= <block>
```

Figure 2: BNF representation of information block

The formal BNF definition for the notion of information block is depicted in Figure 2. We use the page segment shown in Figure 3 as an example. The content of the webpage can be split into page segments of different sizes, denoted as `<block>`s. The whole page segment can be regarded as an information block denoted as `<infoblock>`. A record set information block, denoted as `<recordsblock>`, is a special kind of information block that contains two or more similarly formatted record blocks denoted as `<recordblock>`. There are six record blocks in Figure 3, where each block depicts a single book, and these six record blocks form a record set block. A ordinary Web table is also regarded as a record set block, with each table row as a record block. A record block consists of one or more fields denoted as `<field>`, which are aligned into field columns with fields in other record blocks within the same record set block. A field can have multiple field values, denoted as `<fieldvalue>`, and field values in the same field column generally share the same format and content type. Fields in the same field column may have different number of field values. Returning to the example in Figure 3, each record block has six fields, namely, book image, title, author, published date, rating, and price. The price field has three values for the first four blocks, namely, "Ebook", "Print & Ebook", and "Print", while the last two blocks only have one value. A block can have a heading block denoted as `<heading>`. In this example, the block with the content "Math" is a heading block, and it is associated with the record set block mentioned above.

To obtain such a hierarchical structured content representation, we first perform data record detection on all the webpages, using the RST method proposed in [4]. After the data records are detected, the fields within each data record is aligned by the Partial Tree Alignment algorithm [31]. Heading information are detected by the HTML tags `h1` to `h6`. HTML tables are preprocessed such that the potential field column heading information encoded in HTML `<th>` tag is preserved and assigned to the corresponding table columns.

For instance, the resulting structured content representation for the page segment in Figure 3 is shown in Figure 4. This hierarchical representation can be regarded as an ordered tree structure. In the representation, all the leaf blocks, denoted as gray blocks in Figure 4, correspond to some page segments.



Math

Sort By Publication Date ▾

SciPy and NumPy
By Eli Bressert
November 2012
1.9
Ebook: $9.99
Print & Ebook: $14.29
Print: $12.99

Statistics in a Nutshell, 2nd Edition
By Sarah Boslaugh
November 2012
3.0
Ebook: $31.99
Print & Ebook: $43.99
Print: $39.99

The Manga Guide to Linear Algebra
By Shin Takahashi, Iroha Inoue, Trend-Pro Co., Ltd.
May 2012
5.0
Ebook: $19.95
Print & Ebook: $27.45
Print: $24.95

Machine Learning for Hackers
By Drew Conway, John Myles White
February 2012
4.2
Ebook: $31.99
Print & Ebook: $43.99
Print: $39.99

Hilary Mason: An Introduction to Machine Learning with Web Data
By Hilary Mason
May 2011
4.6
Video: $29.99

Great R: Level 1
By Michael Milton
August 2010
3.5
Video: $34.99

Figure 3: Page Segment Example (taken from O'Reilly website)

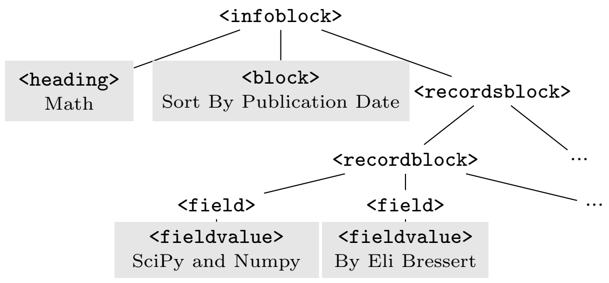


Figure 4: Structured content representation for the page segment in Figure 3

3.3 Entity Detection

After the structured content representation for each page is obtained, we perform named entity extraction on each information block. Stanford Named Entity Recognizer [11] is used to perform NER on all the leaf blocks. The content in semi-structured record sets has not much context information and the entities in record sets may not be detected by the above NER tool. To handle such text data, we make use of existing entity repositories such as Wikipedia and Freebase to detect potentially missed entities. After the initial detection, we continue the detection by exploiting the structured content representation. If multiple field values are detected as certain kind of entity, we have high confidence that other fields in the same field column also contain entities with the same entity type. This technique enables the detection of entities not found in existing entity repositories.

Usually the same entity will be mentioned multiple times at different positions of the same page or across several pages. An entity may appear in different forms such as "California" and "CA". This raises the problem of entity resolution, which recognizes different forms of the same entity and treats them to be the same. We rely on several clues to achieve this goal. Entity aliases in existing entity repositories, such as topic aliases in Freebase and page redirections in Wikipedia, are used to conduct the resolution. Two entities pointing to the same internal webpage also have high probability to be the same entity. We also take word synonyms and phrase abbreviations into consideration.

4. ENTITY LANGUAGE MODEL

The language modeling approach is quite effective and has been widely used for document retrieval [22, 9, 29]. The idea is that since users usually issue queries using the keywords that would likely appear in a relevant document, a document is a good match to a query if the document is likely to generate the query. We adopt the similar idea in entity retrieval, where we first construct a language model for each entity, and then rank the entities based on the probability that the entity's language model generates the keywords in the query. Different from language modeling for document retrieval, where the language model is estimated from the words in the document, we have no "document" for the entity. The only data we have is the webpage where the entity resides, but apparently we cannot directly use the webpage content to estimate the entity language model, since that document is not meant to solely describe the entity. We need to construct a virtual document for each entity that can describe the entity, based on the webpage where entities are extracted.

Inspired by the proximity-based approaches in [18, 20], we construct a virtual document based on the webpage content. Since the terms in the same webpage as the entity exhibit some relationships with the entity, the virtual document of the entity is constructed based on the terms in the webpage content. Specifically, the terms in the virtual document for the entity are propagated from the terms found in the webpage. The language model for an entity e in the webpage content $\mathcal{D}$ can be formulated as:

$$p(t|e,\mathcal{D}) = \frac{c(t,e,D)}{\sum_{t' \in \mathcal{V}} c(t',e,D)}, \qquad (1)$$

where t denotes a term; $\mathcal{V}$ is the virtual document for the entity e; $c(t,e,D)$ is the total propagation count of the term t from all the positions in the webpage content $\mathcal{D}$. The propagation count will take into consideration of the proximity, measured by the distance between the appearance of the term t and the entity e, as well as the structured content of the webpage.

4.1 Structured Information Propagation

Let us consider a webpage D, as exemplified in Figure 3, we transform it into a structured representation as exemplified in Figure 4. All the terms are found in the leaf blocks, denoted as $\{\mathcal{B}^l\}$. For a given entity e found in the leaf block $\mathcal{B}_e$ corresponding to the webpage D, the propagation count for the term t is formulated as:

$$c(t,e,D) = \sum_{\mathcal{B} \in \{\mathcal{B}^l\}} r(\mathcal{B}_e,\mathcal{B}) \sum_{j=1}^{|\mathcal{B}|} \mathbb{1}_{t_j=t} k(d_{\mathcal{B}_e,\mathcal{B}}(e,j)), \qquad (2)$$

where $r(\mathcal{B}_e,\mathcal{B})$ is used to indicate the relevance for the terms in the block $\mathcal{B}$ with respect to the block $\mathcal{B}_e$ containing the entity e; $d_{\mathcal{B}_e,\mathcal{B}}(e,j)$ is the distance function between the entity e and the term at the position j in the block $\mathcal{B}$; $\mathbb{1}_{t_j=t}$ is a binary function indicating whether the term t appears at the position j in the block $\mathcal{B}$; k is the propagation kernel function. By designing different relevance function r and distance function d, we obtain different propagation schemas exploiting the structured content information in webpages. We first investigate various components that we need to consider when designing the propagation schema. Then we continue to present two propagation models in Section 4.2.

4.1.1 Proximity

The proximity principle, which considers the distance between terms and entities, has been extensively used in document retrieval and expert finding [18, 20]. Proximity is also one major component in our entity language model. Terms near the entity should get higher propagation count since they possess stronger relationship with the entity. By incorporating a non-uniform and non-increasing proximity kernel function, the proximity principle favors more on the terms that appears near the entity. The use of proximity can be integrated into our model by setting an appropriate distance function $d_{\mathcal{B}_e,\mathcal{B}}(e,j)$ as depicted in Equation 2.

4.1.2 Block relevance

The block relevance function r measures the relationship between the entity block and the term block. Since the structured content representation inherits semantic relationship between webpage segments, one intuitive relevance measure of two blocks is their relative positions in the tree-structured representation. The shortest path between two blocks, or equivalently, the average of the distances to their lowest common parent, is able to capture such notion of relevancy. Let us define the function that locates the lowest common ancestor block in the structured content representation for the blocks $\mathcal{B}_i$ and $\mathcal{B}_j$ as $lca(\mathcal{B}_i,\mathcal{B}_j)$. The relevance measure $r(\mathcal{B}_e,\mathcal{B})$ for the entity block $\mathcal{B}_e$ and the term block $\mathcal{B}$ is proportional to their shortest path as follows:

$$\begin{aligned} r(\mathcal{B}_e,\mathcal{B}) &\propto \mathit{ShortestPath}(\mathcal{B}_e,\mathcal{B}) \\ &\propto \frac{1}{2}(v(lca(\mathcal{B}_e,\mathcal{B}),\mathcal{B}_e) + v(lca(\mathcal{B}_e,\mathcal{B}),\mathcal{B})), \end{aligned} \qquad (3)$$

where $v(\mathcal{B}_i,\mathcal{B}_j)$ is the number of vertices between blocks $\mathcal{B}_i$ and $\mathcal{B}_j$.

4.1.3 Heading blocks

Heading blocks play an important role since they are intended to describe certain aspect for all the content under the heading. Thus, terms in the heading block should be considered as highly relevant context information even though the entity may be far away from the heading. We denote all the heading blocks as $\{\mathcal{H}\}$, and define a function $parent(\mathcal{B})$ to indicate the parent block for the block $\mathcal{B}$. The set of heading blocks $\mathbb{H}_e$ for the entity e can be represented as:

$$\mathbb{H}_e = \{\mathcal{H} : lca(\mathcal{H},\mathcal{B}_e) = parent(\mathcal{H})\}. \qquad (4)$$

Site menu and page incoming link anchor texts, if any, can be integrated into the hierarchical heading model since they provide a description for the whole webpage content. For example, if a page with a title "Insurance" is under the "Service" menu, all the entities in this page will get the propagation count from the term "service". We treat the site menu and anchor texts as the highest-level heading blocks.

4.1.4 Context blocks

One major component is to find the context for a given entity e. Basically, all the leaf blocks except heading blocks are regarded as the context blocks. However, if the entity

e is located in a record block, we need to refine the content based on the properties of record sets. We usually use records to represent parallel information, meaning that the terms in one record just provide information for that record. For example, if we want to locate the books written by some authors in Figure 3, only the entities in the blocks that share the same record block with the author name block are relevant. Thus it is appropriate to ignore other records in the same record set when we are constructing entity language model for entities appearing in a record. We denote all the record set blocks as $\{\mathcal{R}\}$. Suppose that the entity e is located in the leaf block $\mathcal{B}_e^l$, then the set of the record blocks $\mathbb{R}_e$ which share the same record set with the entity e is defined as:

$$\mathbb{R}_e = \{\mathcal{B}^l : lca(\mathcal{B}^l, \mathcal{B}_e^l) \notin \{\mathcal{R}\}\}. \tag{5}$$

This definition also includes the nested record blocks in higher-level record sets. When performing term propagation, we just need to consider the blocks that are not in $\mathbb{R}_e$, referred to as context blocks and denoted by $\mathbb{C}_e$:

$$\mathbb{C}_e = \{\mathcal{B}^l : \mathcal{B}^l \notin \mathbb{R}_e, \mathcal{B}^l \notin \{\mathcal{H}\}\}. \tag{6}$$

For the example page segment in Figure 3, if we are constructing the context information for all the person entities appeared in the page, all the blocks under the same book record will be included as entity context such as the book title, but the blocks in other book records will be excluded.

4.2 Propagation Schema

As mentioned in Equation 2, a propagation schema is composed of two components, namely, the block relevance function r and the distance function d. We propose two structured propagation models by designing different propagation schemas.

4.2.1 Structured Propagation Model 1

Considering all the factors, the relevance function should depend on the block type, such as whether it is a heading block, and the relative position of the block, such as the shortest path between blocks. The distance function should take into consideration of the term position inside the block. We propose Structured Propagation Model 1 by designing each component in Equation 2 as follows:

$$r(\mathcal{B}_e, \mathcal{B}) = \begin{cases} (1-\beta)\cdot \mathit{ShortestPath}(\mathcal{B}_e, \mathcal{B}) & \mathcal{B} \in \mathbb{C}_e \\ \beta \cdot \mathit{ShortestPath}(\mathcal{B}_e, \mathcal{B}) & \mathcal{B} \in \mathbb{H}_e \\ 0 & \text{otherwise} \end{cases}$$

$$d_{\mathcal{B}_e,\mathcal{B}}(e,j) = \begin{cases} |\mathcal{B}| - j & \mathcal{B} \prec \mathcal{B}_e \\ j & \mathcal{B} \succ \mathcal{B}_e \\ abs\,(j - I_{\mathcal{B}_e}(e)) & \mathcal{B} \equiv \mathcal{B}_e \end{cases}$$

where $|\mathcal{B}|$ denotes the total number of terms in the block $\mathcal{B}$; $I_{\mathcal{B}_e}(e)$ indicates the position of the entity e in the block $\mathcal{B}_e$; abs denotes the absolute function. The order of the leaf blocks, indicated by the symbols $\succ$ and $\prec$, is determined by their relative position in the original webpage.

In this model, the relevance measure r is different for heading blocks and context blocks, and we can adjust the importance of heading blocks by the parameter β. Moreover, the distance function d just depends on the relative position of the term j inside the block $\mathcal{B}$.

4.2.2 Structured Propagation Model 2

In Structured Propagation Model 1, two terms in different blocks may have the same propagation counts, if their corresponding blocks have the same shortest path to the entity block and the terms have the same relative position inside the block. Another strategy is to consider the absolute term distance in the original webpage content. The relevance function r and the distance function d in Equation 2 can be written as:

$$r(\mathcal{B}_e, \mathcal{B}) = \begin{cases} 1-\beta & \mathcal{B} \in \mathbb{C}_e \\ \beta & \mathcal{B} \in \mathbb{H}_e \\ 0 & \text{otherwise} \end{cases}$$

$$d_{\mathcal{B}_e,\mathcal{B}}(e,j) = \begin{cases} |\mathcal{B}| - j + \sum\limits_{\mathcal{B}\prec\mathcal{C}\prec\mathcal{B}_e} |\mathcal{C}| + I_{\mathcal{B}_e}(e) & \mathcal{B} \prec \mathcal{B}_e \\ |\mathcal{B}_e| - I_{\mathcal{B}_e}(e) + \sum\limits_{\mathcal{B}_e\prec\mathcal{C}\prec\mathcal{B}} |\mathcal{C}| + j & \mathcal{B} \succ \mathcal{B}_e \\ abs\,(j - I_{\mathcal{B}_e}(e)) & \mathcal{B} \equiv \mathcal{B}_e \end{cases}$$

where $\sum_{\mathcal{B}\prec\mathcal{C}\prec\mathcal{B}_e} |\mathcal{C}|$ sums over the number of terms in all the leaf blocks in $\mathbb{H}_e$ or $\mathbb{C}_e$ between $\mathcal{B}$ and $\mathcal{B}_e$, depending on whether $\mathcal{B}$ is a heading block or a context block. In this model, no two terms would have the same propagation counts.

We can continue to derive this model and obtain a form by concatenating the terms in the context blocks $\mathbb{C}_e$ into a single pseudo-context document $\mathcal{I}$, and the terms in the heading blocks $\mathbb{H}_e$ into a single pseudo-heading document $\mathcal{J}$. Such derivation can support more efficient implementation of the model. As a result, the propagation count $c(t,e,D)$ for term t can be written as:

$$c(t,e,D) = (1-\beta)\sum_{j=1}^{|\mathcal{I}|} \mathbb{1}_{t_j=t}\, k(d_{\mathcal{I}}(e,j)) + \beta \sum_{j=1}^{|\mathcal{J}|} \mathbb{1}_{t_j=t}\, k(d_{\mathcal{J}}(e,j)), \tag{7}$$

where $d_{\mathcal{L}}(e,j)$ is the number of terms between the entity e and the position j in the pseudo-document $\mathcal{L}$.

4.3 Relation to Existing Proximity-based Retrieval Model

Our proposed structured information propagation, as depicted in Equation 2, generalizes the traditional proximity-based retrieval models, such as the positional language model used in document retrieval [18] and the proximity-based entity retrieval model in [20]. If we set the block relevance measure r to be a constant and the distance function d to be the number of terms between the entity and the propagated term, it can be easily shown that the model is reduced to the proximity-based entity retrieval model proposed in [18]. Hence, the traditional proximity-based retrieval model is just a special case in our propagation model without considering structured content information.

4.4 Propagation Kernels

Any non-uniform, non-increasing function can be used as the propagation kernel function k as depicted in Equation 2. Following the previous work [18], we investigate three different representative kernel functions, namely, Gaussian kernel, Triangle kernel, and Circle kernel, as shown in Figure 5.

1. Gaussian kernel

$$k(d) = exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{8}$$

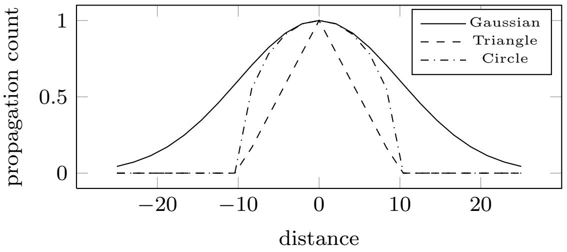


Figure 5: Propagation kernels ($\sigma = 12.5$)

2. Triangle kernel

$$k(d) = \begin{cases} 1 - \frac{|d|}{\sigma} & \text{if } d \leq \sigma \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

3. Circle kernel

$$k(d) = \begin{cases} \sqrt{1 - \left(\frac{|d|}{\sigma}\right)^2} & \text{if } |d| \leq \sigma \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where σ controls the spread of kernel curves.

4.5 Smoothing

One issue in language modeling estimation is smoothing. Since we only have a finite set of terms for each webpage, the maximum likelihood estimation may assign zero probability to the terms not found in the webpage, leading to undesirable results. To tackle this problem, we include a collection language model to provide a background probability [30] for all the terms. We investigate two popular smoothing methods, namely, Dirichlet prior and Jelinek-Mercer.

- Dirichlet smoothing

$$p_\mu(t|e, \mathcal{D}) = \frac{c(t|e, \mathcal{D}) + \mu p(t|\mathcal{C})}{Z_e + \mu} \tag{11}$$

- Jelinek-Mercer method

$$p_\lambda(t|e, \mathcal{D}) = (1 - \lambda)p(t|e, \mathcal{D}) + \lambda p(t|\mathcal{C}) \tag{12}$$

where μ and λ are the smoothing parameters, $p(t|\mathcal{C})$ is the collection language model, and $Z_e = \sum_{t \in \mathcal{V}} c(t, e)$ is the length of the virtual document for the entity e.

5. ENTITY RETRIEVAL

The last component in our framework handles the entity retrieval for a given query. User's query is first analyzed to identify stop structure and key terms. Retrieved entities are ranked based on the probability that the entity language model, as presented in Section 4, generates the query terms.

5.1 Query Analysis

When dealing with verbose queries, we follow the stop structure removal method proposed in [12]. A stop structure is defined as a phrase which provides no information about the information need, such as "Find the homepages of" or "Tell me the". Some of the query narratives mention the enterprise name. For example, for the Blackberry website, the query "Carriers that *Blackberry* makes phones for" contains the enterprise entity name. These terms can be removed from the key terms since all the webpages under the enterprise website are all implicitly related to the enterprise name. Entity type information is considered in the retrieval model as described in Section 5.3.

A user may issue queries using different words with the same meaning. We conduct two kinds of query term expansion. One is for named entity terms and the other is for non-entity general terms. For named entity terms, various synonyms are added to the keyword terms from the Freebase aliases list, such as "Philly" for "Philadelphia". Acronyms in user's query are expanded into full names and then added to the key terms. For non-entity terms, WordNet synonyms and hyponyms for the terms are included. For example, if the query contains the word "musician", then the term "person" or "artist" will also be included in the key terms. This may broaden the query to some extent, nevertheless, it performs reasonably well in our experiments.

5.2 Entity Ranking

Using the Bayes' rule, the probability that a candidate entity e is an answer entity for a given query $\mathcal{Q}$ can be written as:

$$p(e|\mathcal{Q}) = \frac{p(\mathcal{Q}|e)p(e)}{p(\mathcal{Q})}, \tag{13}$$

where $p(\mathcal{Q})$ is the probability of the query; $p(e)$ is the prior probability of the candidate entity e; and $p(\mathcal{Q}|e)$ is the probability of a query given the candidate. $p(\mathcal{Q})$ is the same for all the candidate entities, and it is typical to assume the distribution of $p(e)$ is uniform. Thus, the ranking of the candidate entities is proportional to the probability of the query given the entity $p(\mathcal{Q}|e)$.

The same entity may appear multiple times in the same page or across different pages. We find all the occurrences of the entities, and rank them by the maximal probability of the query $\mathcal{Q}$ given the entity e in the page content $\mathcal{D}$ as follows:

$$p(\mathcal{Q}|e) = \max_{\mathcal{D}} p(\mathcal{Q}|e, \mathcal{D}). \tag{14}$$

The structured positional entity language model $p(t|e, \mathcal{D})$ related to the term t given an entity e expressed in Equation 1 can be employed to compute $p(\mathcal{Q}|e, \mathcal{D})$. Precisely, we use the multinomial unigram language model to estimate the probability that the structured positional language model generates the query terms t as follows:

$$p(\mathcal{Q}|e, \mathcal{D}) = K_q \prod_{t \in \mathcal{Q}} p(t|e, \mathcal{D})^{tf_{t,\mathcal{Q}}}, \tag{15}$$

where $K_q = L_{\mathcal{Q}}!/(tf_{t_1,\mathcal{Q}}! \cdot tf_{t_2,\mathcal{Q}}! \cdots tf_{t_M,\mathcal{Q}}!)$ is the multinomial coefficient for the query $\mathcal{Q}$; $L_{\mathcal{Q}}$ is the length of the query $\mathcal{Q}$; and $tf_{t,\mathcal{Q}}$ is the term frequency for the term t in the query $\mathcal{Q}$. K_q can be ignored since it is a constant for a particular query. Thus, the entity ranking becomes as follows:

$$p(e|Q) \propto \max_{\mathcal{Q}} \prod_{t \in \mathcal{Q}} p(t|e, \mathcal{D})^{tf_{t,\mathcal{Q}}}. \tag{16}$$

5.3 Entity Type Evidence

Our framework also considers evidence from the entity type information if the target entity type is given for the query. We mainly make use of entity type information to filter out irrelevant entities. Entities that do not meet the target entity type in the query are removed from the retrieved entities in this component.

Precisely, suppose that the target entity type extracted from the query is denoted as $\mathcal{T}$. For a particular entity candidate e, detected from the page content, it is associated with a set of entity types $\mathbb{T}_e$. We use the Wikipedia category structure to indicate whether the entity e satisfies the query target type $\mathcal{T}$, denoted as $p(\mathcal{T}|e)$:

$$p(\mathcal{T}|e) = \begin{cases} 1 & cat(\mathcal{T}) \cap \{\mathbb{T}_e \cup \{par(v), v \in \mathbb{T}_e\}\} \neq \varnothing \\ 0 & \text{otherwise} \end{cases}$$

where $par(v)$ gives all the parent categories for the entity type/category v in the Wikipedia category tree, and $cat(\mathcal{T})$ maps the target entity type to some Wikipedia categories. This mapping can be easily prepared in advance.

6. EXPERIMENT

We make use of the datasets used in TREC Entity track involving the ClueWeb09 corpus. There are two main sets of experiments to address several research questions. The first set of experiments aims at analyzing the performance of our model and conducting comparison to an existing entity retrieval model. We also analyze the effects of different kernel functions and smoothing methods. In the second set of experiments, we wish to compare the performance of our framework with the previously reported results by the participants in the Related Entity Finding (REF) task of the TREC Entity track.

6.1 Dataset and Experiment Setup

We run our experiments on the datasets derived from the TREC Entity track. An example query is given in Figure 6. For each query, besides the query narrative, the enterprise website (the entity URL specified by the ClueWeb09 ID) and the target entity type are also given. The evaluation dataset for the REF task of TREC Entity track in 2010, which is composed of 70 queries, is referred to as the TE10 dataset. The evaluation dataset for the REF task in 2011 containing another 50 queries is referred to as the TE11 dataset. The webpages under the enterprise domain crawled in the ClueWeb09 corpus, together with the enterprise Wikipedia pages, if any, are used as sources to locate the answer entities.

```
<query>
 <num>2</num>
 <entity_name>ACM Athena award</entity_name>
 <entity_URL>clueweb09-en0004-21-12770</entity_URL>
 <target_entity>person</target_entity>
 <narrative>Winners of the ACM Athena award.</narrative>
</query>
```

Figure 6: An example query

For some queries in the REF task of TREC Entity track, some answer entities only exist in webpages that are not under the enterprise websites. Another issue is that the REF task evaluates the performance based on the entity's homepage. However, not all the entities have homepages, such as some person answer entities. To conduct experiments more suitable for our objective of enterprise entity retrieval, we selected a subset of queries that have answer entities in the enterprise website. In addition, we manually re-annotated all the answer entities to include the correct entity names, the entity's homepages, and the corresponding Wikipedia pages if any. As a result, the evaluation can be done based on either the entity names, the entity's Wikipedia page or the entity's homepages. This dataset is referred to as the TE-E dataset and is publicly available[4]. Some characteristics of these three datasets are summarized in Table 1. The last row indicates the average number of webpages on the enterprise website.

Table 1: Evaluation datasets characteristics

	TE10	TE11	TE-E
number of queries	70	50	50
average number of answer entities	13.6	8.3	12.9
average number of query terms	9.7	9.9	9.3
average number of webpages	2027	2040	1496

In our experiments, the Wikipedia data dump and the Freebase data dump were used to provide clues for entity extraction and resolution. For fair comparison with previous TREC results, we used the Wikipedia database dump at October 17, 2009 and the Freebase data dump at March 20, 2009, which are roughly the same time when the ClueWeb09 corpus was crawled. The DOM tree structure is constructed using the lxml[5] HTML parser. The collection language model used in smoothing was estimated from the ClueWeb09 corpus, using a total of 251,446 webpages from the enterprise websites related to TREC Entity track. We use the standard TREC evaluation program[6] and report three standard retrieval measures, namely, Mean Average Precision (MAP), Precision at ten (P@10), and R-Precision.

We also conduct a tuning process to find a suitable propagation kernel and determine the parameters in our framework. We used the first 20 queries in the TE10 dataset, which were released in 2009 in the TREC Entity track, as the tuning dataset. After the tuning process, Gaussian propagation kernel is adopted for term propagation and Dirichlet prior is adopted as the smoothing method for the entity language model estimation. The tuned parameters are as follows: $\sigma = 300$, $\beta = 0.8$, and $\mu = 200$. If not specified, the following experiments will use this parameter setting.

6.2 Experiment on Entity Retrieval Models

The first set of experiments aims at assessing the performance of our model and conducting comparison to an existing entity retrieval model. We make use of the TE-E dataset to carry out this set of experiments. The performance of different propagation schemas, choices of kernel functions, and the smoothing methods are also investigated in this set of experiments. Structured Propagation Model 1 and 2 refer to our model as described in Section 4.2.1 and Section 4.2.2 respectively. The comparison model, referred to as the Proximity-based Model, denotes the proximity-based entity retrieval model proposed in [20].

Table 2 depicts the performance of different models. By considering the structured content, both of our models have better performance on MAP compared to the Proximity-based Retrieval Model. Structured Propagation Model 2

[4] Available at http://www.se.cuhk.edu.hk/~textmine/?q=dataset/entity-retrieval

[5] Available at http://lxml.de/

[6] Available at http://trec.nist.gov/trec_eval/

further improves the performance compared to Structured Propagation Model 1. For most queries, our proposed model outperforms the comparison model, which does not consider the structured content representation. There are three queries that both models cannot return any correct entities, due to the fact that there are no explicit mentions of the answer entities in the corresponding enterprise website. For all the fifty queries, our named entity recognition component can detect 91.76% answer entities.

Table 2: Performance on the TE-E dataset

	MAP	P@10	R-Prec
Proximity-based Model in [20]	0.3507	0.3680	0.2518
Structured Propagation Model 1	0.3679	0.3380	0.2772
Structured Propagation Model 2	0.3935	0.4200	0.2931

We also explore the behavior of different kernels with different parameters. Figure 7 depicts the effect of kernels using Structured Propagation Model 2. We can see that Gaussian kernel performs slightly better than other kernels. The figure for the Structured Propagation Model 1 is not shown since it attains similar behavior.

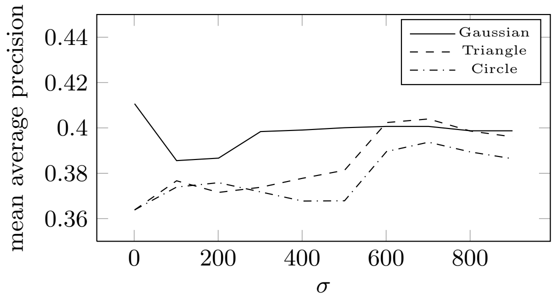


Figure 7: Sensitivity to the parameter σ of different kernels with Dirichlet smoothing ($\mu = 200$)

We also investigate the influence of different smoothing methods and smoothing parameters, as shown in Figure 8 for Structured Propagation Model 2. Dirichlet prior smoothing performs better and it is relative insensitive to the smoothing parameter.

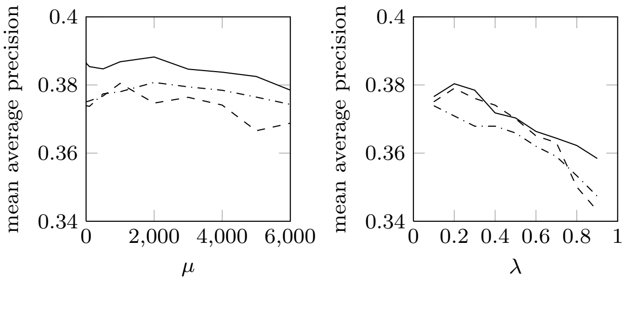


Figure 8: Sensitivity to the smoothing parameter of Dirichlet prior smoothing (left) and Jelinek-Mercer smoothing (right). The legend is the same as that in Figure 7.

For the parameter β used in Equation 7, we find that the model is generally not sensitive to the precise value of β, as long as we set $\beta > 0.5$ to prefer the terms found in the heading blocks.

6.3 Experiments on TREC Entity Datasets

The aim of the second set of experiments is to compare the performance with the previously reported results by the participants in the Related Entity Finding (REF) task of the TREC Entity track. In this set of experiments, we evaluated our Structured Propagation Model 2, the Proximity-based Retrieval Model in [20], and previous TREC Entity track results.

Since the REF task of TREC Entity track evaluates the entity retrieval performance based on entity homepages, we develop a homepage finding algorithm to find the homepages for the retrieved entities, described in the following subsection.

6.3.1 Homepage Finding

Entity homepage usually refers to the official website of an entity, such as the personal webpage for a person entity. We develop a homepage finding algorithm based on a classification method for the retrieved entities. Given a retrieved entity name, we query a search engine using entity name to retrieve a list of relevant pages. For each retrieved page, we generate a feature vector based on the page URL and its content. The feature vectors of the a set of benchmark entities are employed to train an SVM classifier which is used to determine whether a page is the homepage of a particular testing entity.

We exploit three types of features, namely, URL features, page content features, and Wikipedia features. URL features are summarized from the page URL, such as whether the URL fully or partially contains the entity name. Page content features include whether the page title contains the entity name, the frequency of the entity name in the page content, etc. If Wikipedia pages for the entity are retrieved from the search engine, we also extract the Wikipedia-based features, including whether a candidate URL is found in Wikipedia infoboxes, whether a candidate URL is found in Wikipedia external link sections, etc. We observe that the homepages of different entity types have different characteristics so that we train different classifiers for three groups of entities, namely, person, organization, and others.

6.3.2 Result Analysis

The experiment results on the TE11 and TE10 dataset are shown in Table 3 and Table 4 respectively. By explicitly constructing the entity language model, both the proximity-based retrieval model and our structured propagation model outperform the best performance of the TREC participants. By considering the structure of the document, our structured positional entity language model further improves the MAP by four to five percents. This demonstrates the importance of structure information embedded in webpages.

Table 3: Performance on the TE11 dataset

	MAP	P@10	R-Prec
TREC Entity 2011 Best [26]	0.2509	0.3340	0.2908
Proximity-based Model in [20]	0.2943	0.4471	0.3023
Structured Propagation Model 2	0.3457	0.3947	0.3356

Table 4: Performance on the TE10 dataset

	MAP	P@10	R-Prec
TREC Entity 2010 Best [27]	0.2876	0.3936	0.3075
Proximity-based Model in [20]	0.3245	0.3500	0.3212
Structured Propagation Model 2	0.3649	0.4166	0.3542

7. CONCLUSION

We investigate the problem of enterprise entity retrieval, which aims at returning entities as answers for a user query. To tackle this problem, we propose a structured positional entity language model. Combined with our structured content transformation, we can handle entity retrieval in an effective way. Extensive experiments on benchmark datasets demonstrate the effectiveness of our framework.

In the future work, we intend to exploit more sophisticated structured information to improve the entity retrieval. We intend to investigate the incorporation of visual clues to improve the webpage structure transformation.

References

[1] P. Bailey, N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2007 enterprise track. In *The Fourteenth Text REtrieval Conference Proceedings (TREC)*, 2007.

[2] K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.

[3] K. Balog, P. Serdyukov, A. P. D. Vries, P. Thomas, and T. Westerveld. Overview of the trec 2009 entity track. In *The Eighteenth Text REtrieval Conference Proceedings*, 2009.

[4] L. Bing, W. Lam, and Y. Gu. Towards a unified solution: data record region detection and segmentation. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, 2011.

[5] L. Bing, W. Lam, and T.-L. Wong. Wikipedia entity expansion and attribute extraction from the web using semi-supervised learning. In *Proceedings of the sixth ACM international conference on Web search and data mining*, WSDM '13, 2013.

[6] R. Blanco, H. Halpin, D. M. Herzig, P. Mika, J. Pound, and H. S. Thompson. Entity search evaluation over structured web data. In *Proceedings of the 1st International Workshop on Entity-Oriented Search*, EOS '11, 2011.

[7] M. Bron, K. Balog, and M. de Rijke. Ranking related entities: components and analyses. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, 2010.

[8] D. Chakrabarti, R. Kumar, and K. Punera. A graph-theoretic approach to webpage segmentation. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, 2008.

[9] W. B. Croft and J. Lafferty. *Language Modeling for Information Retrieval.* 2003.

[10] Y. Fang, L. Si, Z. Yu, Y. Xian, and Y. Xu. Entity retrieval with hierarchical relevance model, exploiting the structure of tables and learning homepage classifiers. In *TREC*, 2009.

[11] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.

[12] S. Huston and W. B. Croft. Evaluating verbose query processing techniques. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, 2010.

[13] M. Kahng and S.-g. Lee. Exploiting paths for entity search in rdf graphs. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, 2012.

[14] R. Kaptein, P. Serdyukov, A. De Vries, and J. Kamps. Entity ranking using wikipedia as a pivot. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, 2010.

[15] C. Kohlschütter, P. Fankhauser, and W. Nejdl. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, 2010.

[16] C. Kohlschütter and W. Nejdl. A densitometric approach to web page segmentation. In *Proceedings of the 17th ACM conference on Information and knowledge management*, 2008.

[17] M. Lalmas. Uniform representation of content and structure for structured document retrieval. In *20th SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence*, 2000.

[18] Y. Lv and C. Zhai. Positional language models for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, 2009.

[19] G. Miao, J. Tatemura, W.-P. Hsiung, A. Sawires, and L. E. Moser. Extracting data records from the web using tag path clustering. In *Proceedings of the 18th international conference on World wide web*, WWW '09, 2009.

[20] D. Petkova and W. B. Croft. Proximity-based document representation for named entity retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, 2007.

[21] D. Pinto, M. Branstein, R. Coleman, W. B. Croft, M. King, W. Li, and X. Wei. QuASM: a system for question answering using semi-structured data. In *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*, 2002.

[22] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, 1998.

[23] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. In *Proceedings of the 19th international conference on World wide web*, WWW '10, 2010.

[24] A. Tonon, G. Demartini, and P. Cudré-Mauroux. Com ing inverted indices and structured search for ad-hoc object retrieval. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, 2012.

[25] Q. Wang, J. Kamps, G. Ramirez Camps, M. Marx, A. Schuth, M. Theobald, S. Gurajada, and A. Mishra. Overview of the INEX 2012 linked data track. In *CLEF 2012 Evaluation Labs and Workshop*, 2012.

[26] Z. Wang, W. Lv, H. Li, W. Zhou, L. Zhang, X. Mo, L. Zhou, W. Xu, G. Chen, and J. Guo. PRIS at TREC 2011 entity track: Related entity finding and entity list completion. In *TREC*, 2011.

[27] Q. Yang, P. Jiang, C. Zhang, and Z. Niu. Reconstruct logical hierarchical sitemap for related entity finding. In *TREC*, 2010.

[28] X. Yin, W. Tan, and C. Liu. Facto: a fact lookup engine based on web tables. In *Proceedings of the 20th international conference on World wide web*, WWW '11, 2011.

[29] C. Zhai. Statistical language models for information retrieval a critical review. *Found. Trends Inf. Retr.*, 2(3), Mar. 2008.

[30] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2), Apr. 2004.

[31] Y. Zhai and B. Liu. Structured data extraction from the web based on partial tree alignment. *IEEE Trans. on Knowl. and Data Eng.*, 18(12), Dec. 2006.