1st International Workshop on Entity-Oriented Search (EOS)

Balog, K., de Vries, A.P., Serdyukov, P., Wen, J-R.

Proceedings of the ACM SIGIR Workshop
"Entity-Oriented Search (EOS)"
held in conjunction with the
34th Annual International ACM SIGIR Conference
28 July 2011, Beijing, China

ISBN: 978-94-6186-000-2

TU Delft, The Netherlands

contact: balog@acm.org, arjen@acm.org

workshop url: http://research.microsoft.com/sigir-entitysearch/

Preface

Both commercial systems and the research community are displaying an increased interest in returning "objects," "entities," or their properties in response to a users query. While major search engines are capable of recognizing specific types of objects (e.g., locations, events, celebrities), true entity search still has a long way to go.

Entity retrieval is challenging as "objects," unlike documents, are not directly represented and need to be identified and recognized in the mixed space of structured and unstructured Web data. While standard document retrieval methods applied to textual representations of entities do seem to provide reasonable performance, a big open question remains how much influence the entity type should have on the ranking algorithms developed.

This ACM SIGIR Workshop on Entity-Oriented Search (EOS) seeks to uncover the next research frontiers in entity-oriented search. It aims to provide a forum to discuss entity-oriented search, without restriction to any particular data collection, entity type, or user task.

The program committee accepted 11 papers (6 selected for full and 5 selected for short oral presentation) on a wide range of topics, such as learning-to-rank based entity finding and summarization, entity relations mining, spatio-temporal entity search, entity-search in structured web data, and evaluation of entity-oriented search. In addition, the EOS program includes two invited talks from leading search engineers: Paul Ogilvie (LinkedIn) and Andrey Plakhov (Yandex). The workshop also features a panel discussion.

Finally, we would like to thank the ACM and SIGIR for hosting the workshop; Microsoft Research Asia for the computing resources to host the workshop web site; all the authors who submitted papers for the hard work that went into their submissions; the members of our program committee for the thorough reviews in such a short period of time; Paul Ogilvie and Andrey Plakhov for agreeing on giving an invited talk; Yandex for sponsoring the best presentation award.

Krisztian Balog, Arjen P. de Vries,
Pavel Serdyukov, and Ji-Rong Wen
Program Committee Co-Chairs

Table of Contents

Session 3:

EOS Organization

Program Co-Chair:

Krisztian Balog, *(NTNU, Norway)*
Arjen P. de Vries, *(CWI/TU Delft, The Netherlands)*
Pavel Serdyukov, *(Yandex, Russia)*
Ji-Rong Wen, *(Microsoft Research Asia, China)*

Program Committee:

Ioannis (Yanni) Antonellis, *(Nextag, USA)*
Wojciech M. Barczynski, *(SAP Research, Germany)*
Indrajit Bhattacharya, *(Indian Institute of Science, Bangalore, India)*
Roi Blanco, *(Yahoo! Research Barcelona, Spain)*
Paul Buitelaar, *(DERI - National University of Ireland, Galway, Ireland)*
Wray Buntine, *(NICTA Canberra, Australia)*
Jamie Callan, *(Carnegie Mellon University, USA)*
Nick Craswell, *(Microsoft, USA)*
Gianluca Demartini, *(L3S, Germany)*
Lise Getoor, *(University Maryland, USA)*
Harry Halpin, *(University of Edinburgh, UK)*
Michiel Hildebrand, *(VU University Amsterdam, NL)*
Prateek Jain, *(Wright State University, USA)*
Arnd Christian König, *(Microsoft, USA)*
Nick Koudas, *(University of Toronto, Canada)*
David Lewis, *(Independent, USA)*
Jisheng Liang, *(Microsoft Bing, USA)*
Peter Mika, *(Yahoo! Research Barcelona, Spain)*
Marie Francine Moens, *(Katholieke Universiteit Leuven, Belgium)*
Iadh Ounis, *(Univsersity of Glasgow, UK)*
Ralf Schenkel, *(Max-Planck Insitute for Computer Science, Germany)*
Shuming Shi, *(Microsoft Research Asia, China)*
Ian Soboroff, *(NIST, USA)*
Fabian Suchanek, *(INRIA, France)*
Duc Thanh Tran, *(Karlsruhe Institute of Technology, Germany)*
Wouter Weerkamp, *(University of Amsterdam, NL)*
Jianhan Zhu, *(Open University, UK)*

High Performance Clustering for Web Person Name Disambiguation Using Topic Capturing

Zhengzhong Liu Qin Lu Jian Xu

Department of Computing
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
+852-2766-7247

hector.liu@polyu.edu.hk csluqin@comp.polyu.edu.hk csjxu@comp.polyu.edu.hk

ABSTRACT
Searching for named entities is a common task on the web. Among different named entities, person names are among the most frequently searched terms. However, many people can share the same name and the current search engines are not designed to identify a specific entity, or a namesake. One possible solution is to identify a namesake through clustering webpages for different namesakes. In this paper, we propose a clustering algorithm which makes use of topic related information to improve clustering. The system is trained on the WePS2 dataset and tested on both the WePS1 and WePS2 dataset. Experimental results show that our system outperforms all the other systems in the WePS workshops using B-Cubed and Purity based measures. And the performance is also consistent and robust on different datasets. Most important of all, the algorithm is very efficient for web persons disambiguation because it only needs to use data already indexed in search engines. In other words, only local data is used as feature data to avoid query induced web search.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – Clustering; I.2.7 [**Artificial Intelligence**]: Natural Language Processing - Text analysis.

General Terms
Algorithms, Experimentation

Keywords
Web Person Name Disambiguation, Text Clustering

1. INTRODUCTION
When people search for a name, one may intend to find a specific entity, called the namesake associated with that name. Current search engines normally return a large set of documents containing the searched name string based on features such as hit ratio and hyperlinks associated with popular search interest as a whole in the internet rather than features to distinguish different namesakes. For example, when a user submits the query "`George Bush`", the returned documents by a search engine may include mentions such as (1)`George W. Bush (The 43rd President of U.S.` and (2)`George W.H. Bush (The 41st President of U.S.)` that are contained with different ranks. However, the mentions of `George Bush (Professor of Hebrew University, U.S)` would be hard to find as they are being returned as very low ranked files that would normally be missed.

Identifying a specific entity using the current search engine is time consuming because the scale of the returned documents must be large enough to ensure coverage of the different namesakes especially to contain all the documents with namesakes of much less popular persons.

The task of identifying different namesakes through web search is called web persons disambiguation. Recent works on web persons disambiguation have been reported in the WePS workshops [1,2]. The reported systems used different clustering techniques to organize the searched results according to either closed information or open information to disambiguate different namesakes. Some systems have achieved competitive performance by incorporating large amount of open information [7] or by conducting expensive training process [14]. In this paper, we present a novel algorithm based on Hierarchical Agglomerative Clustering(HAC) [16] which makes use of topic information using a so called hit list to make clustering more suitable and effective for web persons disambiguation.

The rest of the paper is organized as follows. Section 2 presents related works. Section 3 describes our algorithm design. Section 4 gives the performance evaluation and Section 5 is the conclusion.

2. RELATED WORKS
Researchers have adopted different assumptions about the data for web persons disambiguation. A common assumption is that each document is associated with only one person and thus a specific namesake [3,4,7]. Others may select some smaller units such as paragraph to represent a single person [18].

Different systems adopted different features for similarity measures. The performance of a system is highly related to the features used. Generally speaking, feature selection can be classified into either local methods or global methods. The local methods select and assign weights to features according to the given collection. Among the local methods, [3] extracts features from the sentences that contain a co-reference of the target name and [15] extracts biographical information from web pages. Many systems in the WePS workshops use simple word tokens from the webpages found in the given corpus [4,7,10]. Some systems use more advanced features like word bigrams [7], phrases [8] or named entities [18]. On the other hand, global methods

EOS, SIGIR 2011 workshop, July 28, 2011, Beijing, China.

incorporate information external to the given dataset such as external corpora or online information for similarity measures. For example, [19] searches for external snippets from Google to directly help in clustering. Other systems may use the external knowledge to tune the weightings of the different features. For example, [7] uses the Google 1T 5-gram data to learn frequencies of bigrams. [14] uses Wikipedia to find phrases in documents.

In addition, because webpages can be noisy, some attempts are made to use only relevant content blocks in webpages. [13] identifies useful content by using predefined templates. [14] uses Gradient Boosting method to learn the content weighting on 1.3 million popular web pages and 400,000 manually annotated query-URL pairs. However, as the types of webpages can vary lot, and new pages are continuously being generated, it may be difficult to capture the most relevant information from the ever changing data types.

In the task summary of the WePS2 workshop, [2] pointed out the importance of selecting the optimal threshold for clustering. The BEST_TOKEN system which knows all the optimal thresholds beforehand, achieve a very good performance with only simple local features. Results show that finding the optimal termination criteria can greatly improve the clustering quality even with only local data. Most prior works on clustering focus on optimization of cluster numbers or threshold for a clustering task. [17] reviewed the set of criteria for cluster number optimization. However, these methods are normally application dependent. The PDDP method used by [6] measures the non-cohesiveness of clusters and split the data in the weakest cohesion point. The Chameleon system [12] adopts a similar method that computes both the inter-connectivity within the cluster and the proximity among clusters. These methods work well on regular clustering problems. However, most of them tend to avoid both singleton and "All-in-One" clusters, whereas in our problem, the sizes of the clusters can vary significantly. For web persons disambiguation, it is important to handle large variations of different cluster sizes.

3. ALGORITHM DESIGN

3.1 Preprocessing

For web persons disambiguation, the data are normally obtained from information already stored in a search engine indexed under the searched name including the webpages, the related snippets and metadata to form a so called local corpus as supplied by WePS workshops. Webpages as documents normally contain a large amount of noise including formatting tags, executable scripts and Meta information. The quality of preprocessing can largely influence the performance of the clustering system.

We use some simple rule-based methods for noise removal. The raw html pages are first parsed by the Beautiful Soup Parser[1] to remove html tags and scripts tags. Only plain text is reserved for clustering. Similar to the measures in [4], for block level tags (such as , <p>), only text blocks with more than 10 words are preserved. The preprocessing module also removes groups of text less than 4 words that are separated by bars or text grouped by <li> tags to eliminate the navigation links in a webpage. After the plaintext is produced, standard sentence segmentation, word tokenization and stemming are applied using the NLTK toolkit[2]. Stop words, punctuations, and numbers less than 4 digits are also removed.

3.2 Feature Selection

It is easy to understand that rich features and extra knowledge beyond the local data collection can improve the system performance. However, getting external information is rather expensive, both on storage and computation. Such information is also sometimes domain specific, which is difficult to obtain in some cases. In practice, we also know that as long as there is information for a specific namesake in the local data, it is often sufficient for human to identify the entity. In this work, we explore methods to make full use of local features, and try to explore the full potential of locally provided information.

Our algorithms use the standard Vector Space Model commonly used by many information retrieval systems. Each document is represented by a vector formed by 7 types of tokens extracted from local data as listed below:

1. **Webpage title**: the title of the webpage is split to single words and added to the feature space.
2. **URL of webpage**: they include both the host name of URL of a webpage and path on the server. We remove the common webpage extensions by a dictionary. Non-meaningful tokens such as pure numbers and punctuations are removed.
3. **Webpage metadata**: only two kinds of metadata in a webpage, "keywords" and "descriptions" are used if they exist. These metadata are highly summarized and informative. Sometimes they contain information that does not appear in the webpage.
4. **Snippet**: Query biased snippets returned by the Yahoo! search engine (included in WePS data) which normally has reference to the name. Snippets are highly summarized fragments of text. Query biased snippets summarize the query context and distant information relevant to the query term [21].
5. **Context Window**: Tokens in a context window within the query name. The window size is selected based on experiments.
6. **Context Sentence**: The whole sentence that contains the query name.
7. **Bag of Words**: Commonly used feature to represent the document. In our system, we simply index all the words in the document as tokens in the feature space.

For a token s, we use the TF.IDF weighting scheme with its weight calculated as:

$$W'_s = \log(tf_s + 1) \times \log(\frac{N_d}{df_s})$$

where tf_s is term frequency of s, N_d is the total number of documents, and df_s is the number of documents in which s is present. Furthermore, we give weight factor to each type of tokens denoted by *WF(s)* because some might be more important than others. Thus, the final normalized weight for each token using cosine co-efficiency is given as:

$$W = \frac{W'_s \times WF(s)}{\sqrt{\sum_{s \in V}(W'_s \times WF)^2}}$$

3.3 Clustering Algorithm

Our clustering algorithm is based on the common Hierarchical Agglomerative Clustering (HAC Algorithm) [16]. In HAC, all documents are treated as singleton clusters initially, and are

[1] http://www.crummy.com/software/BeautifulSoup/

[2] http://www.nltk.org/

referred to as "leaf clusters". In each iteration of HAC, the two most similar clusters will be merged into a larger cluster. The centroid vectors of the two clusters will be merged to a new centroid vector. During the merging of clusters, there are generally two trends in the cluster's centroid:

1. A small set of keywords will be repeatedly matched, thus maintaining a higher weight in the cluster centroid vector.
2. As new tokens are constantly added to the cluster centroid, there are some rare words to be added too with relatively low weight. And these low weight tokens occupy a large portion in the vector.

The HAC algorithm generally works well at the beginning of the clustering. However, when the above two phenomena become more and more prominent, the performance begins to deteriorate. The system may mistakenly merge two large clusters together because of the large number of token hits as shown in **Figure 1**. Also, some newly added tokens may make the topic diverge to a wrong direction. In **Figure 2**, the set of clusters about a journalist may be mistakenly merged together with the cluster about sergeants in a war because of the match on some non-topic words.

Figure 1 Merge of Clusters with low weight terms

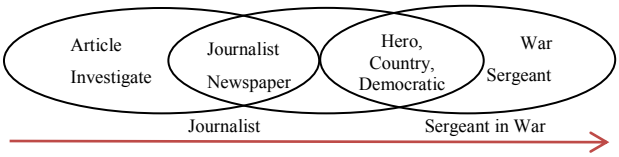


Figure 2 Topic Divergence

3.3.1 Topic Capturing

To solve the problem created by noise data not related to certain topic, we assume that most of the webpages describing the same namesake are likely to have similar topics. For example, documents about a journalist should mainly contain topics about news, journals and so on. Documents about a sergeant, however, should primarily contain topics about war and the like. Based on this assumption, we try to merge document clusters only if they share consistent and useful topics. In other words, our system favors merging clusters that are likely to mention the same topics.

In order to obtain the core topic words, we created a vector called "Hit List" during the clustering process. This vector is maintained for each non-singleton cluster in addition to the centroid vector used in the conventional HAC.

The Hit List of a cluster C is actually the vector containing the shared tokens of the two source clusters that are merged to form cluster C. Each token in the Hit List is associated with a weight, which is the product of its origin normalized weights in the two source clusters. The construction of a Hit List vector can be illustrated using the example in **Table 1**. It is worth noting that the sum of the weights in the Hit List is the dot product of the centroid vector in cluster C_1 and cluster C_2. Because the vectors are already normalized, the dot product represents the cosine similarity of C_1 and C_2. So the Hit List actually records the contribution of each token to the formation of this new cluster C.

Table 1 Example of Hit List construction by merging two clusters of the same size (number of documents in the cluster)

C_1	umass	virginia	research	student
Centroid Vector	0.5	0.1	0.4	0

C_2	umass	virginia	research	student
Centroid Vector	0.5	0.2	0.2	0.1

Merged C	umass	virginia	research	student
Centroid Vector	0.5	0.15	0.3	0.05
Hit List	**0.25**	**0.02**	**0.08**	**0**

* *Hit List for cluster 1 and cluster 2 are omitted.*
** *For demonstration purpose, the centroid vector for the merged cluster is not normalized.*

Table 1 shows that if a token gets a high weight in the Hit List, it should have high weights in its source clusters as well. This means that tokens with high weights in the Hit List are normally important words to documents in the cluster. Accordingly, they can represent the topic of the cluster. As the clustering process continues, the Hit List gradually grows larger. But only the topic words, which are likely to be repeated in many documents, will gain higher weights than general words. So we only keep words with higher weights as topic words. To maintain topic words with significance, we select a fixed threshold that is proportional to the cosine similarity threshold value used in HAC which will be explained in the performance evaluation section.

3.3.2 Similarity Modification Based on Topics

With the help of the Hit List, the similarity measure used in the HAC is slightly modified by a simple scheme: If two clusters that are not likely to be talking about the same topic, their similarity value will be reduced by a penalty value. We consider two cases for the penalty. The first one is when the Hit List of the merged cluster has only few words, which means that the two clusters are generally matched based on the large amount of low weight tokens. The second one is when the Hit List of the merged cluster has only few overlaps with the respective Hit Lists of the source clusters which indicate topic divergence.

For the first case, we use the Vital Match Ratio to handle the problem. Given two clusters, C_1 and C_2 with their feature vector F_1 and F_2, and their corresponding Hit List, H_1 and H_2, and the merged cluster Hit List H_c, The Vital Match Ratio of C, denoted by VMR_c is calculated using the formula:

$$VMR_c = \frac{card(H_c)}{card(C_1) + card(C_2)}$$

where $card(H_c)$, $card(C_1)$, and $card(C_2)$ denote the respective cardinalities of the respective vectors. If VMR_c is less than a threshold, we will give penalty to the similarity by subtracting a penalty score P_c to be determined experimentally.

For the second case, we use the overlapping similarities between H_1 and H_c and between H_2 and H_c to detect the divergence of the clusters to consider penalty. For two vectors V and V', let us use $v(u)$ and $v'(u)$ to denote the weights of a term u in the corresponding vectors. The overlap similarity is then given as:

$$Overlap\_sim(V, V') = \frac{\sum_{u \in V, u \in V'} (v(u) + v'(u))}{\sum_{w \in V} v(w) + \sum_{w' \in V'} v'(w')}$$

If one of the two to be merged clusters is a singleton cluster, which would not have had a Hit List associated with it, we let the

overlap similarity equals to the overlap threshold. Then, we define the divergence value (*DV*) as the harmonic mean of the two values, which is given as:

$$DV = \frac{2}{\frac{1}{Overlap\_sim(H1, Hc)} + \frac{1}{Overlap\_sim(H2, Hc)}}$$

In particular, *DV* value will be 0 if either one of the overlap similarity is 0. If *DV* value is lower than a threshold, we also give a penalty by subtracting a penalty score P_o to be determined experimentally.

4. PERFORMANCE EVALUATION

Our system is developed based on the WePS2 dataset, which contains 30 ambiguous names. Each name is associated with around 150 documents. The total number of documents is 3,444. The names are extracted from different domains, which can help to test the system in real application without any bias. We use both the B-Cubed scores described in [3] and Purity-based scores described in [1] to evaluate our system. The official ranking score for WePS2 is the F-measure (Harmonic Mean) of B-cubed precision and recall, and for WePS1 is the purity based F-measure. The workshops provide two F-measures, one gives equal weighting to precision and recall (α = 0.5), the other give higher weighting to recall (α = 0.2).

As the training data are provided, all the algorithm parameters are determined experimentally based on WePS2 test data. Due to the limit of the paper, we will simply give out the parameters used without giving details of the experiments.

Optimal clustering threshold is difficult to find when the size of the clusters can vary a lot from person to person [2]. In WePS2's local data collection, the minimum number of clusters is 1 and the maximum number of clusters is 56. Also, the number of documents can be from just one document in a cluster to 99 documents in another. The high dimensionality of the document vectors also makes it difficult to model clustering behaviors. In our system, the similarity measurement is Cosine Similarity of two vectors. The algorithm stops if the maximum similarity between clusters is less than the cosine similarity threshold. The threshold is **0.1,** determined experimentally, and also consistent with other systems [9,20]. The weighting factors for different tokens are tuned based on their importance to the clustering, their values are given in **Table 2**. All these parameters are set according to experiments on the WPS2 test data. Experimental data show that metadata and context sentences play more important roles. Snippets and context window are less important perhaps because their information is less coherent.

Table 2 Token Weighting Factors (WF)

Token type	Title	URL	Metadata	Snippets	Context Window	Context Sentence	BOW
WF_t	1	1	2	0.8	0.8	2	1

Threshold values for VMR, the Divergence Value and the corresponding penalty values should be scaled accordingly to the specific application. If these values are higher, then they can have a better control power on topic. The upper bound and lower bound for these values are 1 and 0 respectively. These values should also be proportional to the cosine similarity threshold. Normally, setting the value of penalty scores similar to the cosine similarity threshold will achieve a good performance. The parameters related to the penalties in our system are listed in **Table 3**.

Table 3 Threshold Settings in the Evaluation

VMR Threshold	VMR Penalty (P_c)	DV Threshold	DV Penalty (P_o)
0.02	0.08	0.01	0.1

Table 4 gives the performance evaluation of our system, labeled as HAC_Topic, compared to 2 algorithms within known upper bound and the top 3 performers in the WePS2 evaluation. The BEST-HAC-TOKENS and the BEST-HAC-BIGRAMS systems are the upper bound systems provided by the WePS workshop committee. These two systems have the optimal threshold on each namesake beforehand. The Top1 performer is the PolyUHK system [7] which uses both global and local features. PolyUHK used the Google 1T corpus to learn the weighting of unigrams and bigrams and query Google to find extra information about a person. The Top 2 system UVA_1 [5] uses simple tokens from the html cleaned documents. The Top 3 system ITC-UT-1 [11] uses rich features including named entities, compound nouns and URL links within the local page. **Table 4** shows that our system outperforms all the top 3 teams in both F-0.5 and F-0.2 scores beating systems using both local features and global features. Compared to the systems using local features, ours is at least **5.7%** improvement. This indicates that with a better designed clustering algorithm, the system can effective even if only local features are used. In other words, the clustering algorithm can make full use of local features so the performance can be improved without the loss of run time efficiency through the use of global features.

Table 4 Performance of WePS2 Data on B-Cubed Measures

	F-measures		B-Cubed	
SYSTEMS	α = 0.5	α = 0.2	Pre.	Rec.
BEST-HAC-TOKENS	*0.85*	*0.84*	*0.89*	*0.83*
BEST-HAC-BIGRAMS	*0.85*	*0.83*	*0.91*	*0.81*
Top1:PolyUHK	0.82	0.80	0.87	0.79
Top2:UVA_1	0.81	0.80	0.85	0.80
Top3:ITC-UT_1	0.81	0.76	**0.93**	0.73
HAC_Topic	**0.85**	**0.83**	0.92	**0.82**

It is also worth noting that our HAC_Topic system has shown a similar performance to the BEST-HAC-TOKENS, which is the upper limit of basic token based method. This implies that our method can help find relatively good stopping termination criteria.

Table 5 Performance of WePS2 data on Purity Measures

	F-measures			
SYSTEMS	α= 0.5	α= 0.2	Pur.	Inv_Pur.
BEST-HAC-TOKENS	*0.90*	*0.89*	*0.93*	*0.88*
BEST-HAC-BIGRAMS	*0.90*	*0.87*	*0.94*	*0.86*
Top1:PolyUHK	0.88	0.87	0.91	0.86
Top2:UVA_1	0.87	0.87	0.89	0.87
Top3:ITC-UT_1	0.87	0.83	0.95	0.81
HAC_Topic	**0.90**	**0.89**	**0.94**	**0.88**

Table 5 shows the performance evaluation of the different systems using purity based scores. Again, our system achieves the best result in almost all the performance measures. It even outperforms the best upper bound system BEST-HAC-TOKENS. The consistent high performance in both scoring schemes proves that our algorithm is rather robust which is very important in real applications.

In order to fully validate the effectiveness of our approach, we also tried to apply the algorithm to different datasets. In principle, there are two more datasets to use: WePS1 and WePS3. Even though the WePS3 dataset is relatively large and comprehensive,

the answer set is problematic. It was produced using online crowd sourcing method with little manual verification. So, the set contains incorrect data and also missing data. Thus, comparison to others is not meaningful. Thus, we only used the manually prepared WePS1 dataset for further evaluation and comparison.

The WePS1 dataset contains a test set with 30 names and a training set with 49 names. Our system ran the test set to compare to the other systems in WePS1. The top 3 systems [5,8,18] are all using rich local features such as tokens, URLs, Named Entities and time expressions. **Table 6** shows the performance evaluation based on Purity measures as the official ranking in WePS1 workshop only provided purity-based performance measures, where purity is a measure for precision level and inverse purity is for recall level. As shown in **Table 6,** our system outperforms all the other systems in terms of purity scores. This behavior is expected because the measures are designed to prevent the merging of two clusters referring to different namesakes. Our purity score has a **15.2%** improvement to the best system CU_COMSEM. In terms of the inverse purity score, our system outperforms CU_COMSEM by **3.4%**. The overall improvement in F-score is a significant **10.2%**.

Table 6 Performance of WePS1 data on Purity Measures

	F-measures			
SYSTEMS	α = 0.5	α = 0.2	Pur.	Inv_Pur
Top1:CU_COMSEM	0.78	0.83	0.72	0.88
Top2: IRST-BP	0.75	0.77	0.75	0.80
Top3: PSNUS	0.75	0.78	0.73	0.82
HAC_Topic	**0.86**	**0.89**	**0.83**	**0.91**

We further evaluate the effectiveness of our clustering method compared to the regular HAC method without the use of topic information (labeled as HAC_NoTopic) using WePS2 as training data. **Table 7a** shows the experiment results for evaluation based on the test dataset of WePS1 only and **Table 7b** shows the evaluation using both the test dataset and training dataset of WePS1.

Table 7a Performance of HAC Using WePS1 Test Data

	B-Cubed		Purity		F-Measure	
SYSTEMS	BEP	BER	P	IP	B-Cubed	Purity
HAC_Topic	**0.79**	0.85	**0.83**	0.91	**0.81**	**0.86**
HAC_NoTopic	0.75	0.85	0.67	0.91	0.78	0.76

Table 7b Performance on HAC Using WePS1 Complete Data

	B-Cubed		Purity		F-Measure	
SYSTEMS	BEP	BER	P	IP	B-Cubed	Purity
HAC_Topic	**0.88**	0.82	**0.70**	0.89	**0.84**	**0.76**
HAC_NoTopic	0.84	**0.84**	0.67	**0.90**	0.82	0.75

Table 7a shows that the B-Cubed F-measure is improved by **3.8%**. Furthermore, the Purity based F-measure is improved by **13.1%.** This further shows that the improvement by our system is mainly contributed by the improvement in precision so the resulting data is more reliable. For the full dataset as shown in the **Table 7b**, the corpus is composed of 79 name queries with much more variations in terms of numbers of documents in each collection. It is obvious that when the dataset gets larger, our algorithm has further improvement in all the precision related measures. However, HAC_NoTopic is better in terms of recall related measures. It is certainly understandable that our system has gains in precision at the cost of recall, at least statistically. The important issue is, our algorithm is better for both datasets in terms of F-measure which further confirms that our algorithm can give overall performance improvement compared to the regular HAC. This means that the introduction of the Hit List vector is very effective.

Further investigation shows, however, that our algorithm sometimes can improve both the precision and recall. **Table 8** shows the micro level performance in B-Cubed measures on all query names in the WePS-1 test set. Take the query "Neil_Clark", as an example, the precision is improved by **29%** and the recall is improved by **6%** as well. The overall performance shows our algorithm has significant improvement in precision while keeping the recall at a similar level. This is because we have enough features to distinguish different namesakes without losing any useful information. In other words, the algorithm has the ability to successfully guide clustering to the correct direction.

Table 8 Micro Performance on WePS-1 Test Set (B-Cubed)

	Normal HAC			Using Topic Capturing		
topic	BEP	BER	F-0.5	BEP	BER	F-0.5
Alvin_Cooper	0.85	0.85	0.85	0.85	0.85	0.85
Arthur_Morgan	0.66	0.82	0.73	0.76	0.79	0.77
Chris_Brockett	0.93	0.86	0.89	0.94	0.82	0.88
Dekang_Lin	1.00	0.86	0.93	1.00	0.90	0.95
Frank_Keller	0.87	0.79	0.83	0.86	0.81	0.84
George_Foster	0.72	0.75	0.74	0.71	0.83	0.77
Harry_Hughes	0.86	0.91	0.88	0.88	0.85	0.87
James_Curran	0.71	0.81	0.76	0.66	0.83	0.73
James_Davidson	0.86	0.91	0.88	0.84	0.92	0.88
James_Hamilton	0.63	0.72	0.68	0.75	0.74	0.75
James_Morehead	0.59	0.88	0.71	0.60	0.86	0.71
Jerry_Hobbs	0.92	0.77	0.84	0.92	0.77	0.84
John_Nelson	0.76	0.89	0.82	0.80	0.88	0.84
Jonathan_Brooks	0.87	0.91	0.89	0.87	0.91	0.89
Jude_Brown	0.68	0.85	0.75	0.78	0.83	0.80
Karen_Peterson	0.72	0.98	0.83	0.76	0.98	0.86
Leon_Barrett	0.91	0.78	0.84	0.93	0.78	0.85
Marcy_Jackson	0.73	0.79	0.76	0.71	0.79	0.75
Mark_Johnson	0.58	0.92	0.71	0.77	0.91	0.83
Martha_Edwards	0.42	0.93	0.58	0.51	0.93	0.66
Neil_Clark	**0.68**	**0.81**	**0.74**	**0.97**	**0.87**	**0.92**
Patrick_Killen	0.70	0.81	0.75	0.87	0.76	0.81
Robert_Moore	0.70	0.70	0.70	0.87	0.67	0.76
Sharon_Goldwater	0.98	0.82	0.89	0.99	0.80	0.88
Stephan_Johnson	0.88	0.85	0.87	0.92	0.83	0.88
Stephen_Clark	0.87	0.88	0.88	0.83	0.87	0.85
Thomas_Fraser	0.42	0.88	0.57	0.46	0.85	0.60
Thomas_Kirk	0.57	0.88	0.70	0.71	0.88	0.79
Violet_Howard	0.55	0.96	0.70	0.61	0.96	0.75
William_Dickson	0.59	0.89	0.71	0.55	0.90	0.68
Average	0.74	0.85	0.78	0.79	0.85	0.81

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed an effective HAC algorithm using additional topic information for web persons disambiguation. The experimental results show that our proposed clustering algorithm can achieve very good performance over the conventional methods. The key to the high performance of this algorithm is that it can effectively reduce the over merging of namesakes in clustering especially apparent when the cluster sizes can vary a lot. The disambiguation power of the method is thus improved significantly. As a result, features required in the algorithm are less demanding than other algorithms used for web persons disambiguation. In fact, our algorithm only uses simple local features from the training data which is readily available in most of the current search engines. This means that the processing time and the storage requirement in our system is much less demanding. This is rather important in practice where timely feedback to user queries is essential. As the features selected in our algorithm are already common indexed terms by modern search engines, the method can be developed easily on any existing search engine.

However, there is still room for improvement. Firstly, the parameter settings are done based on WePS2 data. In principle, the parameters are sensitive to the data as they are threshold based algorithms. This is particularly true if the application is used in different domains. Adjusting parameters for a given domain is important for the success for the system. The algorithm will be more robust if the number of parameters can be reduced. Possible reduction can be investigated over the parameters used for the topic related penalties. We can also further investigate methods to reduce the dimension of the feature vectors used in the algorithm. We can also study the possibility of using relatively cheap global data sources for the training phase. Some global data can be achieved offline without knowing user queries, such as pre-compiled corpus for learning term frequencies. The use of such offline data will not greatly affect the query processing time. Other possible directions for feature enrichment include using biographical information from the webpages and utilizing more semantic information such as synonyms information.

It is important to point out that clustering is only the first step in web persons disambiguation. To give a full picture for the users, the system should also be able to label the resulting clusters with the corresponding attributes of the namesakes.

6. ACKNOWLEDGMENTS

The project is partially supported by China Soong Ching Ling Foundation and Poly Project(COMP): RPVW.

7. REFERENCES

[1] Artiles, J., Gonzalo, J., & Sekine, S. 2007. The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. *Proceedings of Semeval*, (June), 64-69.

[2] Artiles, J., Gonzalo, J., & Sekine, S. 2009. Weps 2 evaluation campaign: overview of the web people search clustering task. *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*.

[3] Bagga, A., & Baldwin, B. 1998. Entity-based cross-document coreferencing using the vector space model. *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, 79–85. Association for Computational Linguistics.

[4] Balog, K., Azzopardi, L., & Rijke, M. de. 2005. Resolving person names in web people search. *Weaving services and people on the World Wide Web*, 301–323.

[5] Balog, K., He, J., Hofmann, K., Jijkoun, V., Monz, C., Tsagkias, M., et al. 2009. The University of Amsterdam at WePS2. *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*.

[6] Boley, D. 1998. Principal direction divisive partitioning. *Data mining and knowledge discovery*, *2*(4), 325–344. Springer.

[7] Chen, Y., Lee, S. Y. M., & Huang, C. R.2009. Polyuhk: A robust information extraction system for web personal names. *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*.

[8] Chen, Y., & Martin, J. 2007. Cu-comsem: Exploring rich features for unsupervised web personal name disambiguation. *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, 125-128.

[9] Elmacioglu, E., Tan, Y. F., Yan, S., Kan, M. Y., & Lee, D. 2007. PSNUS: Web people name disambiguation by simple clustering with rich features. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, 268–271.

[10] Han, X., & Zhao, J. 2009. CASIANED: Web Personal Name Disambiguation Based on Professional Categorization. *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2-5.

[11] Ikeda, M., Ono, S., Sato, I., Yoshida, M., & Nakagawa, H.2009. Person Name Disambiguation on the Web by Two-Stage Clustering. *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*.

[12] Karypis, G., & Kumar, V.1999. Chameleon: hierarchical clustering using dynamic modeling. *Computer*, *32*(8), 68-75. doi: 10.1109/2.781637.

[13] Lin, S.-hua, & Ho, J. M. 2002. Discovering informative content blocks from Web documents. *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (p. 588–593). ACM.

[14] Long, C., & Shi, L. 2010. Web person name disambiguation by relevance weighting of extended feature sets. *Third Web People Search Evaluation Forum (WePS-3), CLEF* (Vol. 2010, pp. 1-13).

[15] Mann, G. S., & Yarowsky, David. 2003. Unsupervised personal name disambiguation. *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 -*, 33-40. Morristown, NJ, USA: Association for Computational Linguistics. doi: 10.3115/1119176.1119181.

[16] Manning, D. C., Raghavan, P., & Schutze, H. 2008. Hierarchical Clustering. *Introduction to Information Retrieval*. *Cambridge University Press*, *New York*, *2008*, 377 - 401.

[17] Milligan, G. W., & Cooper, M. C.1985. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*(2), 159–179. Springer.

[18] Popescu, O., & Magnini, B.2007. Irst-bp: Web people search using name entities. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, (June), 195-198.

[19] Rao, D., Garera, N., & Yarowsky, D.2007. JHU1: an unsupervised approach to person name disambiguation using web snippets. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, 2-5.

[20] Smirnova, K. A. E., & Trousse, B. 2010. Using web graph structure for person name disambiguation. *Third Web People Search Evaluation Forum (WePS-3), CLEF* (Vol. 2010).

[21] Tombros, A. and Sanderson, M. Advantages of query biased summaries in information retrieval. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, ACM (1998), 2–10.

Extracting Dish Names from Chinese Blog Reviews Using Suffix Arrays and a Multi-Modal CRF Model

Richard Tzong-Han Tsai and Chun-Hui Chou
Department of Computer Science and Engineering
Yuan Ze University, Taiwan
thtsai@saturn.yzu.edu.tw s976057@mail.yzu.edu.tw

ABSTRACT

Online blog reviews are a useful source of information for creating restaurant directories. An important step in mining restaurant review blogs is extracting dish names. In this paper, we propose a novel two-phase method for extracting dish names from Chinese language blog reviews. In the first phase, we identify candidates using an unsupervised suffix-array-based approach. In the second, we validate the extracted candidates using a CRF-based approach that can simultaneously model the correlation between sets of super/substring candidates. In addition to affix features, our model also considers quotation marks, style (font/color/hyperlink) and image proximity. Our experimental results show that adding these extra features significantly improves affix-only baseline performance by as much as 8.05%. Furthermore, our CRF-based validation outperforms our best ME-based validation configuration by 4.6%, bringing system F-Score to a high of 86.28%.

1. INTRODUCTION

One of the richest sources of online restaurant review is blogs. Not only are blog reviews plentiful, but they often offer fresher or more down-to-earth perspectives than professional reviews. More importantly, blog reviews are more varied so small, unheard-of, or unusual restaurants or even food stands are more likely to receive reviews. Furthermore, because most blogs allow commenting, readers can respond to reviews by adding their own opinions or posting response reviews on their own blogs. The blogging model has been so successful that many newspapers are now adopting it for their online sites (e.g. New York Times Blogs).

Obviously, online blog reviews are a useful source of information for creating restaurant directories, which can be used for entertainment location-based services (LBS) accessible through mobile devices. Since the main focus of a restaurant review is usually the individual dishes served by the restaurant, an important step in mining restaurant review blogs is extracting dish names.

SIGIR 2011 2011 Beijing, China

In this paper we set out to extract dish names from restaurant reviews on Chinese language blogs. Extracting Chinese dish names faces the same problems as extracting any Chinese named entity (people, places, products, etc.), namely, boundary ambiguity due to the lack of spaces between Chinese words and confusion with normal words or phrases. Dish names pose a few specific problems, however. First, there are no comprehensive dictionaries or lists of dish names. Secondly, dish names tend to exhibit more irregular formation than general named entities. For example, a Chinese personal name is usually three characters long beginning with a family name followed by two characters. A dish name, on the other hand, is as likely to exhibit a regular pattern, such as "stir-fried cabbage", as it is to be completely abstract, such as 佛跳牆 (Buddha Jumps Over the Wall). Another recent trend in dish naming is to take a traditional idiom or phrase and substitute food-name homonyms for its characters. For example, the phrase, 忙個沒完 (mang ge mei wan), to be busy without end, becomes 芒個莓玩 (same pronunciation), which translates literally as "strawberry mango ball", a name for a strawberry and mango covered shaved ice dessert. Adding to these irregularities, blog authors tend to use abbreviations or variations of dish names in their reviews. These factors increase the difficulty of extracting dish names from restaurant review pages.

In order to extract dish names without reliable patterns or lists, we have developed a suffix-array-based method to identify character sequences that appears more than twice as dish name candidates. This method is faster and more efficient than the traditional PAT-Tree method. To validate that the character sequences are real dish names, we have also designed a multi-modal candidate validation model using conditional random fields (CRF) [4]. In addition to the prefix/suffix features employed by traditional approaches, our model also considers candidates' quotation marks, style (font/color/hyperlink) and whether or not neighboring sentences contain pictures. Because a single blog review is not sufficient to identify some dish names, we aggregate multiple reviews of individual restaurants.

2. RELATED WORK

There are two main approaches to identifying names or keywords in Chinese text: unsupervised and supervised.

The unsupervised approach does not require annotated corpora. Sproat and Shih [9] developed a purely statistical method that utilizes the mutual information shared by two characters. The main limitation of this method is that it can only deal with two-character words. Chien [3] proposed

a PAT-tree-based method that extracts candidate words by identifying the mutual information of two overlapping patterns with a significance function. The advantage of this method is has high recall and no length limit on extracted words. Unfortunately, constructing a PAT tree is computationally expensive in terms of space and time. In addition, this method also extracts a significant number of non-words.

On the other hand, the supervised approach trains its model on an annotated corpus. Wu et al. [10] constructed a Chinese named entity recognition system by using conditional random fields and character n-gram models. Zhao and Liu [11] proposed a hierarchical-hidden Markov-model-based approach to identify product names from Chinese texts. The benefit of this approach is it takes a candidate's contextual information into its decision. Therefore, this approach strikes a very good balance between precision and recall. However, annotating a large corpus character by character requires a significant amount of manual effort. According to the results of SIGHAN 2006 [5], the recall of out-of-vocabulary words is far lower than that of in-vocabulary words.

3. METHODS

In this paper, we describe a two-phase method for extracting Chinese dish names which combines the high recall of the unsupervised approach and the high precision of the supervised approach.

In the first phase, we identify candidates with an unsupervised approach based on suffix arrays, which are effective for extracting all possible terms, not only in-vocabulary terms. Unlike traditional approaches that extract words from single documents, we use all review documents corresponding to a restaurant because we believe that using multiple documents will increase the frequency and style information for each dish name.

Since the resulting extracted terms are not necessary dish names, in the second phase, we employ a candidate validation model which uses a supervised approach based on CRF. Our approach only requires a list of dish names instead of a corpus of fully annotated sentences, reducing annotation efforts significantly.

Figure 1 shows the main modules and the data flow of our system. We can see that the reviews of a restaurant are first processed in the candidate identification phase before candidate validation. These two phases are explained in the following two sections.

3.1 Candidate Identification

3.1.1 Identifying Dish Name Candidates with Suffix Arrays

We first preprocess all review pages by converting them from HTML format to free text. Then, for each restaurant, we construct a suffix array from its preprocessed reviews to extract all frequent strings that appear more than twice. The suffix array algorithm works as follows:

Suppose S is an input string, I is the set of non-delimiter positions in S, and SA is an array that stores all initial positions of all suffixes of S. At the start, $SA[i]$ is initialized as i. Then, we sort SA according to the strings referred to by each entry in SA in lexicographic order.

Take $S =$ "菲力牛排好吃。菲力牛排分兩種" for example; $I =$ [1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13]. Figure 2 illustrates the construction of SA for S. (a) is before sorting, while (b) is after. We can see that four term candidates: 力牛排, 牛排, 排, 菲力牛排 are found. Single character candidate is discarded in our system.

3.1.2 Candidate Filtering

Suffix-array-based candidate identification extracts many irrelevant non-dish terms. According to our observations, these terms can be detected by checking their character constructions or part-of-speech (POS) constructions. For example, if a term ends in "的/de", it is unlikely to be a dish name since the "的" suffix indicates the word is an adjective. Likewise, a term whose POS sequence ends in a preposition is unlikely to be a dish name. We employ the CKIP POS tagger [2] to label 100 dish names and 500 non-dish names. Then, we compile a list of patterns for non-dish names. On average, approximately 48% of the extracted terms are removed.

3.2 Candidate Validation Model

3.2.1 Formulation

The likelihood that a set of super/substring terms, such as 漢堡, 士漢堡, and 吉士漢堡, are dish names is highly correlated. That is, the label of a term is highly correlated with that of its substrings/superstrings. In order to model this correlation, we label a set of super/substrings simultaneously. We have observed, for example, that the longest proper substring of a dish name term is often not a dish name, such as "士漢堡" and "吉士漢堡" above. Therefore, we represent all super/substring terms in one restaurant's re-views as a sequence of terms from the shortest to longest. Each term corresponds to a state, whose value (D or N) represents whether or not the term is a dish name. Figure 3 depicts our formulation of the above example.

We employ the conditional random fields model because it is good at global sequence tagging in many natural language processing tasks [6][8] and its transition feature models the neighboring labels' correlation well. For each candidate that appears in more than one sequence, we use the label with the highest probability as its label.

3.2.2 Conditional Random Fields

Conditional random fields (CRFs) are undi-rected graphical models trained to maximize a conditional probability [4]. A linear-chain CRF with parameters $\Lambda = \{\lambda_1, \lambda_2, \ldots\}$ defines a conditional probability for a state sequence $\mathbf{y} = y_1 \ldots y_T$ given an input sequence $\mathbf{x} = x_1 \ldots x_T$ to be

$$P_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\mathbf{x}} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right) \quad (1)$$

where $Z_\mathbf{x}$ is the normalization that makes the probability of all state sequences sum to one; $f_k(y_{t-1}, y_t, \mathbf{x}, t)$ is often a binary-valued feature function and λ_k is its weight. The feature functions can measure any aspect of a state transition, $y_{t-1} \rightarrow y_t$, and the entire observation sequence, x, centered at the current time step, t. For example, one feature function might have value 1 when $y_{t-1} = D$ (dish name), $y_t = D$, and Above-Image(x_t) = True. Large positive values for λ_k indicate a preference for such an event; large negative values make the event unlikely.

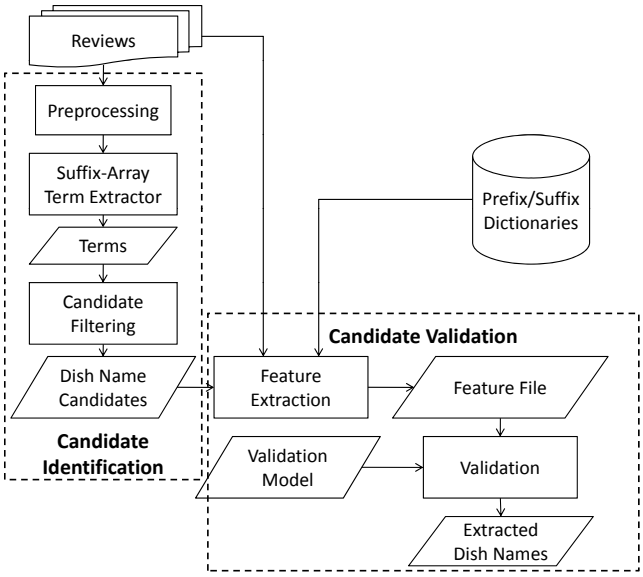


Figure 1: System data flow

S =菲力牛排好吃。菲力牛排分兩種

(Fillet steak is delicious. There are two types of fillet steak)

(a) before sorting	(b) after sorting
SA[1]=菲力牛排好吃。菲力牛排分兩種	SA[1]=力牛排分兩種
SA[2]=力牛排好吃。菲力牛排分兩種	SA[2]=力牛排好吃。菲力牛排分兩種
SA[3]=牛排好吃。菲力牛排分兩種	SA[3]=分兩種
SA[4]=排好吃。菲力牛排分兩種	SA[4]=牛排分兩種
SA[5]=好吃。菲力牛排分兩種	SA[5]=牛排好吃。菲力牛排分兩種
SA[6]=吃。菲力牛排分兩種	SA[6]=吃。菲力牛排分兩種
SA[7]=菲力牛排分兩種	SA[7]=好吃。菲力牛排分兩種
SA[8]=力牛排分兩種	SA[8]=兩種
SA[9]=牛排分兩種	SA[9] =排分兩種
SA[10]=排分兩種	SA[10]=排好吃。菲力牛排分兩種
SA[11] =分兩種	SA[11]=菲力牛排分兩種
SA[12] =兩種	SA[12]=菲力牛排好吃。菲力牛排分兩種
SA[13] =種	SA[13]=種

Figure 2: (a) SA before sorting. SA[i] represents the semi-infinite string starting at position SA[i] and ending at the end of the corpus. (b) SA after sorting. The strings represented by SA are now in lexicographic order. We can see that four term candidates: 力牛排, 牛排, 排, 菲力牛排 are found.

The most probable label sequence for $\mathbf{x}$,

$$\mathbf{y}^* = \arg\max_y P_\Lambda(\mathbf{y}|\mathbf{x}),$$

can be efficiently determined using the Viterbi algorithm [7].

The parameters can be estimated by max-imizing the conditional probability of a set of label sequences, each given their corres-ponding input sequences. The log-likelihood of a training set $\{(x_i, y_i) : i = 1, \ldots, M\}$ is written as:

$$L_\Lambda = \sum_i \log P_\Lambda(\mathbf{y}_i|\mathbf{x}_i)$$
$$= \sum_i (\sum_{t=1}^{T} \sum_k \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t) - \log Z_{x_i})$$

To optimize the parameters in CRFs, we uses a quasi-Newton gradient-climber BFGS [8].

3.2.3 Features

For each candidate term, we extract the following four features from the reviews of the restaurant in which it appears. The first is our baseline feature, and the following three are extra features.

Affix Features.

Traditional Chinese dish names are named regularly. For example, some are composed of a cooking method followed by a material, such as fried rice or steamed pork. Therefore, we analyze 3,052 dish names and extract the 1,036 most frequent 1-character and 2-character prefixes (e.g., 炸/fried, 紅燒/stewed, etc.) and suffixes (e.g., 麵/noodles, 湯/soup, and 漢堡/hamburger, etc.). These extracted prefixes and suffixes are stored in two dictionaries. If a term's prefix/suffix matches an entry in the prefix/suffix dictionary its Prefix/Suffix feature is enabled.

Quotation Feature.

If the term is enclosed in （ ）, 「」, “”, (), [], or ‘’, this feature is enabled; otherwise, it is disaabled.

Image Proximity Feature.

According to our observations, images in reviews are often of food items, and in such cases, the corresponding dish name is almost always mentioned before or after the image. That is, a candidate term appearing in a sentence on either side of an image is more likely to be a dish name. The image proximity feature is designed to exploit this correlation. If an image appears before or after a term, then either the Above- or Below-Image feature is enabled.

Style Features.

We have observed that bloggers often mark up dish names in a different style from surrounding text, such as using bold font or different colors. We have design three different style features to describe this information: Bold, Colored, and Other. If the marked up range does not exactly match the boundaries of a candidate term, we use the following formula to calculate the maximum marked-up length for a term t.

Assume T is the set of all appearances of t in the reviews of a restaurant, r. t_i is the ith appearance in T, s_i is the marked-up sentence covering t_i

$$MaxMarkedUpLength(t, r) = \arg\max_{t_i \in T} \frac{|t|}{|s_i|}$$

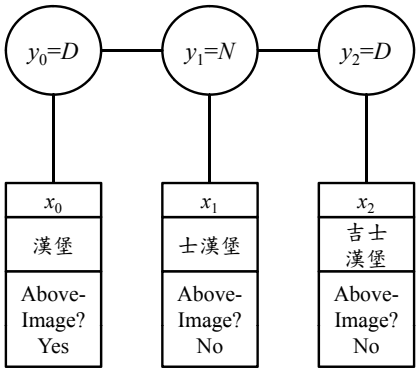


Figure 3: Representation of super/substring candidates “漢堡”, “士漢堡”, “吉士漢堡” in our CRF model

The feature is set to 2 if *MaxMarkedupLength* is greater than our threshold. The feature is set to 1 when *MaxMarkedupLength* is greater than 0 and less than our threshold. Otherwise, the feature is set to 0. In our system, the threshold is set to 0.1.

4. EXPERIMENTS

We conducted experiments with Chinese blog pages to empirically evaluate the performance of our dish name extraction models, focusing particularly on the validity of the methods discussed in the previous sections.

4.1 Dataset

We compiled a corpus of user-written blog reviews from `ipeen.com.tw`, a Taiwanese entertainment portal with social networking and blogging features. Our corpus consists of 1077 restaurant reviews for 294 restaurants (from 529 authors). We used our candidate identification model to list candidate dish names that appeared at least twice in these reviews. Our annotators then verified the list. In total, we annotated 2,272 dish names for the above-mentioned 294 restaurants.

4.2 Experiment Design

We designed two experiments. Experiment 1 measures the effects of using different feature combinations on system performance. To exclude the effects brought by using our new formulation, we use the maximum entropy (ME) model [1] because ME labels all candidates individually and its feature cannot model the correlation between labels of super/substring candidates. Experiment 2 compares the performance of ME-based and CRF-based approaches using best feature combination from Experiment 1. We train all system configurations on 30 randomly chosen training sets (g_1,. . ., g_{30}) taken from our main dataset. Each set contains reviews for 150 restaurants. After the training process, we test all systems on thirty 30-restaurant test sets (trained on g_1 for use with test set 1, and trained on g_2 for use with test set 2, etc.). We then sum the scores for all thirty test sets, and calculate the averages for performance comparison.

4.3 Evaluation Metrics

The results are given as F-scores and defined as $F = (2PR)/(P + R)$, where P denotes the precision and R denotes the recall. The formulas for calculating precision and

Table 1: Performance of each configuration of ME-based system. A, Q, I, and S denote Affix, Quotation, Image Proximity, and Style features, respectively

	A	Q	I	S	$\hat{P}$ (%)	$\hat{R}$ (%)	$\hat{F}$ (%)	$\hat{S}_F$ (%)	ΔF (%)	t	$F > F_{Baseline}$? (t >1.67?)
$ME_{Baseline}$	√				72.24	75.36	73.63	2.89	-	-	-
ME_{A+Q}	√	√			74.22	74.93	74.31	2.88	0.68	0.62	N
ME_{A+I}	√		√		77.98	80.95	79.35	5.24	5.72	8.17	Y
ME_{A+S}	√			√	76.68	80.55	78.46	2.52	4.83	6.49	Y
ME_{ALL}	√	√	√	√	81.94	81.61	81.68	5.59	8.05	7.01	Y

Table 2: Performance comparison of ME-based and CRF-based systems

	A	Q	I	S	$\hat{P}$ (%)	$\hat{R}$ (%)	$\hat{F}$ (%)	$\hat{S}_F$ (%)	F_{CRF}-F_{ME} (%)	t	$F_{CRF} > F_{ME}$? (t >1.67?)
ME_{ALL}	√	√	√	√	81.94	81.61	81.68	5.59	-	-	-
CRF_{ALL}	√	√	√	√	84.06	88.72	86.28	2.61	4.60	4.08	Y

recall are as follows:

$$P = \frac{\text{\# of correct recognized dish names}}{\text{\# of recognized dish names}}$$

$$R = \frac{\text{\# of correct recognized dish names}}{\text{\# of true dish names}}$$

4.4 Results

Table 1 and 2 shows all the configurations and the summarized results. The latter are reported as the mean precision ($\hat{P}$), recall ($\hat{R}$), and F-score ($\hat{F}$) of thirty datasets. We examine the detailed statistics of all the experiments in Tables 1, and 2. In the tables, as well as $\hat{P}$, $\hat{R}$, and $\hat{F}$, we also list the sample standard deviation of the F-score ($\hat{S}_F$) for each configuration. We apply a two-sample t test to examine whether one configuration is better than the other with statistical significance. The null hypothesis, which states that there is no difference between the two configurations, is given by

$$H_0 : \mu_A = \mu_B$$

where μ_A is the true mean F-score of configuration A, μ_B is the mean of configuration B, and the alternative hypothesis is

$$H_1 : \mu_A > \mu_B$$

A two-sample t-test is applied since we assume the samples are independent. As the number of samples is large and the samples' standard deviations are known, the following two-sample t-statistic is appropriate in this case:

$$t = \frac{\bar{x}_A - \bar{x}_B}{\sqrt{\frac{S_A^2}{n_A} + \frac{S_B^2}{n_B}}}$$

If the resulting t score is equal to or less than 1.67 with a degree of freedom of 29 and a statistical significance level of 95%, the null hypothesis is accepted; otherwise it is rejected.

Experiment 1

Table 1 shows the results of Experiment 1. Here, **A**, **Q**, **I**, and **S** stand for Affix features, Quotation feature, Image Proximity features, and Style features, respectively. The baseline configuration, which is denoted as $ME_{Baseline}$, employs only **A**. The configurations created by adding **Q**, **I**, and **S** to **A** one by one are denoted as ME_{A+Q}, ME_{A+I}, and ME_{A+S}, respectively. We use ΔF to denote the F-score difference between each configuration's F-score and $ME_{Baseline}$'s. We can see that the F-scores increase by 0.68%, 5.72%, and 4.83%, respectively. ME_{A+I} and ME_{A+S} perform significantly better than $ME_{Baseline}$. From this comparison, we can also see that the Image Proximity features are the most effective features, while the Quotation feature is the least effective.

Lastly, we add **Q**, **I**, and **S** to **A** and denote this configuration as ME_{ALL}. It outperforms $ME_{Baseline}$ by 8.05%, a statistically significant difference. From these results, we have demonstrated that the best configuration is ME_{ALL}. We employ this feature set in Experiment 2.

Experiment 2

Table 2 shows the results of Experiment 2. We found that CRF_{ALL} outperforms ME_{ALL} by 4.60%, a statistically significant difference. The CRF-based system had both better recall and precision.

5. DISCUSSION

5.1 Contribution of Extra Features

After examining the features of the dish names that were extracted by ME_{ALL} but not by $ME_{Baseline}$ in the thirty experiments, we found that 36% of them do not match any entry in the prefix/suffix dictionary (e.g., 摩摩喳喳/Bubur Cha-Cha, a well-known Nyonya dessert). The other 64% of them match an entry in only one of the dictionaries. That is, these names do not match known/common affixes that indicate they are dish names. Fortunately, according to our analysis, 33% names have Quotation features enabled, 83% names have Image Proximity features enabled, and 57% have Style features enabled. Obviously, these names can be extracted by employing extra features. From this analysis, we can also see that the Quotation feature is the least frequently enabled. This explains why the quotation feature is the least

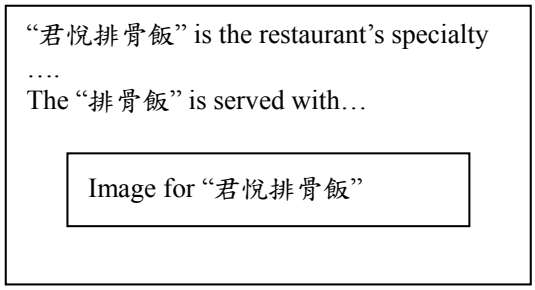


Figure 4: Visualization of two super/substring candidates in a review (排骨飯's Above-Image feature is enabled, while 君悅排骨飯's is not)

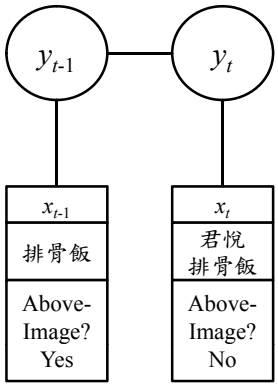


Figure 5: Representation of super/substring candidates "排骨飯" and "君悅排骨飯" in our CRF model

effective extra feature.

5.2 Contribution of CRF

To analyze CRF's performance, we looked at the dish names that were recognized by CRF_{ALL} but not by ME_{ALL}. We found that these names usually super/substrings. In ME_{ALL}, since super/substring candidates are predicted separately, if some of these candidates' extra features are disabled, they may be not recognized. However, since CRF models the correlation between labels of super/substring candidates, it is more likely to recognize these. In the example in Figures 4 and 5, we can see that the Above-Image feature of "君悅排骨飯/Emperor Pork Chop with Rice" is disabled, while that of "排骨飯/ Pork Chop with Rice", which actually refers to the Emperor Pork Chop, is enabled. The CRF feature that describes this correlation is "$y_{t-1} = D$, $y_t = D$ and Above-Image(x_t) = True". In other words, if a candidate is a dish name and it is right above an image, its superstring candidate is also a dish name. Because this feature has a higher weight than other transition features associated with Above-Image, such as "$y_{t-1} = N$, $y_t = N$ and Above-Image(x_t) = True", CRF can successfully detect 君悅排骨飯.

5.3 Cases CRF Performs Poor

CRF errors can be divided into two types: false positives and false negatives. According to our observations, false positives can be further divided into three types: restaurant names, food styles (e.g., Thai food), and conjunctions, which are candidates containing multiple dish names (e.g., 漢堡和薯條/hamburger and French fries). False positives are often extracted because, just like dish names, they are keywords of restaurant reviews and have extra features enabled.

On the other hand, we found that most false negatives are candidates that lack neither affix information nor extra information. Deeper natural language processing may be necessary to correctly identify many of these false negative candidates.

6. CONCLUSION

To take advantage of the high recall of the unsupervised approach while maintaining the high precision of the supervised approach, we have proposed a novel two-phase method in this study. In the first phase, we identify candidates using an unsupervised suffix-array-based approach, which can process data more quickly and efficiently than traditional approaches.

In the second phase, we validate the candidates extracted in the first phase. To save annotation effort, we only annotate dish names in each restaurant's reviews rather than annotating every sentence. Since there is strong correlation between the labels of super/substring candidates, we propose a novel CRF-based approach that can model each set of super/substring candidates as a sequence and label them simultaneously. In addition to the prefix/suffix features employed by traditional approaches, our model also considers candidates' quotation marks, style (font/color/hyperlink) and image proximity. Because a single blog review is not sufficient to identify some dish names, we aggregate multiple reviews of individual restaurants.

Our experimental results show that with the extra features added our system significantly outperforms the baseline system using only affix features. Using the CRF-based model which labels super/substring candidates simultaneously, our system's F-score is further improved to 86.28%, which is significantly better than the best ME-based system (81.68%) in our experiment.

7. REFERENCES

[1] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, 1996.

[2] L.-p. Chang and K.-j. Chen. The ckip part-of-speech tagging system for modern chinese texts. In *Proceedings of the 1995 International Conference on Computer Processing of Oriental Languages*, pages 172–175, Hawaii, 1995.

[3] L.-F. Chien. Pat-tree-based adaptive keyphrase extraction for intelligent chinese information retrieval. *Information Processing and Management*, 35(4):501–521, 1999.

[4] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289. Morgan Kaufmann Publishers Inc., 2001.

[5] G.-A. Levow. The third international chinese language processing bakeoff: word segmentation and named entity recognition. In *Proceedings of the 5th SIGHAN Workshop on Chinese Language Processing*, pages

108–117, Sydney, Australia, 2006.

[6] H. Nanba, H. Taguma, T. Ozaki, D. Kobayashi, A. Ishino, and T. Takezawa. Automatic compilation of travel information from automatically identified travel blogs. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 205–208, Suntec, Singapore, 2009. Association for Computational Linguistics.

[7] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Readings in speech recognition*, pages 267–296. Morgan Kaufmann Publishers Inc., 1990.

[8] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 134–141, Edmonton, Canada, 2003. Association for Computational Linguistics.

[9] R. Sproat and C. Shih. A statistical method for finding word boundaries in chinese text. *Computer Processing of Chinese and Oriental Languages*, 4(4):336–351, 1990.

[10] C.-W. Wu, S.-Y. Jan, R. T.-H. Tsai, and W.-L. Hsu. On using ensemble methods for chinese named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 142–145, Sydney, Australia, 2006.

[11] J. Zhao and F. Liu. Product named entity recognition in chinese text. *Language Resources and Evaluation*, 42(2):197–217, 2008.

LADS: Rapid Development of a Learning-To-Rank Based Related Entity Finding System using Open Advancement

Bo Lin, Kevin Dela Rosa, Rushin Shah, Nitin Agarwal
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213 USA
{bolin,kdelaros,rnshah,nitina}@cs.cmu.edu

ABSTRACT

In this paper, we present our system called LADS, tailored to work on the TREC Entity Track Task of Related Entity Finding. The LADS system consists of four key components: document retrieval, entity extraction, feature extraction and entity ranking. We adopt the open advancement framework for the rapid development and use a learning-to-rank approach to rank candidate entities. We also experiment with various commercial and academic NLP tools. In our final experiments with the TREC 2010 dataset, our system achieves the fourth rank compared to the fifteen teams who participated in TREC 2010.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *information filtering retrieval models, search process, selection process*.

General Terms

Algorithms, Design, Experimentation

Keywords

Named Entity Recognition, Learning to Rank, Information Retrieval

1. INTRODUCTION

We focus on the task of related entity finding, as defined by the TREC 2011 Entity Track. This is essentially an entity-oriented web search task. As input, we are given "topics", which consist of the name of an input entity, the type of the target entity, and a narrative describing the nature of the relationship between entities. There are four different target entity types: organization, location, person, and product. The goal of the task is to find homepages, represented by their ClueWeb09 id [2], for target entities for a given topic. This is a challenging task due to several factors; one being that web search engines are optimized to find documents and not entities, and another being that it is harder to precisely convey the semantics of a relation to a search engine. Our approach to this task is notable for two reasons: Firstly, in order to achieve fast iteration and the best possible results, we used an open advancement approach and designed a modular architecture that allows us to easily plug in different NLP components, including off-the-shelf commercial ones. Secondly, we made use of a rich feature set to rank candidate entities, and experimented with a Learning to Rank (LETOR) approach to combine these features in a sophisticated manner. We were able to create a system that achieved results close to the best published numbers; with the advantage of requiring very little development time.

EOS, SIGIR 2011 Workshop, July 28, 2011, Beijing, China.

2. RELATED WORK

The TREC evaluations have been the pre-eminent venue for advancements in the field of information retrieval. The TREC Entity track was added a few years ago with the aim of encouraging entity oriented research. The Related Entity Finding task is currently the main task in this track. The Clueweb 09 corpus that we used was compiled with the aim of allowing researchers to develop and test our system on a complete snapshot of the web, as it existed in 2009. There has also been a lot of prior work in developing the open advancement software engineering approach, particularly in the field of question answering.

3. DATASET AND EVALUATION

3.1 Dataset

We use topics from the 2010 REF data set. This data set consists of 50 topics, with a heavy skew towards the "organization" target entity type (60%). Of these topics, 47 were ultimately judged, with homepages being pooled (depth 20) from all of the participants in the 2010 evaluation [8].

Homepages are retrieved from the Clueweb 09 corpus, and acceptable homepages are ranked as either being "primary", or pages that are devoted and in control of the entity (i.e. http://www.metallica.com/), and "related", or pages that are devoted but not in control of the entity (i.e. http://www.last.fm/music/Metallica). For the 2010 offering of the task, Wikipedia pages are disallowed as homepages. An example query from the 2010 data set, in XML format is in figure 1:

Figure 1: Example TREC 2010 Entity Track Topic

```
<query>
  <num>25</num>
  <entity_name>U.S. Supreme Court</entity_name>
  <entity_URL>clueweb09-en0012-87-
19363</entity_URL>
  <target_entity>organization</target_entity>
```

<narrative> **From what schools did the Supreme Court justices**
receive their undergraduate degrees?
</narrative>
</query>

3.2 Evaluation

We use NDCG@R as our primary metric to evaluate our results. We set the gain to 1 for relevant pages, and 3 for primary pages. In addition, we also report the number of primary and relevant pages retrieved, and a few other standard information retrieval metrics such as R-precision (Rprec), mean average precision (MAP), and precision at 10 (P@10).

4. ARCHITECTURE AND COMPONENTS

4.1 Architecture

As mentioned earlier, one of our key objectives in designing our system was to have the ability to develop parts independently and combine them quickly, and ensure quick experimentation and frequent integration of components. To this end, we designed our system to consist of interchangeable components with common interfaces, and a specific pipeline set dynamically via configuration files. Our pipeline contains the following abstract base components:

- Query Analysis: Performs an initial analysis of the input topic queries
- Document Retrieval: Searches for documents based on input from the query analysis
- Entity Extractor: Extracts candidate entities from the retrieved document pool
- Feature Extractor: Extracts features for each candidate entity
- Entity Ranker: Ranks entities based on their feature values
- Homepage Retrieval: Finds homepages for the ranked entity list

Each of these components is described in detail in the following subsection.

4.2 Components

4.2.1 Query Analysis

We use OpenEphyra [5] to extract keywords from query narratives and supplied this information to subsequent components in the pipeline. We do not perform more sophisticated processing with query analysis because of its limited value in increasing the overall accuracy of our system.

4.2.2 Document Retrieval

We experiment with using web-based search as well as Indri-based local search for document retrieval. For both cases, we vary parameters such as the number of documents retrieved, and the use of full query narratives versus only keywords.

4.2.2.1 Web Search

We wrap two different commercial web search engines, including Yahoo BOSS [7] (build your own search service), and Microsoft Bing [4]. For our final system, we use Yahoo BOSS since it was more reliable and flexible with respect to behavior and API rate limitations; though anecdotally, the results from both services are quite similar.

4.2.2.2 Local Search

We use a JNI-based Indri interface [6] that searches the ClueWeb09 category B corpus (a set of 50 million high-quality English language webpages, including all of Wikipedia) [2]. We do not use any of Indri's sophisticated query operators, as we are interested in treating the web and local search engines as black box search engines with minimal changes to the queries provided as input.

4.2.3 Entity Extraction

We adopted and experimented with state-of-art named entity extraction systems, including academic [11] and commercial products [1], to extract candidate entities from the retrieved documents for feature extraction and entity ranking. The input for the entity extraction component is HTML web page and the desired output is the set of candidate entities extracted from the web page. We implemented and investigated various heuristics to improve such webpage-based entity extraction.

4.2.3.1 Stanford NER

Stanford Named Entity Recognizer (Stanford NER) is a commonly used academic tool for named entity recognition known for the accuracy of its results. It incorporates conditional random field-based classifier to label the named entities in the text. Yet Stanford NER is originally designed and trained with plain text without HTML decorations. Thus we made several efforts to adapt the Stanford NER for our purpose of extracting entities from HTML web pages.

Firstly, we incorporated a standard HTML text extractor, JSoup [3], to perform text extraction from HTML web pages. It mainly removes the HTML tags, formatting information and meta-data. Secondly, we varied the system by incorporating structure information in the HTML web pages. The intuition is that simply removing the HTML tags in the web pages eliminates the structured information associated with them. Yet such information might be useful for considering the candidacy of the entities. By applying Stanford NER directly on the extracted text, we failed to use the structure information. Thus we decided to use the readily available structure information in the HTML web pages - the HTML elements (e.g. "body", "title", "anchor text", "table" etc.). We hypothesize that since many queries are targeted towards a list of entities thus the candidate entities are more likely to appear in the lists or tables inside the HTML web pages. Thirdly, we improved the system by using the heuristics of injecting delimiters to the HTML web page. In our early error analysis, we found that the removal of the HTML tags in the HTML text created this problem. HTML web pages sometimes use the HTML elements as text delimiters; however, our HTML text extractor removes all these delimiters to produce the plain text for Stanford NER to work with. Since Stanford NER is trained on News texts with numerous text delimiters such as punctuation marks, it fails to separate the neighboring entities in our case. We then tested our heuristics to inject artificial delimiters to the HTML web pages. While there are many artificial delimiters we could use, we chose the punctuation marks "," since it has less ambiguity than "." (which can be considered as a mark for initial other than a sentence break) or "-" (which can be used to connect two words). We inserted the punctuation marks "," before every occurrence of HTML tag-closing statement "</" in the HTML text before the HTML text extraction. Fourthly, we attempted and experimented with several heuristics to filter ill-formed entities in order to improve the system. We designed and tried three different filtering

heuristics to remove the ill-formed candidate entities. The first filtering heuristics consider the number of characters in the surface form of the token, discard the entity if it is shorter than a threshold value. The second filtering heuristics consider the number of tokens in the entity, discard the entity if the number of tokens is less than a minimum value or larger than a maximum value. We tried a few different values for the thresholds in these two filters and eventually picked 4 for the minimum number of characters in the entity surface form, 1 for the minimum number of tokens and 5 for maximum number of tokens in the entity. The third filtering heuristics consider the tokens in the entity, if they consist of only function words, the entity is discarded. The function word list is obtained from the open sourced question answering tool OpenEphyra [5].

4.2.3.2 AlchemyAPI NER

AlchemyAPI Named Entity Extractor (NEE) is one of the state-of-art commercial tools for named entity extraction. We conducted an extensive survey of both state-of-art academic and commercial tools and found that AlchemyAPI NEE is one of the best among them and provides the best balance of extraction accuracy and processing speed.

AlchemyAPI NEE is capable of identifying people, companies, organizations, cities, geographic features, and other typed entities within HTML text, plain text, or web-based content. It supports entity disambiguation which links the extracted entity to its corresponding entry in external database including DBpedia, Freebase, OpenCyc etc.. It also supports entity type identification for 34 different entity types such as automobile, city, facility.

We implemented AlchemyAPI in our system with manually created mapping between its 34 entity types to the 4 entity types in TREC entity track.

4.2.4 Feature Extraction

We investigated a wide set of features that we thought might be good indicators of how relevant a candidate entity is to the query.

- **Frequency Based Features**

These features are based on the number of times a candidate entity shows up in search results. In particular, we count

- F1 - The total number of occurrences of a candidate entity in all the search results
- F2 - The number of unique search results that an entity occurs in

- **Density Based Features**

These features consider the quality of web pages that a candidate entity occurs in, as judged by the number of other candidate entities in these web pages. Specifically, for a particular candidate entity E_C we count:

- D1 - The number of all entities in a webpage that E_C occurs in (summed over all the search results that E_C occurs in).
- D2 - The number of unique entities in a webpage that E_C occurs in (summed over all the search results that E_C occurs in).

- **Proximity Based Features**

These features are based on the concept of ranking entities according to how closely they occur to query keywords in the retrieved web pages. We use two such features:

- P: The minimum distance between any query keyword and a candidate entity E_C in a webpage (averaged across all the search results that E_C occurs in).
- P_N : Product of cumulative proximity between keywords and entities in retrieved documents and number of such documents. Formally,

$$Score(c) = numDocuments(c) \times \sum_{\cdot} f(dist(k,c),\delta)$$

where *dist* is the proximity and *f* is any aggregate function such as summation or maximum.

- **Semantic Similarity Based Features**

We evaluate semantic similarity between words using the Jiang Conrath metric [12], which considers the WordNet distance between two strings. We apply this metric to count the following specific features:

- S1: Similarity between the query narrative and the snippet of a search result (Averaged across all search results for an entity E_C).
- S2: Similarity between the keywords in the query and the type of the candidate entity
- S3: Similarity between the keywords in the query and the Freebase description of the candidate entity

- **Average Rank of Web Pages**

We consider the average rank of the web pages that a candidate entity E_C occurs in, with the intuition that entities that appear in higher ranked pages on average are more likely to be relevant. We refer to this feature as AvgRank subsequently.

4.2.5 Entity Ranking

We noticed that almost all the systems that have participated in the TREC entity track in previous years used some sort of linear weighting scheme to combine different features in order to rank entities, and one of our motivations was to use a learning to rank (LETOR) approach for this task and see whether such an approach would outperform traditional linear weighting schemes. We therefore developed an SVM model to rank candidate entities.

A significant problem in using an SVM for this task is finding gold standard data to train such a model, because the relevance judgments from the previous years that are available to us are in the form of binary values that indicate whether an entity is relevant or not, while our desired output from the SVM ranker is a real-valued score for each candidate entity. To get around this problem, we train an SVM using this binary-formatted data, but while testing on new instances, we have it output a score between 0 and 1, indicating how likely an entity is to be relevant.

More precisely, we train on a subset of the queries, and during

Table 3: Effect of various improvements on Stanford NER

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
Stanford	31	108	0.0681	0.096	0.0551	0.078
Stanford-Injection	**50**	**160**	**0.1**	**0.1306**	**0.0792**	**0.1093**
Stanford-Minmax	30	108	0.0702	0.0949	0.0551	0.078
Stanford-Minmax-Keyword	27	108	0.0766	0.0995	0.0582	0.0781

training we check if a candidate entity for a particular query is present in the relevance judgment file and marked as a primary or relevant entity. If so, this constitutes a positive instance for the SVM, and if not, a negative instance. We encounter many more negative instances than positive ones, so we keep a fixed ratio of negative instances for each positive instance found, and throw away the rest of the negative instances. We then perform a grid search to find the optimal parameters for training and subsequently train our SVM. During testing, we have it output a relevance score for each candidate entity and classify entities according to this score. We use the LibSVM [9] library to implement SVMs.

4.2.6 Homepage Retrieval

We started with a naive homepage retrieval strategist that returns the first page from a search engine which is not a Wikipedia page, since the Wikipedia domain is explicitly disallowed for the TREC Related Entity Finding task. We picked this strategy because search engines like Google & Yahoo do a very good job at returning homepages, or at least very devoted and informative pages, for entities & topics, and in fact this turned out to be a surprisingly difficult strategy to beat. One change that was a significant improvement over this naïve strategy was using a list of "blacklisted domains" for use in filtering irrelevant homepages. The domains we filtered were primarily content farms, such as "answers.com", "about.com" and "mahalo.com". Content farms are typically web pages that that have high concentrations of text that are specifically designed to be ranked highly by web search engine algorithms and are thus unlikely to be considered a relevant or primary page with respect to the TREC Entity task. In other words, instead of rejecting Wikipedia pages from consideration, we reject any page from a blacklisted domain.

5. EXPERIMENTS AND ANALYSIS

In this section we describe the results of each of our component experiments, and provide an analysis of the experimental results.

5.1 Document Retrieval

We experimented with using both full query narratives and keywords only, and found that both strategies yielded the same number of related pages. Using the full narrative had a slight edge in retrieving more primaries, but NDCG@R was higher for the key word strategy. This is due to the fact that the key word search returned slightly better quality documents, which were ultimately beneficial in entity ranking.

We also used both web-based (Yahoo, Bing) and local (Indri) search engines, and found that web search clearly out performs the Indri search on all counts, since it gets significantly better quality pages which resulted in more related and primary pages.

Table 1: Performance with full query narrative versus using keywords only

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
Full Narrative	40	183	0.1021	0.1176	0.072	0.0861
Key Word	40	176	0.1106	0.1182	0.0662	0.0909

Table 2: Performance with web search (Yahoo) versus Indri for document retrieval

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
Indri	37	154	0.1	0.0978	0.0513	0.0855
Web Search	40	176	0.1106	0.1182	0.0662	0.0909

5.2 Entity Extraction

We used heuristics and filtering techniques to improve Stanford NER on web-page entity extraction. The baseline run "Stanford" is the same as in previous experiment by directly applying Stanford NER on the text extracted from the entire web-page. Other runs show the result from using Stanford NER together with the delimiter injection heuristics ("Stanford-Injection"); and using Stanford NER together with the entity length filtering heuristics which filters the entity with too many / few tokens or too few characters ("Stanford-Minmax"); and using Stanford NER with the entity length filtering heuristics and keyword filtering heuristics which filters the entity containing only function words from the OpenEphyra's function word list ("Stanford-Minmax-Keyword"). The following table shows the measures of performance from these runs.

As observed, delimiter injection outperforms others by 30% in all the measures, which confirms our early error analysis that there are around 10%-20% ill-formed entities are related to the removal of delimiters in HTML text extraction. The other filtering techniques also helped improve P@10, NDCG@R and MAP relatively by 5%-10%. Eventually, we selected "Stanford-Injection" and "Stanford-Minmax-Keyword" as the two competitive settings for Stanford NER in our final runs.

We also compared performance of our system with the use of Stanford NER versus AlchemyAPI NEE. The performances of the two runs differ in different metrics. In terms of P@10 and Rprec, the "Best-Alchemy" outperforms the "Best-Stanford" relatively by 20%-30%. However, in NDCG@R and MAP, "Best-Stanford" performs slightly better than "Best-Alchemy".

Also with respect to NDCG@R, the best run using Stanford NER with delimiter injection is the best run of our entire system. We further investigated this problem and discovered that "Best-Stanford" produced more relevant results (as of "# Relevant") and more total correct results (as the sum of "# Relevant" and "# Primary") in terms of number of home pages. But "Best-Stanford" suffered in the primary results it returned (as of "# Primary") thus in the measures of P@10 and Rprec. We hypothesized that by using Stanford NER, we obtained more entities than using AlchemyAPI NEE. However, the entities returned by AlchemyAPI NEE is more accurate than that by Stanford NER, which might have significantly affected the down-stream homepage retrieval component.

Table 4: Performance with Stanford NER versus AlchemyAPI

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
Stanford NER	**48**	159	0.1064	**0.1352**	**0.0815**	0.1163
AlchemyAPI	36	**168**	**0.1362**	0.1339	0.08	**0.1302**

5.3 Feature Extraction

We tested the relative performance of our extracted features, by using each feature individually to rank the candidate entities.

We notice a number of interesting trends in these results. For the frequency based and density based features, we find that counting all occurrences of an entity in a document instead of unique occurrences of the entity (i.e. F1 and D1 instead of F2 and D2)

leads to slightly better performance. Also, the frequency based features perform better than the equivalent density based ones.

The proximity feature P_N is the best performing feature on almost all metrics; notably, it does better than the pure proximity feature P and all the semantic similarity features S1, S2 and S3. Surprisingly, AvgRank (the average rank of web pages), which is quite a naive feature, is the next best performing feature. Amongst the semantic similarity based features, S1 (similarity between query narrative and candidate webpage snippet) performs reasonably well, but S2 (similarity between the given target type and the candidate entity type) and S3 (similarity between given target type and candidate entity freebase type) both give very poor performance. We believe that while type similarity information might still be valuable, the way in which we were interpreting semantic similarity might be flawed and hence cause such poor results.

5.4 Entity Ranking

We tried different variants of our SVM Ranker: We trained one variant V1 using the default parameters supplied by the LibSVM [9] tool, and another one V2 using tuned parameters that we calculated using a grid search. We also used a feature selection tool and trained another variant V3 that used only the features suggested by this tool, namely F1, D2 and P.

As expected, we find that the model that uses the tuned parameters performs better than the one that uses default parameters. Surprisingly however, we find that neither of the models performs as well as just using the proximity feature P_N alone (results using individual features for ranking are detailed in the previous section). This contradicts our initial assumption that the use of SVMs for ranking entities would produce better performance than using any individual feature alone. We also find that the model V3 that uses the subset of features suggested by the feature selection tool doesn't perform as well as the tuned model V2 that uses all the features in the refined feature set. This suggests that feature selection is not very effective for this problem.

As a baseline, we also trained a linear combination model that simply combined all of our features (except S2 and S3, which performed quite poorly on their own), with equal weights. Surprisingly, this model performs slightly better than the tuned SVM model using the same feature set. This negates our assumption that SVMs would be a more effective way to combine different types of features for ranking than linear combination.

However, none of the above models perform as well as simply using the proximity feature P_N alone to rank candidate entities (results using individual features for ranking are detailed in the previous section). We tried running an exhaustive parameter sweep, since this might have yielded a set of weights that performed better than the feature P_N alone, but the total number of possible models was exponential in the number of features, and since each model required re-running at least the homepage retrieval component of our system, we quickly encountered API rate limits (even when we cached previously found web pages). Ultimately, this parameter sweep proved intractable. It is an interesting point to note that even though linear combination rankers are conceptually much simpler than SVMs, optimizing their parameters requires more resources for our system than optimizing the parameters of our SVM model.

Table 5: Performance of the system with individual features used to rank candidate entities

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
F1 (Frequency)	**34**	147	0.1064	0.1146	0.0665	0.1034
F2 (Frequency)	**34**	147	0.1064	0.114	0.0658	0.1032
D1 (Density)	32	142	0.0957	0.1071	0.0607	0.0855
D2 (Density)	32	141	0.0957	0.1063	0.0604	0.0861
P (Proximity)	33	147	0.0957	0.112	0.0628	0.0847
S1 (Semantic)	31	145	0.1106	0.1121	0.0646	0.1065
S2 (Semantic)	33	138	0.0255	0.0392	0.0212	0.036
S3 (Semantic)	32	136	0.0321	0.0401	0.0228	0.0341
AvgRank	33	143	0.1	0.1164	0.062	0.1085
C (Combination)	32	**150**	**0.1128**	**0.1226**	**0.0696**	**0.1123**

Table 6: Performance with various SVM models for ranking candidate entities

	# Relevant	# Primary	P@10	NDCG@	MAP	Rprec
V1 (un-tuned)	30	141	0.0915	0.1025	0.0583	0.0964
V2 (tuned)	33	141	0.1	0.1073	0.0651	0.0924
V3 (feature-selection)	32	141	0.0934	0.1041	0.0603	0.0945
Linear Combination	33	145	0.1125	0.1187	0.0665	0.1111

Table 7: Improvement due to perfect homepage retrieval

	NDCG@R	MAP	Rprec
Regular H/P Retrieval	0.1182	0.0662	0.0909
Perfect H/P Retrieval	**0.1976**	**0.1386**	**0.141**

5.5 Homepage Retrieval

The evaluations and experiments of LADS system shown in previous sections are inherently imperfect because the only homepages considered relevant according to the relevance judgment file from the 2010 TREC entity track are based on the pooled results submitted by the participants to the track. Results other than the ones existed in the relevance judgment file will not be considered. Our system often produces the correct entities and obviously relevant homepages but failed to match the ones included in the track's judgment file. Since we are more interested in the problem of discovering related entities, and not necessarily their homepages, we conduct an experiment where we assume we have a perfect homepage retrieval system which gives exactly the same homepages as those in the relevance judgment file. If an entity retrieved by the system match exactly some entity in the relevance judgment file, we consider it relevant regardless of its homepage. We can see from Table 7 that this results in a dramatic improvement in the performance of our system. We use this modification as part of our system when making our final runs, the results of which are reported in the following section.

6. FINAL RESULTS

We picked the following configurations to be our final runs, and their results in comparison with the best and median automatic results from the TREC 2010 Entity task evaluation are reported in Table 8.

- **Run 1**: Baseline: Yahoo, 40 documents, AlchemyAPI, Ranking using PN, no blacklist filtering

- **Run 2**: Run 1 with SVM ranker instead of proximity feature P_N
- **Run 3**: Run 1 with the blacklist homepage filter applied
- **Run 4**: Run 1 with Stanford NER instead of Alchemy NEE.

Our results are consistently above the median value, with the best results being for Run 1, and would place us around the 4th place among all the teams.

7. DISCUSSION

Table 8: Best final runs of our system

	NDCG@R	MAP	Rprec
2010 Best System	0.3694	0.2726	0.3075
2010 4th Best System	0.1696	0.0953	0.1453
Run 1	**0.2036**	**0.1406**	**0.1687**
Run 2	0.1754	0.1096	0.1322
Run 3	0.1885	0.1272	0.1276
Run 4	0.1926	0.1188	0.1393

Some of the main advantages of our system include the modular and pluggable software engineering approach we used to implement it, our usage of commercial IR and NLP tools, the diverse feature set we implemented to rank candidate entities, and our use of a learning-to-rank (LETOR) approach in the form of our SVM Ranker. Our architecture allowed us to iterate quickly and independently even though there are a lot of moving parts in our system, and it also allows for easy improvements to our system in the future. We didn't achieve the improvements that we expected from using SVMs to rank candidate entities, but the feature set we used to rank candidate entities is diverse enough that we are optimistic of obtaining better performance in the future as we work on better ways to combine these features, including potentially more sophisticated LETOR approaches than our current one. We also found that while commercial NLP tools may not achieve state-of-the-art performance compared to academic systems, they are often more robust and have certain other advantages. For example, AlchemyAPI, the commercial NER system that we used for entity extraction, has a more fine-grained type system than academic NER systems such as Stanford NER. Consequently, we adapted a number of such tools within our system.

8. CONCLUSION

In terms of future work, we plan to investigate ways to better incorporate the source entity in our system, using structured queries for Indri search, using additional sources of information for homepage retrieval besides just surface forms of entities, and looking for alternative ways to frame entity ranking as a learning-to-rank task. We also intend to open-source our system (after submission to TREC), and we hope that it will be valuable to other research groups wishing to work in the field of Related Entity Finding, or related areas such as Question Answering.

9. ACKNOWLEDGEMENTS

We would like to thank Professors Robert Frederking, Anatole Gershman, Jamie Callan and Eric Nyberg at Carnegie Mellon University for their support and guidance.

10. REFERENCES

[1] Alchemy API. [Online]. Available: http://www.alchemyapi.com

[2] ClueWeb09. [Online]. Available: http://boston.lti.cs.cmu.edu/clueweb09/wiki/tiki-index.php

[3] JSoup: Java HTML Parser. [Online]. Available: http://jsoup.org

[4] Microsoft Bing API. [Online]. Available: http://www.bing.com/developers/

[5] OpenEphyra Question Answering System. [Online]. Available: http://www.ephyra.info/

[6] The Lemur Project. [Online]. Available: http://www.lemurproject.org/

[7] Yahoo BOSS API. [Online]. Available: http://developer.yahoo.com/search/boss/

[8] K. Balog, P. Serdyukov, and A.P. de Vries. Overview of the TREC 2010 Entity Track. In *Proceedings of TREC 2010*, 2010.

[9] C. Chang and C. Lin. LIBSVM : A Library for Support Vector Machines. 2001. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[10] Y. Fang, L. Si, N. Somasundaram, Z. Yu and Y. Xian. Purdue at TREC 2010 Entity Track. *In Proceedings of TREC 2010*, 2010.

[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 of the Association for Computational Linguistics*, 2005.

[12] J. J. Jiang and D. W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Computing Research Repository – CORR*, 1997.

[13] P. Jiang, Q. Yang, C. Zhang, and Z. Niu. Beijing Institute of Technology at TREC 2010: Notebook Paper. In *Proceedings of TREC 2010*, 2010.

[14] D. Wang, Q. Wu, H. Chen, and J. Niu. A Multiple-Stage Framework for Related Entity Finding: FDWIM at TREC 2010 Entity Track. In *Proceedings of TREC 2010*, 2010.

Finding Support Documents with a Logistic Regression Approach

Qi Li, Daqing He
School of Information Sciences
University of Pittsburgh
{qil14, dah44}@pitt.edu

ABSTRACT
Entity retrieval finds the relevant results for a user's information needs at a finer unit called "entity". To retrieve such entity, people usually first locate a small set of support documents which contain answer entities, and then further detect the answer entities in this set. In the literature, people view the support documents as relevant documents, and their findings as a conventional document retrieval problem. In this paper, we will state that finding support documents and that of relevant documents, although sounds similar, have important differences. Further, we propose a logistic regression approach to find support documents. Our experiment results show that the logistic regression method performs significantly better than a baseline system that treat the support document finding as a conventional document retrieval problem.

Categories and Subject Descriptors:
H.3 Information Storage and Retrieval;

General Terms
Algorithm, Features, Experimentation

Keywords
Entity retrieval, Learning to Rank, Logistic Regression, Evaluation

1 SUPPORT DOCUMENTS IN ENTITY RETRIEVAL
Most search engines are still document retrieval systems, which they return a ranked list of documents as the results for satisfying a user's information needs. However, in many cases, people would like to know the exact answers to a query, like "what is the product of Medimmune Inc.", instead of a document containing the answers. This scenario enforces the study of sub-document retrieval, which includes passage retrieval, question answering, and entity retrieval. We are interested in entity retrieval, where answer entities are the required return results.

We believe that there are important differences between conventional document retrieval and entity retrieval. Although document retrieval engines analyze hyperlinks and anchor texts, they still assume the "bag of words" model in the document units. Moreover, the relevance judgments are also on the document level so that a document would be judged relevant regardless of how small the relevant piece of text is in relation to the rest of the document. Entity retrieval, on the other hand, assumes that the answer entities have specific relationship requirements expressed in the query, and the retrieval of them, therefore, has to consider not only mining the documents for the entities, but also identifying the right relationship among the entities

Entity retrieval, like conventional information retrieval tasks, requires effectively and efficiently finding of answer entities from a large corpus of unstructured (e.g., web pages) or a semi-structured documents (e.g., Wikipedia pages). In order to achieve such goal, we believe that an entity retrieval framework designed based on the principle of separating word-independent factors and word-dependent factors should be established. At the word-independent stage, the "bag-of-words" representation can be applied to efficiently find the support documents. At the word-dependent stage, deeper text analyses will be performed on the small set of support documents to extract answer entities.

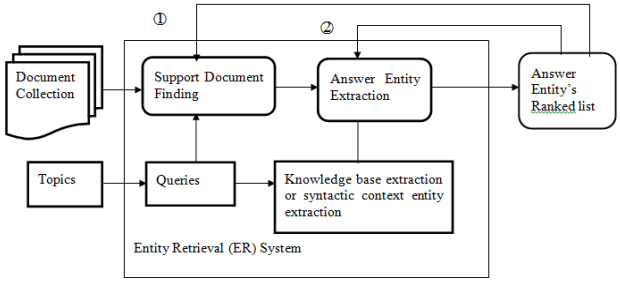


Figure 1 A Two-Layer Retrieval and Extraction Probability Model (TREPM)

This entity retrieval framework is called Two-layer Retrieval and Extraction Probability Model (TREPM) [1]. It decomposes entity retrieval into two layers: support document finding and answer entity extraction (see Figure 1). The inputs to the framework include documents (HTML pages or plain texts) and users' information needs (the search task description with the required entity type). The outputs are a ranked list of entities. The support document finding layer has the tasks of retrieving a small set of the support documents, each of which potentially contains the answer entities. The answer entity extraction layer extracts the answer entities from the support documents. The support document finding layer only deals with word-independency factors and considers the term co-occurrences (i.e. the independence of the terms in the document) in order to efficiently find support documents. All the semantic related analyses, therefore, should be postponed into answer entity extraction. Support documents should contain as many answer entities as possible. The reason that the number of support documents should be as small as possible is because answer entity extraction is a complicated and time-consuming task. The smaller the support document set, the more efficient the entity retrieval process is.

EOS, SIGIR 2011 workshop, July 28, Beijing, China

With the probability model, we describe the entity retrieval problem in the TREPM model as $p(e \mid q,t)$, that is, the probability of an entity e to be the answer entity given the query q and the target entity type t. If we consider all documents, then the formula changes to:

$$p(e \mid q,t) = \sum_{d} p(e,d \mid q,t) = \sum_{d} p(d \mid q,t)\, p(e \mid d,q,t)$$

However, it is not efficient to consider extracting answer entities from all the documents in the collection. Therefore, we choose support documents $d_{\mathrm{sup}port}$ to estimate this probability.

$$p(e \mid q,t) \approx \sum_{d_{\mathrm{sup}port}} p(d_{\mathrm{sup}port} \mid q,t)\, p(e \mid d_{\mathrm{sup}port}, q,t)$$

The first part, in fact, is the support document finding, and the second part is the answer entity extraction from the support documents.

We believe that support documents are different from the relevant documents in conventional document retrieval because support documents, not only have to have relevant information as relevant documents do, they also need to meet two extra criteria. First, we prefer short support documents rather than longer ones. With shorter document length, relevant information often is highly condensed, which makes answer entity extraction relatively easier. Second, we prefer support documents containing as many potential answer entities as possible so that the extraction of answer entities would be more efficient. . For example, if we treat "Products of Medimmune, Inc." as a document retrieval problem, the expected answer lists ranked in the decreasing relevant scores are http://www.ethyol.com/, http://www.Flumist.com/, and http://www.medimmune.com/about_us_products.aspx because the first two documents directly talks about one specific product whereas the third page contains miscellaneous information about the whole products of the company. However, in support document finding task, we would prefer the third page over the first two pages because our entity extraction layer can obtain all product information from that paper rather than mining multiple pages. Now the question is how support documents can be found?

In this study, we propose a logistic regression method for the support document finding. That is, with a model learned from the training data sets, the system can predict the probability of a document to be the support document. This method can combine pre-defined features from various considerations for the ranking task. The machine learning method—logistic regression—is applied to predict the probability. Experiments on the TREC Entity Extraction Task (2009 and 2010) data sets evaluate whether the logistic regression method can improve support document finding.

2 RELATED WORKS

The main goal of this work is to investigate the methods efficiently finding the support documents in the entity retrieval tasks under the TREPM model. Previous work treats support document finding as a conventional document retrieval problem. For example, Fang et al applied the structured retrieval on document, passage and entity level to find the relevant documents [2]. McCreadie et al applied the similar idea of structure language models on webpage title and body level for document findings [3]. Zheng et al applied the language model but only on document and snippet (50-word window size) level [4]. Some other teams consider the query constructions to refine the queries issued to search engines. For example, Vydiswaran et al tried to identify the information need (the narrative part of the topic) as a structured query which was represented as a relation including the relation description, the entity of focus, and the entity of interest [5]. Yang, Jiang, Zhang, & Niu, 2009 also did some query re-constructions by adding the synonym of topic entities into the query for searches [6].

Most systems treat support document finding as a conventional document retrieval problem: generate the various queries from information needs to collect support documents. However, this approach has the following limitations. Firstly, it is hard for a system to decide how to generate a proper query for a topic. For example, it is hard to decide whether it is better using topic entities as queries (e.g., "Claire Cardie") or it is better using descriptions as queries (e.g., "students of Claire Cardie") for a particular topic, especially when the topic is tricky. The query such as "organizations that award Nobel prizes" is easily confused with the query like "organizations awarded Nobel prizes". Secondly, the conventional document retrieval approach highly relies on the ranking, so that a proper threshold is required for cutting out the support documents. However, how to find the proper number for the threshold is hard. If the threshold is too high, it will bring a big support document set; if the threshold is too low, it will miss the low ranked support documents. Furthermore, the entity type is also important factor for finding support document, and how to integrate the type information in the retrieval, especially in the documents without category information, is also a problem. To tackle the problems of conventional document retrievals mentioned above, this work proposes a logistic regression method for support document findings.

3 FINDING SUPPORT DOCUMENTS WITH THE LOGISTIC REGRESSION APPROACH

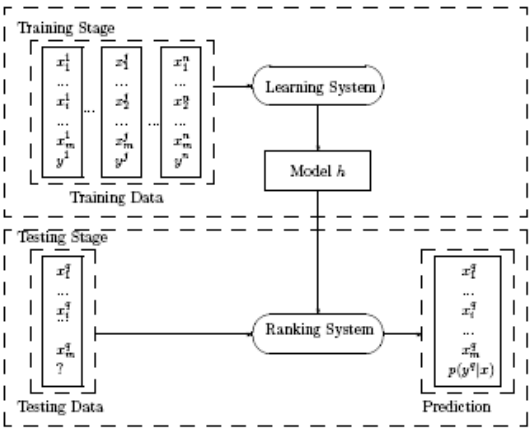


Figure 2 . The logistic regression framework

Learning to rank or machine learned ranking is a type of supervised machine learning method to automatically construct a ranking model from training data, such that the model can sort new objects according to their degrees of relevance, preference, or importance [7]. In this work, we interpret the support document finding task as a learning to rank problem. That is, a learning task predicts the support document according to the training data.

In recent years, more and more machine learning technologies have been used in information retrieval tasks for training the ranking model, such as the work on relevance feedback and automatically tuning the parameters of existing IR models. Most of the state-of-the-art learning to rank methods operates on the combining features extracted from query-document pairs through discriminative training, as seen in Figure 2. In this work, we propose using the logistic regression function on the learning to rank method to estimate the probability of a document to be the support document.

3.1 The logistic regression framework

The approach for support document finding in this study adapts the same structures of the general learning to rank method. We summarize the framework as follows:

- The input space is composed of feature vectors for each single document, represented as $(x_1, x_2, ...x_i, ..., x_m)$, and the corresponding labels y, which indicate whether a document is a support document. Therefore, the input training space is denoted as:

  $$\left\{(x_1^1,...x_i^1,...,x_m^1, y^1),...(x_1^j,...x_i^j,...,x_m^j, y^j),...,(x_1^n,...x_i^n,...,x_m^n, y^n)\right\}.$$

- The output space contains the prediction of the probability of each single document to be the support document according to the query, that is, $p(y=1|(x_1, x_{2,...}, x_m))$

- The hypothesis space contains functions that take the feature vectors as inputs and predict the probability of a document to be a support document. The function will be learned from the training data set. Logistic regression is a generalized linear model used for the probability estimation. It was first used in the TREC-2 conference by Berkeley researchers [8], and then it was extended into the medical and social science fields. In this study, we also use logistic regression for support document finding for the probability estimation. Logistic regression uses a sigmoid linear function. That is,

  $$p(y=1|(x_1,...,x_m)) = \frac{1}{1+\exp(w_0 + \sum_i w_i x_i)}$$

- The optimal function examines the accurate prediction of the ground truth label for each single document. With the logistic regression model, the prediction function directly predicts the probability of a document to be the support document with the given features. Therefore, The training data are used to estimate the parameters of w_i. It will be calculated as following:

  $$w_0^{t+1} \leftarrow w_0^t + \eta \sum_j [y^j - p(y^j = 1 | x_1^j,...,x_m^j, w)]$$

  For $i = 1,...,m$

  $$w_i^{t+1} \leftarrow w_i^t + \eta \sum_j x_i^j [y^j - p(y^j = 1 | x_1^j,...,x_m^j, w)]$$

  Here, η is the step size. The iteration step will be continuous until the parameter converges.

3.2 Features for the logistic regression method

Applying the logistic regression method to support document finding raises the question: what types of information should be used in the learning process? Two principles are followed in the process of feature selections: the feature should not be limited by the instances; and the feature should be general enough and domain independent so that the model could be generalized to other topics regardless of the domain. According to the above two principles, four types of features are generated for support document finding: query features, document features, rank features, and similarity features.

3.2.1 Query features

Query features or linguistic features are selected according to the principle described in Jones' studies [9]. They are the isolated characteristics of elements in queries (e.g., the length of query and the length of narrative) and hits (e.g., the percentage of overlap terms between the query and the document title). This study used the following features:

EntityNarrative is the feature that indicates if the query is generated from the topic entity or the narrative of information needs. In the pilot study, we find that both query generations are useful for some topics. Therefore, in the logistic regression method, we choose both methods to generate queries: the topic entities as queries and the narratives as queries.

EntityType is the target entity types required by each topic. Its value can be persons, locations, products, and organizations.

LengthEntity is the character length of topic entities without stop words.

LengthNarrative is the character length of narratives without stop words.

LengthRelation is the absolute character length difference between the topic entity and the narrative without stop words, i.e., LengthRelation = |LengthNarrative-LengthEntity|.

TokenLengthEntity is the token length of topic entities without stop words.

TokenLengthNarrative is the token length of narratives without stop words.

TokenLengthRelation is the absolute token length difference between the topic entities and the narratives without stop words, i.e., TokenLengthRelation = |TokenLengthNarrative - TokenLengthEntity|

IsSameEntity is to indicate whether topic entity has different entity surfaces in topic descriptions. If it is different, then the score is 1, otherwise it is 0. For example, the query described as "Journals published by the AVMA" has the topic entity of "American Veterinary Medical Associations" for the acronym term "AVMA" in the narrative part.

Hits is the numbers of relevant documents retrieved by the search engine.

Hitstrend is a binary feature with the value of (1, -1). It compares the hits of the topic-entities-as-queries and the narratives-as-queries for the same topic. If the number of hits from the topic-entities-as-queries is larger than the number of hits from the narratives-as-queries, then Histrend = 1. Otherwise, Histrend = -1.

3.2.2 Document features

Document features describe the characteristics of documents. The Wikipedia pages are supposed to have more authoritative information, so they are more likely to be the support. In this work, we especially detect Wikipedia as an important source for support documents. In the future, other sources with high quality

pages as support documents can be included, such as the entity's homepage. We define the following features:

IsWikipedia is a binary feature (1 or 0) to indicate whether this hit is from Wikipedia.

IsEntityWikipedia is a binary feature (0 or 1) to indicate whether this hit refers to a Wikipedia page, whose entry name is the same as the topic entity itself. For example, for the topic of "Medimmune, Inc.", the value of IsEntityWikipedia is equal to 1 for the hit of http://en.wikipedia.org/wiki/MedImmune.

3.2.3 Rank features

Rank related features are based on the rank information to indicate the popularity of the documents. These features can also give useful hints for support document findings. For example, we assume that the higher rank of a document, the more possible it is to be the support documents. We list the following features:

DocRank is the rank of the returned URLs from the search engines for each query.

RankScore is the normalized ranking score for each hit. It is calculated by summing up the reverse of rank for the same URL in the same topic. This score will merge the results on both the entities as queries and the narratives as queries. It is denoted as follows:

$$RankScore(URL) = \sum_{URL} \frac{1}{rank_{url}}.$$

NewRank is the new rank list ranked according to the RankScore, which considers the same URL in the same topic but retrieved by different queries.

3.2.4 Similarity features

Similarity features measure the similarity between the query and its retrieved document. We assumes that the shorter the semantic distances (measured by the semantic similarity) between a query and a document, the higher chance it is a document to be the support document. For example, for the query of "products of Medimmune Inc.", if the document title is also "products of Medimmune Inc.", then it is highly probable to be a support document for this query. We design some term distance measures to estimate the similarity. However, term distance measures suffer some drawbacks, such as missing the corresponding synonym sets or abbreviation forms. For example, "AVMA" is the acronym of "American Veterinary Medical Association". Therefore, semantic measurements are introduced. Some systems use a thesaurus to map the synonyms or abbreviations, e.g., WordNet or Wikipedia. Because it is hard to find the corresponding entries in the thesaurus for all queries narrated in sentences, an alternative, the WebDice coefficient, is introduced to the problem of word distances. They are defined as follows:

TitlePrecision is the rate of the overlapping terms between a query and its retrieved document's title to the number of terms in the query. This feature represents the similarity between a query and its hit. The terms exclude the stop words. For example, the TitlePrecision score of the topic "Products of Medimmune, Inc." is 0.667 for the document http://www.medimmune.com/ with the title of "Medimmune, Inc." The number of the overlapping terms in the query and the title is 2 (only the terms of "Medimmune" and "Inc" are counted), and the number of the terms in the query is 3 (only the terms of "products", "Medimmune" and "Inc" are counted).

$$\text{TitlePrecision} = \frac{num\_of\_terms\_in(query \cap title)}{num\_of\_terms\_in(query)}.$$

TitleRecall is the rate of the overlapping terms in the query and in the returned documents' titles to the number of terms in the title which represents the similarity between a query and its hits. Here, the terms exclude the stop words. For example, the TitleRecall score of the topic "Products of Medimmune, Inc." is 1 for the document http://www.medimmune.com/ with the title of "Medimmune, Inc.". The number of the overlapping terms between the query and the document is 2 (only the terms of "Medimmune" and "Inc" are counted), and the number of the terms in the query is 2 (only the terms of "Medimmune" and "Inc" are counted).

$$\text{TitleRecall} = \frac{num\_of\_terms\_in(query \cap title)}{num\_of\_terms\_in(title)}.$$

TitleDistance is the feature to measure whether the query terms are close to each other in the title part. We assume that a document with its title containing all query phrases close to each other is more relevant than one with the title containing the query keywords in a large window size. TitleDistance is the rate of query length to the scope of query terms in the title, as follows:

$$\text{TitleDistance} = \frac{num\_of\_terms\_in(query)}{num\_of\_terms\_in(scope\_of\_query\_terms\_in\_title)}$$

ContentPrecision is similar to TitlePrecision, but replaces the title part for the hit's content.

ContentRecall is similar to TitleRecall but replaces the title part for the hit's content part.

ContentDistance is similar to TitleDistance, which measures the query terms in the content part.

$$\text{ContentDistance} = \frac{num\_of\_terms\_in(query)}{num\_of\_terms\_in(scope\_of\_query\_terms\_in\_Content)}.$$

WebDiceOrg is to define the similarity between two queries by measuring the Web space similarity of two relevant documents retrieved by the two queries for the same topic. It is the approximation of F-measure in the web. Page counts of the query "P AND Q" can be considered as the co-occurrence of two words "P" and "Q" on the web. For example, the page count of the query of "Journals published by the AVMA" is 145,000. The page count for the document of "AVMA Journals" is 245,000. The page count for the document of "AVMA Journals - Reprints, ePrints, Permissions" is 159. From the page count similarity, "Journals published by the AVMA" is closer to "AVMA Journals" than "AVMA Journals - Reprints, ePrints, Permissions". The WebDiceOrg coefficient is to measure this similarity. Moreover, this coefficient has been demonstrated to outperform the other three modified co-occurrences (i.e. WebJaccard, WebOverlap, and WebPMI) in [10]. Therefore, in this study, we only use WedDiceOrg. The WebDiceOrg is defined as follows:

$$WebDiceOrg(query, title) = \begin{cases} 0 & if\ (H(query \cap title) \le c \\ \dfrac{2H(query \cap title)}{H(query) + H(title)} & otherwise \end{cases}$$

where H(query) denotes the page counts for the query of "query" in a search engine, and d denotes the page counts for the query of "query and title". c is a predefined threshold (e.g., c=5) to reduce the adverse effects caused by the random co-occurrence.

WebDice is the normalized WebDiceOrg score with the maximum value of WebDiceOrg, so that its value is between 0 and 1:

$$WebDice(query, title) = \frac{WebDiceOrg(query, title)}{\max(WebDiceOrg(query, *))}$$

4 EVALUATION

Seventy topics from the TREC entity retrieval 2009 and 2010 are used for the evaluation. The evaluation measurements are precision, recall and F-measure. Two experts were involved in assessing the ground truth of support documents for each topic. The requirement for the support document markup is to find at least one support document, which can provide the answers, for each topic. Moreover, the requirement for Wikipedia articles is to find corresponding Wikipedia articles for each topic if they exist. There are total 74 supporting documents annotated. The steps for support document annotations are as follows: firstly, experts generate proper queries to a search engine to find the possible support documents. Then according to the rank hits returned by the search engine, two annotators evaluate whether the hit is the support document for further answer entity extracting. For every topic, at least one support document must be found, and if there are more than ten support documents found, annotators only judge the first ten hits.

The experiment was designed to investigate whether the logistic regression approach can improve the performance of support document finding compared to the baseline systems.

- Baseline System I: the topic entities as queries for support document findings. In the experiment, we use the Google search engine and only consider the top 16 documents as support documents for the evaluation.
- Baseline System II: the narrative as queries for support document findings. The Google search engine is used to collect the support documents, and only top the 16 documents are considered as support documents for the evaluation.
- Baseline System III: the mixture support document rank list from the topic entities as queries and the narrative as queries. The mixture support document list ranks the documents from Baseline System I and Baseline System II with the following score:

$$ds(doc) = \sum_{query} \frac{1}{Orignal_{rank}(doc, query)}$$

- Experiment system: the logistic regression algorithm trains a model based on the features mentioned above and then applies this model to estimate the support document finding. The support documents are the documents from Baseline System I and Baseline System II. The document information includes their rankings, hits' URLs, hits' titles, hits' summaries, and query's page counts. For each hit, we mark down whether it is the support document according to the reference standard, i.e., whether this page contains the answer entities (ground truth). If this page contains the answer, it will be labeled as 1; otherwise, it will be labeled as 0. For the logistic regression algorithm, a ten-fold cross validation will be conducted. Firstly, the corpus is randomly divided into 10 folds. Every time, we train on the 9 folds and test on the last fold. The logistic regression can estimate the probability of a document to be the support document. We rank the documents according to the probabilities and choose the top 16 documents as support documents for the evaluation. With the 16 documents, precision, recall, and f-measure are calculated. The final precision, recall, and f-measure are the average results of the 10-fold evaluation.

Figure 3 shows the results of the baseline systems and the logistic regression method for the support document findings. Precision, recall, and f-measure at rank 1 to 16 are reported. The result of the logistic regression method is the average score of the 10-fold cross validation. Comparing the two baseline systems, Baseline System II (the narratives as queries) is significantly better than Baseline System I (the entities as queries) (for the two-tail t-test, $p<0.0001$). This indicates that in most cases, the narrative parts still are the better sources for the support document finding. There are no significant differences between Baseline System II (the narratives as queries) and Baseline System III (the mixture model) for the precision, recall, and the f-measure. The precision and f-measure of the logistic regression method are significantly better than the three base systems (for two-tail t-test, $p<0.0001$). However, there is no significant difference in recall.

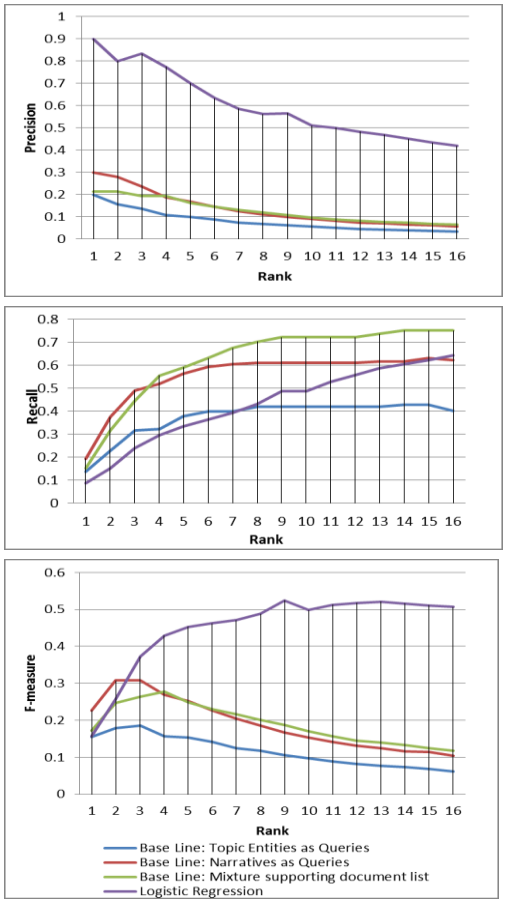


Figure 3 Precision, recall, and f-measure for the baseline systems and the logistic regression method

With Figure 4 of the error rates on four topic types, we found that for the logistic regression method for the support document finding, the topics about products are the hardest. We can see that the errors from products are higher than the other three. One reason is that the type of products usually is general category names, which need to be clarified in the special retrieval task. For example, CDs and software are assigned as products. Another reason is that the training sets for the products are too small.

Table 1: The error rates of four topic types

Topic Types	All Numbers	Error Numbers	Percentage
locations	224	3	0.013393
organizations	1408	55	0.039063
persons	480	20	0.041667
products	96	9	0.09375

Furthermore, the co-efficiency study for all features used in the logistic regression method is conducted. The higher score of the feature, the more important it is in the model. We discuss the results as follows: The Wikipedia entity page is one of the most important features. When a document from a Wikipedia page with the same entry as the entity name, it is more valuable than the other Wikipedia pages, according to the weight scores of isWikipedia and isEntityWikipedia. The rank of the document in a ranked list is also another factor in the learning method, especially the normalized ranking score which merges the multiple query results. If a document keeps appearing in the returned lists from different queries for the same topic, it has a higher probability be a support document. Except for the type of products, entity types have low effects on the learning. It is a hint that the more complicated entity type (e.g., products), the more important it is in the support document finding. Term length measures are better than the character length measures, which can be concluded from the weights of NarrativeTermLength vs NarrativeLength and EntityTermLength vs EntityLength. The document title part is more important than the hit's abstract part for the similarity measure between the query and the document. The "recall" of the query in the hit's title and abstract is more important than the "precision". It can be concluded from that ContentRecall, TitlePrecision, and TitleRecall are more important than ContentPrecision. Webdice does help to recognize the support documents, but the various hit measurements, such as query hits, have no effects.

5 CONCLUSION AND FUTURE WORKS

The task of support document finding is to find the documents containing the answer entities effectively and efficiently. In previous work, the conventional document retrieval is the most popular method for the support document finding. However, this method is threatened by some limitations. Although in most cases the narrative part is the best source for query generation, in some cases it will destroy the support document findings. For example, when the answer entities are only part of the Web pages (e.g., "students of Claire Cardie"), the topic entity is a better choice for the query generation. The direction of relation between the topic entities and the answer entities is another difficulty for query generation. For example, the query of "organizations that award Nobel prizes" presents the relation between answer entity (organization) and topic entity (Nobel prizes) as "award", which is the same as the query of "organizations that were awarded Nobel prizes" in the retrieval task with the assumption of the bag-of-words. It is also hard to find the support documents for topics using some terms never appearing in the corpus, such as "What are some of the spin-off companies from the University of Michigan?" With the above considerations, the lists of candidate support documents from different query generation strategies are generated. We propose a logistic regression method to estimate the probability of each document to be the support document by considering the above features. There are a total of 28 features used for the task, and they cover query features, hits features, and linguistic features. The results indicate that the logistic regression method is significantly better than the three baseline systems which treat the support document finding as a conventional document retrieval problem. Although the logistic regression method can improve the precision of the support document finding, the recall is still low. In future studies, we will investigate methods to improve the discovery of the more support documents. For example, since Wikipedia is one of important source for the support documents, the Wikipedia page with the same entry name as answer entities should be further processed to separate out more relevant features for support document finding.

REFERENCES

[1] Li, Q. and He, D. 2010. *Searching for Entities: When Retrieval Meets Extraction*. The Nineteenth Text Retrieval Conference (TREC 2010). Gaithersburg, MD.

[2] Fang, Y., Si, L.; Yu, Z.; Xian, Y.; Xu, Y. 2009. *Entity Retrieval with Hierarchical Relevance Model,Exploiting the Structure of Tables and Learning Homepage Classiers*. .the Eighteenth Text REtrieval Conference (TREC 2009). Gaithersburg, MD.

[3] McCreadie, R., et al. *University of Glasgow at TREC 2009: Experiments with Terrier*. 2009. the Eighteenth Text Retrieval Conference (TREC 2009). Gaithersburg, MD.

[4] Zheng, W. et al. *UDEL/SMU at TREC*, 2009, the Eighteenth Text Retrieval Conference (TREC 2009). Gaithersburg, MD.

[5] Vydiswaran, V., Ganesan, K., Lv, Y., He, J., and Zhai, C. (2009). *Finding Related Entities by Retrieving Relations: UIUC at TREC 2009 Entity Track*. In Proceedings of the Eighteenth Text REtrieval Conference (TREC 2009).

[6] Yang, Q., Jiang, P., Zhang, C., and Niu, Z. (2009). *Experiments on Related Entity Finding Track at TREC 2009*. In Proceedings of the Eighteenth Text Retrieval Conference (TREC 2009).

[7] Liu, T.-Y. (2009). *Learning to Rank for Information Retrieval*. Foundation Trends of Information Retrieval.

[8] William S. Cooper, Fredric C. Gey, and Daniel P. Dabney. *Probabilistic retrieval based on staged logistic regression*. In 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, Denmark, June 21-24, pages 198–210, New York, 1992. ACM.

[9] Jones, R., Rey, B., Madani, O., and Greiner, W. (2006). *Generating query substitutions*. In Proceedings of the 15th international conference on World Wide Web, WWW'06, page 387-396, New York, USA.

[10] Bollegala, D., Matsuo, Y., and Ishizuka, M. (2007). *Measuring semantic similarity between words using web search engines*. In Proceedings of the 16th international conference on World Wide Web, WWW '07, pages 757{766, New York, NY, USA.

The Sindice-2011 Dataset for Entity-Oriented Search in the Web of Data

Stéphane Campinas[*]
Renaud Delbru[*]
Diego Ceccarelli[‡]
Krisztian Balog[†]
Thomas E. Perry[*]
Giovanni Tummarello[*]

[*]Digital Enterprise Research Institute
National University of Ireland, Galway
Galway, Ireland
firstname.lastname@deri.org

[†]Norwegian University of
Science and Technology
Trondheim, Norway
krisztian.balog@idi.ntnu.no

[‡] ISTI-CNR
Dipartimento di Informatica
Università di Pisa
Pisa, Italy
diego.ceccarelli@isti.cnr.it

ABSTRACT

The task of entity retrieval becomes increasingly prevalent as more and more (semi-) structured information about objects is available on the Web in the form of documents embedding metadata (RDF, RDFa, Microformats, and others). However, research and development in that direction is dependent on (1) the availability of a representative corpus of entities that are found on the Web, and (2) the availability of an entity-oriented search infrastructure for experimenting with new retrieval models. In this paper, we introduce the Sindice-2011 data collection which is derived from data collected by the Sindice semantic search engine. The data collection (available at `http://data.sindice.com/trec2011/`) is especially designed for supporting research in the domain of web entity retrieval. We describe how the corpus is organised, discuss statistics of the data collection, and introduce a search infrastructure to foster research and development.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software

General Terms

Measurement, Performance, Experimentation

Keywords

Entity search, Web of Data, Entity corpus

1. INTRODUCTION

Entity retrieval, that is, returning "objects" (such as people, organisations, locations, products, etc.) in response to users' information needs, has received considerable attention recently from various research communities, as well as from the commercial sector. According to a recent study, more than half of web queries target a particular entity or instances of a given entity type [9]. Supporting the search and discovery of entities, therefore, is essential for ensuring a satisfying user experience.

EOS, SIGIR 2011 workshop, July 28, Beijing, China

The field of Information Retrieval (IR) is characterized by rigorous attention to evaluation and measurement. International benchmarking campaigns, such as the Text REtrieval Conference (TREC) and the Initiative for the Evaluation of XML Retrieval (INEX) play a key role in fostering IR research by providing a common platform, evaluation methodology, and relevance judgements to assess the quality of information access systems. The introduction of the expert finding task at the TREC Enterprise track in 2005 was a significant milestone on the path to entity-oriented retrieval. The goal of the expert finding task is to create a ranking of people who are experts in a given topical area [4]. Later, in 2007, INEX launched an Entity Ranking track [10]; here, entities are represented by their Wikipedia page and two tasks are considered: (i) *entity ranking*, where a query and target categories are given, and (ii) *list completion*, where a textual query, example entities, and, optionally, target categories are provided as input. In 2009, the Entity track at TREC started with the goal to perform entity-oriented search tasks on the Web, and defined the *related entity finding* (REF) task [2]. REF requests a ranked list of entities (of a specified type) that engage in a given relationship with a given source entity. The collection used there is a general Web crawl and entities are identified by their homepages. The 2010 edition of the track introduced an *entity list completion* task [3], similar to that of INEX, but the collection is a Semantic Web crawl, specifically, the Billion Triple Challenge 2009 (BTC-2009) dataset[1]. Looking at these developments over time, a shift of emphasis can be observed from the document-oriented web to the data-oriented web, or "Web of Data."

From the Web of Documents to the Web of Data.

Compared to the Web of Documents, the Web of Data is much more structured. However, since each Web of Data source might have its own defined schema, ranging from loosely to strictly defined, the data structure does not follow strict rules as in a database. Even within a given a data source, the schema might not be fixed and may change as the information grows. The information structure evolves over time, and new records can require new attributes. We therefore consider the Web of Data as being *semi-structured* [1].

The most important property of the Web of Data is that it is naturally organised around entities[2], and that each of these entities is uniquely identified by a Uniform Resource Identifier (URI). The

[1] `http://vmlion25.deri.ie/`

[2] Note that under standard Semantic Web terminology, this concept is referred to as *resource description*. We use "entity" instead of "resource" in order to emphasize that a resource describes an entity.

Web of Data, therefore, provides a natural setting for entity retrieval, also recognized by the Semantic Web Workshop series that organised the Semantic Search Challenge in 2010[3] and in 2011[4]. Two tasks are addressed: (i) *entity search* (2010 and 2011), where queries refer to one particular entity [7], and (ii) *list search* (2011), where queries target a group of entities that match certain criteria (similar in spirit to the list completion tasks at INEX and TREC, but here only a keyword query is provided, without type information or example entities).

The data collection used in both editions of the Semantic Search Challenge and also at the 2010 TREC Entity track, as a representative of Web of Data, is the BTC-2009 dataset. This collection, however, is not representative anymore of what can be found on the Web. It is mainly composed of RDF documents crawled in early 2009, and do not contain (or only in a very small proportion) data from RDFa or Microformats; as we show in Section 4, RDFa and Microformats make up more than half of the data in our collection. There exists a newer version of the BTC collection, BTC-2010[5] that contains 3 billion statements crawled in the early 2010. However, the aforementioned problem still persists.

Contribution.

The first contribution of this work is a new data collection, referred to as the *Sindice-2011 Dataset*, that is an accurate reflection of the current Web of Data. This dataset is made available to be used by the broad research community. One particular type of usage is to evaluate semantic search systems using this dataset. For instance, the 2011 edition of the TREC Entity track uses this collection to evaluate entity-related search tasks. Other communities could use this dataset for evaluating entity coreference resolution techniques or for benchmarking RDF data management systems.

Specifically, the Sindice-2011 Dataset contains 11 billion statements from 231 million documents that correspond to 1.738 billion entities. The exact values are reported in the Table 2. The data has been collected by the Sindice semantic search engine [8] from 2009 to 13/05/2011. The Sindice-2011 Dataset has been designed with the central aim of supporting research in web entity search. To conform to this particular need, the original Sindice crawl, which is composed of and organised around semi-structured documents, is processed and transformed into a corpus of entities.

Our second contribution is a collection of tools to help researchers working with this data. Specifically, these tools, written in Java, provide methods to process, index, and search for entities in the Sindice-2011 Dataset.

The remainder of the paper is organised as follows. Section 2 presents the Web of Data model as well as our entity-centric model. In Section 3, we discuss the creation and the organisation of the data collection. Section 4 provides statistics about the dataset. Section 5 introduces the tools and infrastructure for facilitating IR research and development using this collection. Finally, we conclude in Section 6.

2. WEB OF DATA

We use the term *Web of Data* to refer to the collection of machine processable content, exposed on the Web through metadata standards, e.g., Microformats, RDF, or RDFa. Without entering into a discussion about terminology, we use Web of Data as a casual synonym for the Semantic Web (or Web 3.0), and view it as a superset of Linked (Open) Data. Next in this section, a brief explanation of the RDF data model is given before presenting the Web of Data model. Readers familiar with these concepts may wish to proceed to the next section.

2.1 Resource Description Framework

The Resource Description Framework (RDF) is a generic data model that can be used for interoperable data exchange. In a nutshell, RDF facilitates the machine understanding of resource (entity) descriptions, hence allowing an automated processing of them. In RDF, a resource description is composed of statements about a given resource. A statement is a triple consisting of a subject, a predicate and an object, and asserts that a subject has a property with some value. A set of RDF statements forms a directed labelled graph. In an RDF graph, a node can be of the three types: URI, literal and blank node. An URI serves as a globally-unique identifier for a resource. A literal is a character string with an optional associated language and datatype. A blank node represents a resource for which a URI is not given. A blank node is considered as an identifier which is always scoped to the containing graph, e.g., a web document.

2.2 Web of Data Model

We define the Web of Data as part of the Hypertext Transfer Protocol (HTTP) accessible Web that returns semi-structured information using standard interchange formats and practices. The standard data interchange formats include HTML pages which embed RDFa or Microformats as well as RDF models using different syntaxes such as RDF/XML. It is possible to abstract the following core concepts in semi-structured data web publishing:

A Dataset is a collection of entity descriptions. One dataset is usually the content of some database which powers a web application exposing metadata, be this a dynamic web site with just partial semantic markups or a Semantic Web database which exposes its content, such as a Linked Open Data dataset. Datasets, however, can also come in the form of a single RDF document, e.g., an individual FOAF[6] file posted on a person's homepage.

An Entity description is a set of assertions about an entity and belongs to a dataset. Typical examples of entities include documents, people, events, products, etc. The assertions provide information regarding the entity such as attributes, for instance, the firstname and surname for a person, or relationships with other entities, for instance, the family members for a person. Each entity description can be uniquely identified, either using a URI or a blank node jointly with the graph identifier it originates from, i.e., the document's URL.

A View represents a single accessible piece of information, i.e., a web document, that provides a full or partial view over the content of the dataset. In the case of Linked Open Data, a typical view is the RDF model returned when one dereferences the URI of a resource. The Linked Open Data views are one-to-one mappings from the URI to the complete entity description. This, however, is more of an exception than a rule; other kinds of web data publishing mostly provide only partial entity descriptions in the views. For example, in the case of Microformats or RDFa, views are pages that talk about different aspects of the entity, e.g., one listing social contacts for a person, another listing personal data and a third containing all the posts by a user.

[3] http://km.aifb.kit.edu/ws/semsearch10/
[4] http://km.aifb.kit.edu/ws/semsearch11/
[5] http://km.aifb.kit.edu/projects/btc-2010/
[6] FOAF: http://www.foaf-project.org/

3. CREATION AND ORGANISATION OF THE DATA COLLECTION

The Sindice-2011 Dataset is based on documents containing semi-structured data gathered from the Web since 2009. More information about the provenance and content of the data is provided in §3.1. Given the initial format of the data, i.e., a set of semi-structured documents, an extraction step is necessary in order to transform the crawled data into a collection of entities (§3.2). Finally, because of the inherent structure of the Web of Data, we organise the collection of entities into two reusable hierarchical structures, each one providing a particular view of the information about an entity (§3.3).

3.1 General Characteristics of the Data Collection

Sindice started crawling Semantic Web data in 2009. Since then, the data has been continuously refreshed by the Sindice crawler. This ensures a relatively fresh snapshots of the Web of Data. The crawl covers various domains: e-commerce, social networks, social communications, events, scientific publications, and more. Part of the data is coming from Linked Data[7] datasets, such as DBpedia or Geonames. However, the collection also covers data published on the Web by individuals or large companies, such as LinkedIn, BestBuy, IMDB, and so forth. In total, the data is coming from more than 200 thousand second-level domains, as reported in Table 2. The collection covers various standardised data formats for web data publishing, such as RDF, RDFa, and Microformats. All the data is converted to RDF using Any23[8]. The dataset is highly heterogeneous. As shown in Table 2, the entities are described using more than 600 thousand ontologies and more than 5 million predicates.

3.2 Entity Extraction

A pre-processing step is required in order to extract entities from documents. The entity extraction algorithm is implemented using the MapReduce [5] programming model. In the map function in Alg. 1, the input key $< c >$ is the URL of a document, and the value $< s, p, o >$ is a triple in the document. The map phase simply outputs the same triple for two different join keys, one triple for the entity appearing as a subject and one triple for the entity appearing as an object. The outputs are then partitioned, sorted, and merged by the MapReduce framework. In the reduce phase, defined in Alg. 2, all the triples for each join key $< c, e >$ are aggregated together to build a graph containing the incoming and outgoing relations of an entity e. The resulting entity graph is similar to the sub-graphs pictured using dashed lines in Figure 1a.

Furthermore, we filter all entity graphs that are composed of only one or two triples. In general, such entity graphs do not bear any valuable information, and can be removed in order to reduce the noise as well as the size of the dataset.

The Table 1 reports two examples of extracted entities. For each one, the URL of the document, the identifier of the entity (URI or blank node identifier), and the content of the entity description is provided. Keeping track of the URL of the document is important when an entity is identified by a blank node, as the only way to uniquely identify such an entity is to combine the URL with the blank node identifier.

3.3 Data Organisation

The data collection is organised using two different structures, each one providing a particular view over the entities. The *document-entity centric format* (Sindice-DE) provides a representation of the entity within its context, the context, here, being a document. The *entity-document centric format* (Sindice-ED), on the other hand, provides a complete view of an entity across multiple contexts. These two structures are described in the next subsections.

Algorithm 1: Sindice-DE map function

input : Key $< c >$, Tuple $< s, p, o >$
output: Key $< c, e >$, Tuple $< s, p, o >$

`output` $(< c, s >, < s, p, o >)$
`output` $(< c, o >, < s, p, o >)$

Algorithm 2: Sindice-DE reduce function

input : Key $< c, e >$, List<Tuple> T
output: Key $< c, e >$, Graph g

g $\longleftarrow \emptyset$
for $< s, p, o > \in T$ **do**
  g.add$(< s, p, o >)$
end
`output` $(< c, e >,$ g$)$

3.3.1 Document-Entity Centric Format

In this collection, data is organised hierarchically in a document-entity centric format: there is one node in the first level of the hierarchy (directory in the file system) for each document, and each second-level node (sub-directory) below represents an entity within a document. Such a collection can be directly generated from the output of the reduce function of Alg. 2. This collection is useful to keep a global view of the context (i.e., document) where the entity is found. In general, a document provides contextual information about an entity, such as a list of people known by the entity in a social network or a list of publications authored by the entity in scientific publisher databases. Such a data structure is useful in certain retrieval tasks, where contextual information can help in disambiguating an entity.

3.3.2 Entity-Document Centric Format

In this collection, data is organised hierarchically in an entity-document centric format: there is one node in the first level of the hierarchy (directory in the file system) for each entity, and each second-level node (sub-directory) below represents a document that contains information about the entity. This collection is useful in providing a global view of an entity across multiple contexts, i.e., documents where pieces of information about the entity have been found. Such a data structure is useful in certain retrieval tasks, where the completeness of information prevails over quality. Aggregating information across contexts, however, is a double-edged sword. It leads to a more complete picture about an entity, but can also lead to entity descriptions of lower quality, as there is no control on what people publish about an entity on the web.

To create such a collection, a second data pre-processing step over the output of Alg. 2 is required in order to group entity information across contexts. The entity aggregation algorithm is also implemented using the MapReduce programming model. In the map function in Alg. 3, the input key $< c, e >$ is composed of the URL of a document and the identifier of an entity within this document. The value is the entity graph computed in Alg. 2. The map phase simply annotates the graph with the URL of the document, and outputs the entity identifier as the join key and the annotated

[7]Linked Data: `http://linkeddata.org/`
[8]Any23: `http://any23.org/`

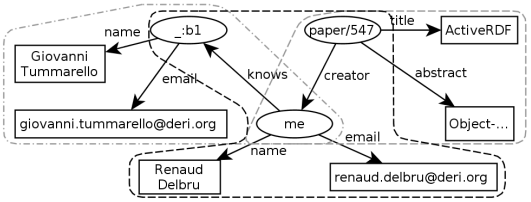
URL: http://dbpedia.org/resource/Modern_Times_(film)
ID: http://dbpedia.org/resource/Modern_Times_(film)

<http://dbpedia.org/resource/Modern_Times_(film)>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://dbpedia.org/ontology/Film> .
<http://dbpedia.org/resource/Modern_Times_(film)>	<http://dbpedia.org/property/director>	<http://dbpedia.org/resource/Charlie_Chaplin> .
<http://dbpedia.org/resource/Modern_Times_(film)>	<http://dbpedia.org/property/name>	"Modern Times" .
<http://dbpedia.org/resource/Modern_Times_(film)>	<http://dbpedia.org/property/writer>	"Paulette Goddard" .

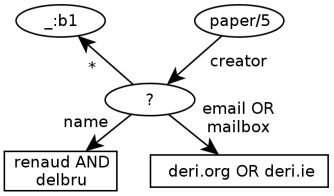
URL: http://belfast.ratemyarea.com/events/bumps-babies-and-pare-201663
ID: _:node997bf4dc6291719673b348d1a331f71

_:node997bf4dc6291719673b348d1a331f71	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2006/vcard/ns#VCard> .
_:node997bf4dc6291719673b348d1a331f71	<http://www.w3.org/2006/vcard/ns#fn>	"The Windmill Restaurant" .
_:node997bf4dc6291719673b348d1a331f71	<http://www.w3.org/2006/vcard/ns#org>	_:node15qbkbg04x42625 .
_:node997bf4dc6291719673b348d1a331f71	<http://www.w3.org/2006/vcard/ns#url>	<http://belfast.ratemyarea.com/places/the-windmill-restaura-105433> .

Table 1: Examples of extracted entities



(a) Visual representation of an RDF graph. The RDF graph is divided (dashed lines) into three entities identified by the nodes *_:b1*, *me* and *paper/547*.



(b) Star-shaped query matching the entity me where ? is the bound variable and ⋆ a wildcard

Figure 1: In these graphs, oval nodes represent resources and rectangular ones represent literals. For space consideration, URIs have been replaced by their local names.

graph as the value. The outputs are then partitioned, sorted, and merged by the MapReduce framework. In the reduce phase, the identify function (i.e., a function that copies the supplied intermediate data to the output) is applied and the output is a list of annotated graphs for each entity.

Furthermore, as a blank node identifier is only valid within its context, entities with blank-node identifiers are filtered from this collection. We also filter all incoming relations of type `rdf:type`. An entity representing a common concept reused on the web, such as `foaf:Person`, can have hundreds of millions of incoming links. These incoming links may consist of a huge amount of data, adding heavy load to the processing, while providing only very limited information about the entity; therefore, it is preferable to filter them. For the same reasons, we also filter incoming relations of type `dbpedia:wikilinks`.

Algorithm 3: Sindice-ED map function

input : Key $< c, e >$, Graph g
output: Key $< e >$, Graph g

g.setContext(c)
`output` ($< e >$, g)

4. STATISTICS

In this section, we report and discuss statistics about the data collection. We organise these statistics into three groups: data heterogeneity (§4.1), entity size (§4.2), and term distribution (§4.3). Additionally, we check the coverage of the dataset for results from previous evaluation campaigns (§4.4).

4.1 Data Heterogeneity

Figure 3a shows the distribution of data format across documents, i.e., how many documents are using a particular data format.

Description	Statistic
Documents	230,643,642
Domains	4,316,438
Second-level domains	270,404
Entities	1,738,089,611
Statements	11,163,347,487
Bytes	1,274,254,991,725
Literals	3,968,223,334 (35.55%)
URIs (as objects)	5,461,354,581 (48.92%)
Blank nodes (as objects)	1,733,769,572 (15.53%)
Unique ontologies	663,062
Unique predicate URIs	5,396,273
Unique literal words	114,138,096
Unique URIs (as objects)	409,589,991

Table 2: Statistics about the Sindice-2011 dataset

It is worth noting that one document can use more than one format. More than half (142 million) of the documents are containing some RDFa markup, a third (87 million) are containing plain RDF. If we aggregate the different Microformats together, more than half (130 million) of the documents are containing some Microformats markup. This shows one aspect of the diversity of the data collection, but it also illustrates that all standardised data formats well covered.

With respect to schema heterogeneity, the collection of entities is described using more than 600 thousand ontologies and with more than 5 million predicates, as reported in Table 2. Figure 3b depicts the distribution of the frequency of ontologies used across documents, i.e., the probability for an ontology to be used in exactly n documents. The distribution shows a power-law distribution following a Zipf function with a slope of $\alpha = 2.27$. Notice that most ontologies (99%) are used only once. However, the distribution tail is sparse, suggesting that a few ontologies are used in a large proportion of documents. For example, one ontology (`http://purl.org/dc/terms/`) is used in more than 150 million documents and another one (`http://www.w3.org/2006/vcard/ns\#`) is used in more than 64 millions documents. In Figure 3c, we report the distribution of frequency of predicates used across documents, i.e., the probability for a predicate to be used in exactly n documents. Similarly to the distribution of frequency of use of ontologies, the frequency of use of predicate URIs shows a power-law distribution following a Zipf function with a slope of $\alpha = 1.45$. A large number of predicates (98%) are used only once. However, the distribution tail is sparse, suggesting that a few predicates are used in a large proportion of documents. For example, two predicates are used in more than half of the documents (`http://www.w3.org/1999/02/22-rdf-syntax-ns\#type` in 155 million documents and `http://purl.org/dc/terms/title` in 140 million documents), and six other predicates are used in more than a quarter of the documents.

Figure 2 depicts the number of statements for each triple pattern, where U stands for URI, L for Literal, and B for Blank Node. For example, the pattern UUU represents triples with URIs at the subject, predicate, and object positions. It is worth noticing that the dataset consists of a large proportion of blank nodes, most of them coming from Microformats documents, which are removed from Sindice-ED.

With respect to entity types covered by the dataset, it is difficult to give a precise number for a particular concept, as one concept can be described by many different classes and ontologies. Instead, we report some observations from the list of the top-100 classes based on their document frequency and their probability distribution[9]. The most frequently used types describe persons and organizations. Products and business entities defined using the *GoodRelations* e-commerce ontology[10] are also very frequent. Further popular types concern reviews, bibliographic resources, locations and geographical data, bio-medical data, events, movies, and music.

4.2 Entity Size

We measure the size of an entity in terms of the number of triples in its RDF graph. Recall that we removed all entities having less than 3 triples from the Sindice-2011 Dataset. Figure 4a and Figure 4b show the distribution of entity sizes, i.e., the probability for an entity to have exactly n triples, for Sindice-DE and for Sindice-ED, respectively. The distributions show a power-law distribution following a Zipf function with a slope of $\alpha = 3.6$ for Sindice-DE and $\alpha = 2.6$ for Sindice-ED. In Sindice-DE, most entities are very small: 25% of them have three of four triples and 60% have ten or fewer triples. The average entity size is 5.32 triples, with a maximum entity size of 19,617 triples. In Sindice-ED, the entities are generally larger, with 20% of them having three or four triples and 75% having ten or fewer triples. The average entity size is 14.19 triples, with a maximum entity size of 5,454,094 triples. Such difference in size is explained by the fact that entities in Sindice-ED are aggregates of information coming from multiple documents.

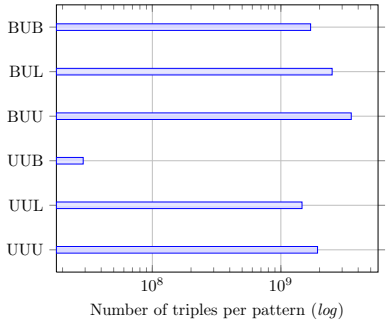


Figure 2: Distribution of triple patterns.

We also compute the length of a literal as the number of words in a literal. We exclude stop words and single character terms. Figure 4c shows the distribution of literal lengths, i.e., the probability for a literal to have exactly n words. The distribution shows a power-law distribution following a Zipf function with a slope of $\alpha = 2.58$. Most of the literals are very small, with 57% of them having exactly one word and 97% having ten or fewer words. The average literal length is 2.57 words, with a maximum literal length of 166,127 words.

4.3 Term Distribution

Fig 5a plots the distribution of words in literals and Figure 5b shows the distribution of URIs. The graphs depict the probability for a word or a URI to appear in exactly n documents. These distributions exhibit a similar slope ($\alpha \simeq 2$), indicating that a URI or a word follows the same probability of occurrence throughout the documents. However, the number of unique URIs (409,589,991) is orders of magnitude greater than the number of unique terms (114,138,096), which is due to the inherent function of URIs being used as unique identifiers (hence their cardinality is theoretically infinite).

In Figure 5c and in Figure 5d we plot, respectively, the distribution of literal terms over the predicates and the distribution of URIs over the predicates. The graphs show the probability, for each literal term or URI, to appear in exactly n distinct predicates. As in the previous graphs, these distributions exhibit a power-law distribution with slopes 2.36 (for literal terms) and 3.22 (for URIs). We note that the Zipf's α parameter in the URIs distribution is distinctly higher than in the literal terms distribution. This means that URIs often tend to appear jointly with a small number of predicates, while in literals this behavior is less pronounced. This is probably due to the fact that a term can represent different things and thus can appear in a larger set of predicates with different meanings.

4.4 Coverage

We checked all the URIs in the relevance assessment files from the 2010 and 2011 editions of the Semantic Search Challenge and from the ELC task of the 2010 TREC Entity track. The Sindice-

[9] `http://data.sindice.com/trec2011/resources/top100_class_distribution`

[10] `http://www.heppnetz.de/projects/goodrelations/`

2011 Dataset covers 99% of these URIs; the missing ones are either (1) not exist anymore on the Web or (2) synthetic URIs that were generated to replace blank nodes with URIs in the BTC-2009 dataset.

5. SEARCH TOOLS AND INFRASTRUCTURE

To support IR research and development with the Sindice-2011 Dataset, we provide a search infrastructure based on the Semantic Information Retrieval Engine, SIREn [6]. SIREn is an IR system designed for searching entities, specifically, according to the requirements of the Web of Data. Given a description of an entity, i.e., a star-shaped query such as the one in Figure 1b: locate the most relevant entities. Since RDF is semi-structured, SIREn is aimed to support three types of queries: (1) full-text search (keyword based) when the data structure is unknown, (2) semi-structured queries (complex queries specified in a star-shaped structure) when the data schema is known, or (3) a combination of the two (where full-text search can be used on any part of the star-shaped query) when the data structure is partially known. SIREn also supports top-k query processing using an adaptation of the well-known TF-IDF scoring function, providing a standard baseline for experimentation.

The developed tools provide functionalities to process, index, and retrieve entities. Using this infrastructure, new entity retrieval models and algorithms can be rapidly deployed and tested on top of the Sindice-2011 Dataset. For example, one can implement a two-step retrieval process where the infrastructure is used to retrieve a subset of the data collection that might be relevant to a query, and then, in a subsequent step, more advanced processing and ranking techniques are performed on this result set. One can also (1) implement on top of the infrastructure a query expansion technique that combines full-text and structured queries, (2) modify the way entities are ranked by adding weights to certain attributes or values, or (3) (an advanced user) can change how entities are indexed and implement her own query ranking function directly within the query processing framework. The search infrastructure can be downloaded from `https://github.com/rdelbru/trec-entity-tool`.

6. SUMMARY

In this paper, we have presented the *Sindice-2011 Dataset*, a collection providing an accurate reflection of the current *Web of Data*, i.e., a collection of semi-structured data (e.g., RDF, RDFa, Microformats, etc.). Statistics about the dataset are reported, which can be useful for developing appropriate search systems. With a sum of 11 billion statements and with information organized around 1.7 billion entities, the Sindice-2011 Dataset represents a real-world data collection that allows the research community to make a significant step forward in web entity search. In order to further support the research in that direction, we also provide a search infrastructure based on *SIREn*, the Semantic Information Retrieval Engine. The Sindice-2011 Dataset is available at `http://data.sindice.com/trec2011/`.

Bibliography

[1] S. Abiteboul. Querying Semi-Structured Data. In *Proceedings of the 6th International Conference on Database Theory*, pages 1–18, 1997.

[2] K. Balog, A. P. de Vries, P. Serdyukov, P. Thomas, and T. Westerveld. Overview of the TREC 2009 entity track. In *Proceedings of the Eighteenth Text REtrieval Conference (TREC 2009)*. NIST, 2010.

[3] K. Balog, P. Serdyukov, and A. P. de Vries. Overview of the TREC 2010 entity track. In *Proceedings of the Nineteenth Text REtrieval Conference (TREC 2010)*. NIST, 2011.

[4] N. Craswell, A. D. Vries, and I. Soboroff. Overview of the TREC-2005 enterprise track. In *The Fourteenth Text REtrieval Conference Proceedings*, TREC 2005, 2006.

[5] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51:107–113, January 2008.

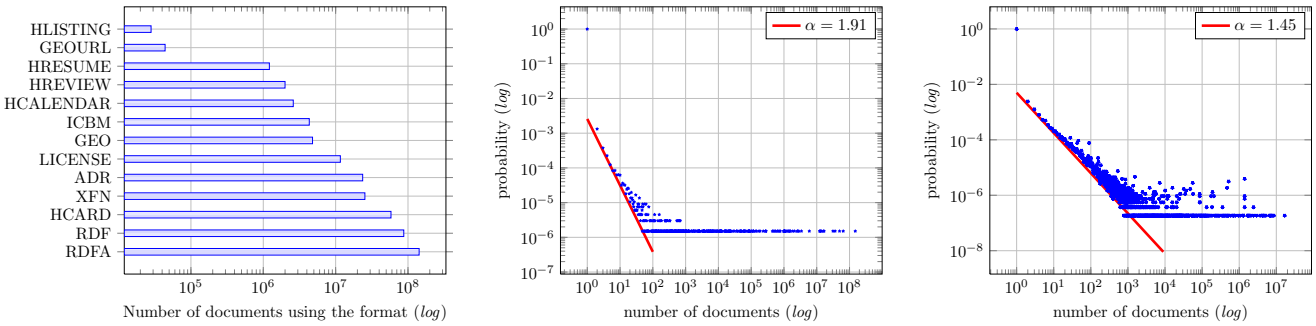
[6] R. Delbru, S. Campinas, and G. Tummarello. Searching web data: An entity retrieval and high-performance indexing model. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2011.

[7] H. Halpin, D. M. Herzig, P. Mika, R. Blanco, J. Pound, H. S. Thompson, and D. T. Tran. Evaluating ad-hoc object retrieval. In *Proceedings of the International Workshop on Evaluation of Semantic Technologies*, IWEST 2010, 2010.

[8] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello. Sindice.com: a document-oriented lookup index for open linked data. *Int. J. Metadata Semant. Ontologies*, 3:37–52, November 2008.

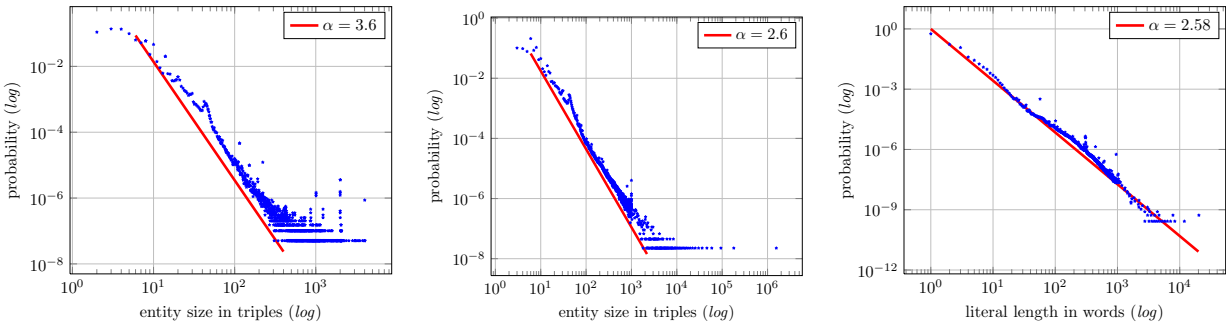
[9] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. In *Proceedings of the 19th international conference on World Wide Web*, pages 771–780, New York, New York, USA, 2010. ACM Press.

[10] A. P. Vries, A.-M. Vercoustre, J. A. Thom, N. Craswell, and M. Lalmas. Overview of the INEX 2007 entity ranking track. In N. Fuhr, J. Kamps, M. Lalmas, and A. Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *Lecture Notes in Computer Science*, pages 245–251. Springer Verlag, Heidelberg, 2008.



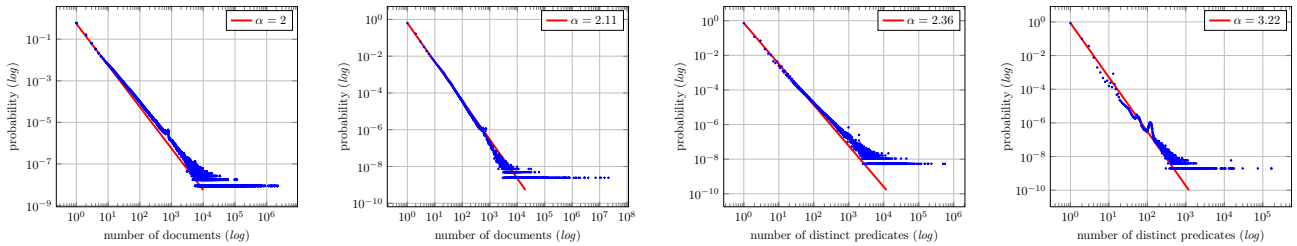
(a) Data formats (b) Frequency of ontologies used (c) Frequency of use of predicate URIs

Figure 3: Distributions of semantically structured data.



(a) Entity sizes in Sindice DE (b) Entity sizes in Sindice ED (c) Literal lengths

Figure 4: Entity size distributions.



(a) Terms in literals (b) URIs (c) Literal terms over predicates (d) URIs over predicates

Figure 5: Term distributions.

Cross-Domain Bootstrapping for Named Entity Recognition

Ang Sun Ralph Grishman
New York University
719 Broadway, Room 706
New York, NY 10003 USA
{asun, grishman}@cs.nyu.edu

ABSTRACT
We propose a general cross-domain bootstrapping algorithm for domain adaptation in the task of named entity recognition. We first generalize the lexical features of the source domain model with word clusters generated from a joint corpus. We then select target domain instances based on multiple criteria during the bootstrapping process. Without using annotated data from the target domain and without explicitly encoding any target-domain-specific knowledge, we were able to improve the source model's F-measure by 7 points on the target domain.

Categories and Subject Descriptors
I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *Text analysis*

General Terms
Languages

Keywords
named entity identification and classification, domain adaptation, bootstrapping

1. INTRODUCTION
Named Entity Recognition (NER) is a fundamental information extraction task with the objective of identifying and classifying proper names into certain pre-defined categories such as *persons*, *organizations* and *locations*. Supervised NER systems perform well when they are trained and tested on data from the same domain. However, when testing on a new domain which is different or even slightly different from the domain they were trained on, their performance usually degrades dramatically. For example, Ciaramita and Altun [8] reported that a system trained on the CoNLL 2003 Reuters dataset achieved an F-measure of 0.908 when it was tested on a similar Reuters corpus but only 0.643 on a Wall Street Journal dataset.

The performance degradation phenomenon occurs when one has access to labeled data in one domain (the *source domain*) but has no labeled data in another domain (the *target domain*). This is a typical situation as one might be able to expend the limited effort required to annotate a few *target* examples as a test bed but cannot afford to annotate additional examples for training purpose. However, it is usually the case that we have access to abundant unlabeled data in the target domain.

This paper works on this common scenario where we have access to labeled data in the *source domain* and only unlabeled data in the *target domain*. We propose a cross-domain bootstrapping (CDB) algorithm to iteratively adapt the source domain model to the target domain. Specifically, we first train an MEMM (maximum entropy Markov model [27]) source/seed model using the labeled data in the source domain and then apply it to the unlabeled data pool of the target domain. We then select *good* instances based on multiple criteria and use them to re-train and upgrade the seed model.

CDB differs from previous bootstrapping algorithms in several aspects. First, the seed model is generalized with word clusters. A model trained on the source domain may perform poorly on the target domain partly because there are many target domain specific words (for both names and context words) that have not been observed in the source domain. This motivates our work to use word clusters as additional features to generalize the seed model. The assumption is that even if we have not observed a *target* word *Wt* in the source domain, another word *Ws* in the source domain might share the same cluster membership with the word *Wt*. The cluster level feature still fires even if the lexical feature is absent from the source domain. More specifically, we mix the labeled source domain corpus with the unlabeled target domain corpus and generate the Brown word clusters (Brown et al., [5]) from this joint corpus. We then extract cluster memberships as features to augment the feature based NER system trained on the source domain.

CDB is novel in its multi-criteria-based instance selection method. Standard bootstrapping usually adopts a single criterion which is based on the *confidence* measure only, promoting those instances that are most confidently labeled from the unlabeled data. This might not be a problem when the data used for training the seed model and the unlabeled data are drawn from the same domain. However, in our cross domain setting, the most confidently labeled examples are those that have been observed in or are most similar to the source domain. CDB uses multiple criteria to select instances that are novel, confident, representative and diverse. It first uses novelty as a filter, maintaining only these instances that are specific to the target domain. It then ranks these novel instances based on a confidence measure. Top ranked instances contribute to a candidate set. Finally, it applies representativeness

EOS, SIGIR 2011 workshop, July 28, Beijing, China.

and diversity measures to all the candidates and selects a subset of them for promotion.

The rest of this paper is organized as follows: The next section positions us with respect to related work. Section 3 briefly introduces our NER task and source and target domains. Section 4 describes the CDB algorithm in detail. We present an experimental study in Section 5 and conclude in Section 6.

2. Related Work

There is a large body of domain adaptation research on different NLP tasks. Due to space limitations, we only discuss work related to NER.

Supervised domain adaptation for NER works on the scenario where one has labeled data from both the source and the target domains [11, 14]. Daumé III [11] has shown that a better model can be learned from the labeled data by making three copies of the features: general, source-dependent and target-dependent. Without labeled data from the target domain, it is impossible to distinguish and jointly learn the three types of features. Our work also generalizes and augments features but is obviously different from the above approaches in that the word cluster features are extracted from an unlabeled corpus.

Semi-supervised domain adaptation for NER deals with the situation such as ours where one only has labeled data from the source domain but not the target domain [23, 39]. (We can also refer to this branch of research as unsupervised learning because there is no supervision from the target domain.) Jiang and Zhai [23] studied domain adaptation from an instance weighting perspective and proposed a balanced bootstrapping algorithm in which the small number of instances promoted from the target domain was re-weighted to have an equal weight to the large number of source instances. Their instance selection was based on a confidence measure. Wu et al. [39] described a domain adaptive bootstrapping framework where the instances were selected based on informativeness. Neither of the two approaches generalized their seed models as we have done and both of them used a single instance selection criterion instead of the multiple criteria we have used.

Standard bootstrapping for domain-specific NER or semantic lexicon acquisition works on the *target domain* directly (both the seed examples and the unlabeled data are from the target domain) and typically adopts a confidence measure for selecting new instances [20, 28, 31, 40]. It has been shown that seed selection is very important for standard bootstrapping [37]. The way we generalize our seed model is similar, but not identical to seed selection in a sense that both of the approaches try to provide a better starting point for bootstrapping.

3. Task and Domains

Our NER task is similar to those defined in some benchmark evaluations such as MUC-6 [18], CoNLL-2003 [35] and ACE-05 [1]. Given a raw sentence, the goal is to identify name expressions and classify them into one of the following three types: PER (person), ORG (organization) and GPE (Geo-Political entity). We choose to work with these three types as they are the most frequent ones in our target domain. Figure 1 illustrates examples from both domains.

Source Example: <GPE>U.S.</GPE> <ORG>Defense</ORG> *Secretary* <PER>Donald H. Rumsfeld</PER> *discussed the resolution …*

Target Example1: *The ruler of* <GPE>Saudi Arabia</GPE> *is* <PER>Fahad bin Abdul Aziz bin Abdul Rahman Al-Sa ?ud</PER>.

Target Example2: … *where Sheikh* <PER>Abdul Sattar al-Rishawi</PER> *and the* <ORG>Anbar Salvation Front</ORG> *became a force for stability*.

Figure 1: Examples of NER task and domains

Our **target domain** documents are from publicly available reports (in English) on terrorism, such as those from the Combating Terrorism Center at West Point[2]. There are many domain-specific characteristics of our target domain. Just to mention a few: it is noisy as we automatically convert PDF documents to text files (footnotes might be inserted in the middle of natural language sentences and punctuation might not be converted correctly such as the example "Al-Sa?ud"); it is Arabic name (transliterated) rich and the naming convention is different from English names; name variation is another noticeable problem.

We choose the ACE-05 annotated data as the **source domain** because the degree of overlap between the ACE-05 data and the target domain data is higher than the MUC-6 and the CoNLL-2003 datasets, which are from the 90s.

4. Cross-Domain Bootstrapping

We first present an overview of the CDB algorithm, and then we describe in detail the generalization of a seed model with word clusters and the multi-criteria-based instance selection method.

4.1 Overview of the CDB Algorithm

The input to the algorithm is a labeled dataset from the source domain and an unlabeled dataset from the target domain, denoted by D_S^L and D_T^U respectively. Let G denote the growing set which contains selected instances during each round t and is initialized to be an empty set at round 0; the CDB algorithm repeats the following steps until it meets a stopping criterion.

1. Train an NER seed model M_t with $D_S^L \bigcup G_t$, *generalize* it with word clusters
2. Label D_T^U using M_t
3. Select $D_T^L \subseteq D_T^U$ based on *multiple criteria*
4. Update: $G_{t+1} = G_t \bigcup D_T^L$ and $D_T^U = D_T^U \setminus D_T^L$

The output of the CDB algorithm is an NER model which will be used to identify and classify named entities in the target domain. It is important to mention that the seed model M is generalized with word clusters at each round, not just at the beginning of the algorithm.

[1] http://www.itl.nist.gov/iad/mig/tests/ace/2005/doc/ace05-evalplan.v3.pdf

[2] http://www.ctc.usma.edu/publications/sentinel

4.2 Seed Model Generalization

Ungeneralized seed model: NER is typically viewed as a sequential prediction problem. Given a sentence containing a sequence of tokens, the goal is to assign a name class to each one of the tokens. Formally, let $s = (t_1, ..., t_N)$ be an input sequence of tokens and $c = (c_1, ..., c_N)$ be the output sequence of name classes, the prediction problem is to estimate the probability $P(c \mid s)$. To facilitate the learning procedure, we use the standard BIO decoding scheme. Each name type *c*, other than the type *O* (not a name), is split into subtypes *B-c* (beginning of *c*) and *I-c* (continuation of *c*). Although the less used BILOU (beginning, inside, last, outside and unit-length) scheme was claimed to outperform the BIO scheme in [30], we did not observe the same behavior in our target domain (see Section 5.2). The BIO representation gives us 7 classes (3 name types $\times$ 2 subtypes + 1 not a name class).

We build an MEMM [27] model with the following customary features: 1) current token t_i; 2) lower case tokens in a 5-token-window $(t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2})$; 3) a word type feature (all-capitalized, initial-capitalized, all-digits, etc.) for each token in the context window; 4) previous prediction c_{i-1}; 5) conjunction of c_{i-1}, 1), 2) and 3); 6) gazetteer membership of context tokens in a dictionary of country and US state names. Note that we do not extract POS tags as features since a good target domain POS tagger is not available to us.

As the model makes a local decision, that is, it predicts the name class for each individual token based on its feature vector, it might produce illegal transitions between name classes (e.g., B-PER followed by I-ORG). So we run the Viterbi algorithm to select the sequence of name classes with the highest probability.

Generalizing seed model with word clusters: The sparsity of lexical features is a notorious problem in many supervised NLP systems. Recent advances in generating word classes from unlabeled corpora and adding them as features has proven to be an effective way of generalizing the lexical features to alleviate sparsity [29, 30, 36]. The unavailability of a cross-domain unlabeled corpus hinders the direct adaptation of this technique to our cross-domain setting. Ideally, we would prefer an unlabeled corpus containing words of both domains so that the word classes can generalize for both domains. So we propose to generate a joint corpus by mixing the labeled source data with the unlabeled target data.

We then follow previous research and use the Brown algorithm [5] to generate word clusters from the joint corpus. The Brown algorithm is a hierarchical clustering algorithm which initially assigns each word to its own cluster and then repeatedly merges the two clusters which cause the least loss in average mutual information between adjacent clusters based on bigram statistics. By tracing the pairwise merging steps, one can obtain a word hierarchy which can be represented as a binary tree. A word can be compactly represented as a bit string by following the path from the root to itself in the tree, assigning a 0 for each left branch, and a 1 for each right branch. Table 1 shows some words and their bit string representations obtained from the joint corpus.

By using prefixes of different lengths one can produce word clusters of various granularities so as to avoid the commitment to a single cluster. We used clusters with lengths 4, 6, 10 and 20 and augmented the *previous*, *current* and *next* token features with word clusters [29, 30, 36]. For example, when we extract features for the *current* token "John", we will add a cluster feature *curPrefix6 = 110100* when we use length 6. (Note that the cluster feature is a nominal feature, not to be confused with an integer feature.) Now, even if we have not observed "Abdul" in our source domain, its cluster level feature still fires given that the *curPrefix6* feature is the same for both "John" and "Abdul".

Table 1: An example of words and their bit string representations. Bold ones are transliterated Arabic words.

Bit string	Examples
110100011	*John*, *James*, *Mike*, *Steven*, *Dan*, ...
11010011101	***Abdul***, ***Mustafa***, ***Abi***, ***Abdel***, ...
11010011111	***Shaikh***, ***Shaykh***, ***Sheikh***, ***Sheik***, ...
1101000011	*President*, *Pope*, *Vice*, ...
111111110	***Qaeda***, ***Qaida***, ***qaeda***, ***QAEDA***, ...
00011110000	*FDA*, *NYPD*, *FBI*, ...
000111100100	***Taliban***, ...

It is worth mentioning an additional benefit of using word clusters: different Arabic name variants are grouped together such as variants of "Shaikh" and variants of "Qaeda" in Table 1. Without analyzing and comparing the internal structure of the words, such as computing the edit distance between different words, the clustering algorithm itself is able to capture this domain-specific knowledge.

4.3 Multi-Criteria-based Instance Selection

Most standard bootstrapping algorithms use a confidence measure as the single selection criterion. In practice, this works well under the single domain setting. In a cross-domain setting like ours, the most confidently labeled instances are highly correlated with the source domain and hence contain little information about the target domain. In contrast, the CDB algorithm adopts an instance selection method based on multiple criteria.

Instance: We define an instance $I = \langle v, c \rangle$ as the feature vector *v* and the name class *c* of the current token t_i under consideration. Although *sentence* seems to be a more natural unit than *token* for a bootstrapped NER system [21], our sentences contain many target domain specific names and context words which undermines the reliability of any confidence measure defined over a sentence. However, when broken down to the token level, it is easy to design a relatively reliable confidence measure, as the feature vector *v* is essentially extracted from a short context window $(t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2})$. Also, the feature vector does contain the transition from the previous name class to the current class as we include the prediction of the previous token t_{i-1} as a feature (The class of the previous token is known after we run Viterbi over the whole sentence). Moreover, the NER model outputs normalized probabilities predicting the name classes based on the vector *v* and it is convenient to add the vector to the feature file for re-training the NER model.

Novelty: Novelty prefers an instance *I* that contains target-domain-specific tokens in its context

window $(t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2})$ and can be confidently labeled by the seed model. If all the context tokens have been observed in the source domain then the instance contains less target domain information than others. However, if all the 5 tokens are target-domain-dependent then the seed model's prediction of the instance might not be reliable. So we tried different values (1, 2 and 3) for the number of target-domain-specific tokens in the context window and different positions (token index in the range [*i-2, i+2*]) and found that the following simple measure worked the best: if the current token t_i is the only target-domain-specific token then the instance is considered to be novel.

Confidence: A reliable confidence measure is crucial to the success of a bootstrapping algorithm. If one bogus instance is selected, it will leads to the selection of many other bogus instances. CDB's confidence measure not only considers how confident an instance is labeled locally but also globally.

The ***local confidence*** of an instance *I* is defined as the posterior entropy of the 7 name classes c given the instance's feature vector *v*.

$$LocalConf(I) = -\sum_{c_i} p(c_i \mid v) \log p(c_i \mid v)$$

It is non-negative. It achieves its minimum value 0 when the MEMM model predicts a class c_i with probability 1 (more precisely when $p(c_i \mid v) = 1$). It achieves its maximum value when the predictions are evenly distributed over the 7 classes. So the lower the value, the more confident the instance is.

The ***global confidence*** concerns how other occurrences of the *current* token t_i in the instance *I* are labeled in the whole corpus. The linguistic intuition here is that one name usually belongs to one class in a corpus [13, 24, 38]. The CDB algorithm would prefer to select name tokens that can be consistently labeled in the whole corpus. So we gather all other occurrences of t_i in the corpus, generate their feature vectors and take as the name class of each occurrence the one with the highest probability returned by the MEMM model. The BI tags of the class are then deleted, for example, B-PER would become PER. This is because a name token can be at different positions (e.g., *Smith* can be either B-PER or I-PER). So global confidence uses 4 name classes instead of 7. We then compute the global confidence as below:

$$GlobalConf(I) = -\sum_{c_i} p(c_i) \log p(c_i)$$

where $p(c_i)$ is the corpus level probability of t_i belonging to the class c_i. It is defined as the number of times t_i is predicted with the class c_i divided by the total number of occurrences of t_i in the corpus. For example, if "Abdul" appears 10 times in the corpus and is predicted 8 times as PER, then the probability of "Abdul" belonging to the class PER is 0.8. Similar to the *local confidence* measure, the lower the value of the *global confidence*, the more confident the instance is.

We then propose a final measure to combine the two confidence measures. We simply take the product of the two measures.

$$ComConf(I) = LocalConf(I) \times GlobalConf(I)$$

Density: In addition to the most confident instances, CDB also aims to select the most representative instances. We use a density measure to evaluate the representativeness of an instance. The density of an instance *i* is defined as the average similarity between *i* and all other instances *j* in the corpus. The most representative instance is the one with the largest density value.

$$Density(i) = \frac{\sum_{j=1 \wedge j \neq i}^{N} Sim(i, j)}{N-1}$$

where *N* is the total number of instances in the corpus and $Sim(i, j)$ is the similarity between the two instances, which is defined as the standard *Jaccard Similarity* between the feature vectors *u* and *v* of the two instances. The *Jaccard Similarity* between *u* and *v* is defined as the number of matched features of *u* and *v* divided by the number of unique features in the union of *u* and *v*. The match function for a feature *f* returns 1 if the values of *f* in *u* and *v* are the same and 0 otherwise.

$$Sim(u, v) = \frac{|u \bigcap v|}{|u \bigcup v|}$$

Alternatively, we could find the angle between the two feature vectors and compute the *Cosine Similarity* between them. However, as all the features for NER take discrete values the simpler *Jaccard Similarity* suffices to capture the similarity between feature vectors.

Diversity: CDB also aims to select instances as diverse as possible. Intuitively, if we have observed an instance and its similar instances a sufficient number of times then we cannot learn more *new* information from them. Take the instance "*, said * in his*" for example, where * is the *current* token, which we restrict to be a target-domain-specific token (*novelty*) and is highly likely to be a *person*; it is *confident* at both local and global levels given that the context is salient and it is probably very *dense* too. However, repeatedly selecting such instances is a waste of time because no additional benefit can be gained for CDB.

So *globally*, once an instance has been selected, it is removed from the unlabeled target corpus. The CDB algorithm will never select it again in the following rounds. *Locally*, in a single round, when we evaluate an instance *i*, we will compare the *difference* between *i* and all the instances *j* that have already been selected. If the *difference* is large enough, we accept *i*; otherwise we reject it. One possibility of measuring the *difference* is to directly use the similarity measure $Sim(i, j)$. But this tends to reduce the chance of selecting *dense* instances given that a dense instance has many similar instances and tends to occur more frequently than others. For example, if we already selected the instance "*, said Abdul in his*", the chance of selecting other similar instances "*, said * in his*" is low. We then turn to a compromise measure to compute the *difference* between instances *i* and *j* which is defined as the difference of their density values. By setting a small threshold for $diff(i, j)$, dense instances still have a higher chance to be selected while a certain degree of diversity is achieved at the same time.

$$diff(i, j) = Density(i) - Density(j)$$

Order of applying different criteria: CDB first applies the *novelty* measure to all the instances in the corpus to filter out non-novel instances, and then it computes the *confidence* score for each novel instance. Instances are then ranked in increasing order of confidence score (lower value means higher confidence) and the top ranked *M* instances will be used to generate a candidate set. CDB now applies the *density* measure to all the members in the candidate set and ranks the instances in descending order of density (larger value means higher density). Finally, CDB accepts the first instance (with the highest density) in the candidate set and selects other candidates based on the *diff* measure.

5. Experiments

5.1 Data, Evaluation and Parameters

Source domain data: Table 2 summarizes the source domain data (ACE 2005) used in this paper. The 2005 dataset contains 6 genres: Broadcast Conversations (bc), Broadcast News (bn), Conversational Telephone Speech (cts), Newswire (nw), Usenet (un) and Weblog (wl). We randomly selected 10 documents from each genre for testing purposes.

Table 2: Source domain data.

Genre	Training (#doc)	Test (#doc)
bc	50	10
bn	216	10
cts	29	10
nw	97	10
un	39	10
wl	109	10
Total	540 (285K words)	60 (31K words)

Target domain data: Table 3 lists the sizes of the unlabeled and the labeled corpus as well as the number of instances of the 3 name types in the labeled corpus. The labeled data (for testing purpose) is annotated according to the ACE 2005 guideline[3]. This test data and the word clusters generated from the TDT5 are available at http://cs.nyu.edu/~asun/#DataSet&Software.

Table 3: Target domain data.

Data	Size
Unlabeled/Labeled	10M/23K words
PERSON	771 instances
ORGANIZATION	585 instances
GPE	559 instances

Corpora for generating word clusters: To study the impact of unlabeled corpora on cross-domain NER, we downloaded the word clusters generated by Ratinov and Roth [30] and Turian et al., 2010 [36]. Following them, we used Liang's implementation of the Brown algorithm and generated 1,000 word clusters for both the TDT5 (the English portion) and the joint corpus [25]. The TDT5 is selected because it contains news from the year 2003 and some ACE 2005 training documents are also from 2003.

Table 4: Corpora for generating word clusters

Data	Size(#words)
Reuters 1996[4]	43M
Cleaned RCV1 [5]	37M
TDT5 (LDC2006T18)	83M
Joint Corpus (ACE training + Unlabeled Target data)	10M

Evaluation: Evaluation is done at the named entity level, not the BIO tags level. Boundary errors are penalized. We used the standard precision, recall and F1 score.

Parameters: As the CDB algorithm uses several parameters, we summarize them in Table 5 for easy reference. Because CDB runs the Viterbi algorithm on each sentence, it is time consuming to run it on the whole unlabeled data. So we divided them sequentially into 6 batch sets and picked a random set for bootstrapping in each iteration. The candidate set contains more instances than the CDB algorithm will actually select because of the density and diversity measures. As the majority of tokens belong to the not-a-name class, we select the same amount of name/not-a-name instances in order to provide a balanced distribution between the name and the not-a-name classes. We tried many different parameter values and those in Table 5 were selected by eyeballing the quality of selected instances.

Table 5: Parameters for CDB.

Parameter	Size or Value
Batch Set	60K sentences (roughly 1.7M tokens)
Candidate Set	2000/2000 name/not-a-name instances
D_T^L	300/300 name/not-a-name instances
Iterations	30
$diff(i, j)$	0.001

5.2 Performance of the Source Domain Model

We build two *source* models using the ACE 2005 training data in Table 2. The first model is an HMM model with the BILOU states encoding. The second model is the MEMM model with the BIO encoding using the conventional features as described in Section 4.2 (no word cluster features). Their performances on both the source and the target domains are summarized in Table 6.

Table 6 shows that although both models achieve F1 of 80s on the source domain, they generalize poorly on the target domain. Comparing the HMM model with the MEMM model, it seems that the feature based model generalizes better to the target domain than the generative HMM model even though we did use the word type features as a back-off model similar to [1]. We did

[3] http://projects.ldc.upenn.edu/ace/docs/English-Entities-Guidelines_v5.6.1.pdf

[4] http://cogcomp.cs.illinois.edu/page/software_view/4

[5] http://metaoptimize.com/projects/wordreprs/

not observe the advantage of using the BILOU scheme as reported in [30] for both the source and the target domains. Although we could not determine the exact reason why this happens for the source domain, for the target domain, it contains long transliterated Arabic names implying that the state transition from "I" to "I" is more common in the target than the source domain. This type of transition might not be observed enough and estimated sufficiently by the fine grained BILOU scheme.

Table 6: Performance of source models over the 3 name types

Model	P	R	F1	Domain
HMM(BILOU)	82.49	81.16	81.82	Source
HMM(BILOU)	53.29	57.52	55.33	Target
MEMM(BIO)	84.68	81.54	83.08	Source
MEMM(BIO)	70.02	61.86	65.69	Target

5.3 Performance of the Generalized Model

We augmented the source model with word clusters (as described in Section 4.2) from the four unlabeled corpora in Table 4. Their performance on the target domain is shown in Table 7.

Table 7 shows the superiority of using a joint corpus to generate word clusters: the 10M words joint corpus outperformed the other 3 larger corpora. The TDT5 corpus is more than 8 times larger than the joint corpus, but is still 1 point behind. Using word clusters from the Reuters corpora (Reuters 1996 and RCV1) have shown to improve NER systems' performance on the CoNLL 2003 NER task [30, 36]. But they provided limited performance gain for our model when testing on the target domain. The results shown here indicate the necessity of using a joint corpus or ideally a general purpose corpus for generalizing the source domain model for cross-domain NER.

Table 7: Performance of the generalized source model

Word clusters	P	R	F1
No Cluster	70.02	61.86	65.69
Reuters 1996	69.26	64.26	66.67
RCV1	66.33	64.42	65.36
TDT5	70.76	66.51	68.57
Joint Corpus	72.82	66.61	**69.58**

5.4 Performance of CDB

We start with the generalized source/seed model and run the CDB algorithm on the unlabeled target domain corpus using the parameters specified in Table 5. As mentioned earlier, the seed model is generalized sequentially, that is, word cluster features are used during each round. We plot F1 score against iteration in Figure 2. The results are obtained by testing the updated model during each round on the labeled target domain data. The results are averaged on 10 runs.

There are several clear trends in Figure 2. First, without using the novelty measure (the line at the bottom), CDB performs worse than GSM. Although the seed model is already generalized with word clusters, the most confidently labeled instances might still be more similar to the source than the target domain. This indicates that novelty is a necessary measure for cross-domain NER. Comparing the two confidence measures: *ComConf* and *LocalConf*, in general, *ComConf* outperforms *LocalConf*. After using the novelty measure, all instances are *new* to our seed model. So there is some degree of uncertainty when the model tries to make a local prediction. Not only considering the local prediction, but also considering how the same token is labeled globally, the *ComConf* measure seems to be a better choice in a cross-domain setting.

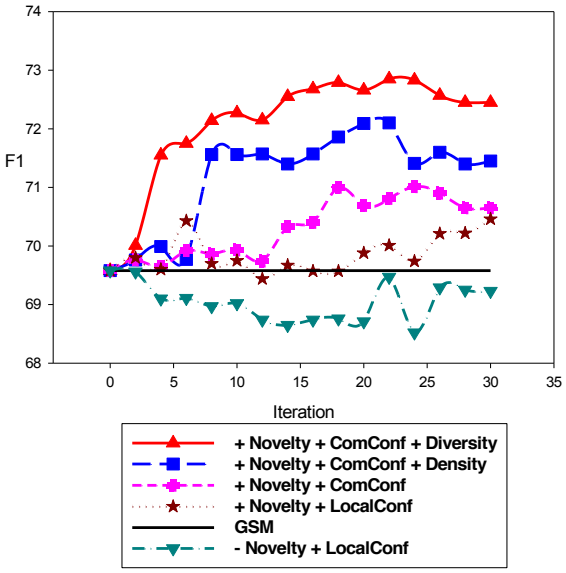


Figure 2: Performance of CDB. GSM stands for *generalized seed model* at iteration 0; + means with and – means without.

Regarding the density and the diversity measures, both of them further improve the performance. Density, however, does not perform well in the first 6 iterations. We checked the instances that had been selected during these iterations and found that many of them appear with very *strong* context words such as *Mr.*, *President*, *General* and *said*. They are considered representative instances according to our density measure. They can be regarded as cross-domain contexts which might have been learned by the generalized and un-generalized source domain models. In contrast, the diversity measure not only considers how representative an instance is but also prefers a certain degree of difference among the selected instances. Hence, the diversity measure has achieved the best result CDB could get so far. The best F score with the diversity measure is 72.85, a 7.16 improvement compared to the source model. The F score at the last iteration is 72.45, a 6.76 improvement compared to the source model.

We also run a standard bootstrapping procedure with the un-generalized seed model and with the same parameters used for the CDB procedure. The performance trends of using different instance selection criteria are similar to those of the CDB algorithm. The best F score, 70.12, is also obtained with the diversity measure. This further confirms that the multiple criteria proposed in this paper are better than a single criterion. CDB with generalized seed model outperformed the standard bootstrapping by more than 2 points which further indicates the usefulness of the combination of feature generalization and multi-criteria-based instance selection methods proposed in this paper.

6. Conclusion

We have described a general cross-domain bootstrapping algorithm for adapting a model trained only on a source domain to a target domain. We have improved the source model's F score by around 7 points. This is achieved without using any annotated data from the target domain and without explicitly encoding any target-domain-specific knowledge into our system. The improvement is largely due to the feature generalization of the source model with word clusters and the multi-criteria-based instance selection method.

Our immediate future work is to find a natural stopping criterion for the bootstrapping procedure, perhaps through the detection of *semantic drift* [28, 34]. Gazetteer resources have proven to be a powerful knowledge base for improving NER performance [9]. The only gazetteer in CDB now is a country and US state list. So another promising research avenue is to study how to automatically learn or mine a target domain named entity dictionary to further improve our system's performance.

7. ACKNOWLEDGMENTS

Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7058. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

8. REFERENCES

[1] D. M. Bikel, S. Miller, R. Schwartz and R. Weischedel. 1997. Nymble: a high performance learning name-finder. In *Proc. of ANLP*.

[2] J. Blitzer, R. McDonald, and F. Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proc. of EMNLP*.

[3] A. Borthwick, J. Sterling, E. Agichtein and R. Grishman. 1998. Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition. In *Proc. of Sixth WVLC*.

[4] A. Borthwick. 1999. A Maximum Entropy Approach to Named Entity Recognition. Ph.D. thesis, New York University.

[5] P. F. Brown, V. J. D. Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

[6] C. Chelba and A. Acero. 2004. Adaptation of maximum entropy capitalizer: Little data can help a lot. *In Proc. of EMNLP*, pages 285–292.

[7] H. L. Chieu and H. T. Ng. Named entity recognition with a maximum entropy approach. In *Proceedings of CoNLL-2003*.

[8] M. Ciaramita and Y. Altun. 2005. Named-entity recognition in novel domains with external lexical knowledge. In *Advances in Structured Learning for Text and Speech Processing Workshop*.

[9] W. W. Cohen and S. Sarawagi. 2004. Exploiting Dictionaries in Named Entity Extraction: Combining SemiMarkov Extraction Processes and Data Integration Methods. In *Proc. of KDD*.

[10] H. Daumé III and D. Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

[11] H. Daumé III. 2007. Frustratingly easy domain adaptation. In *Proc. of ACL 2007*.

[12] M. Dredze, J. Blitzer, P. Talukdar, K. Ganchev, J. Graca, and F. Pereira. 2007. Frustratingly Hard Domain Adaptation for Parsing. In *Proc. of CoNLL*.

[13] J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL 2005*.

[14] J. R. Finkel and C. D. Manning. 2009. Hierarchical bayesian domain adaptation. In *Proc. of NAACL*.

[15] R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proc. of HLT-NAACL*.

[16] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*.

[17] R. Florian; J. Pitrelli; S. Roukos; I. Zitouni. Improving Mention Detection Robustness to Noisy Input. In *Proc. of ENMLP 2010*.

[18] R. Grishman and B. Sundheim. Message Understanding Conference - 6: A Brief History. In *Proceedings of the COLING, 1996*.

[19] R. Grishman, D. Westbrook and A. Meyers. 2005. NYU's English ACE 2005 System Description. In *Proc. of ACE 2005 Evaluation Workshop*. Washington, US.

[20] R. Huang and E. Riloff. 2010. Inducing Domain-specific Semantic Class Taggers from (Almost) Nothing. In *Proc. of ACL*.

[21] H. Ji and R. Grishman. 2006. Data Selection in Semi-supervised Learning for Name Tagging. In *ACL 2006 Workshop on Information Extraction Beyond the Document*.

[22] J. Jiang and C. Zhai. 2006. Exploiting domain structure for named entity recognition. In *Proceedings of HLT-NAACL'06*.

[23] J. Jiang and C. Zhai. 2007. Instance weighting for domain adaptation in nlp. In *Proceedings of ACL*.

[24] V. Krishnan and C. D. Manning. 2006. An effective two stage model for exploiting non-local dependencies in named entity recognition. In *Proc. of ACL*.

[25] P. Liang. 2005. Semi-Supervised Learning for Natural Language. Master's thesis, Massachusetts Institute of Technology.

[26] D. Lin and X. Wu. 2009. Phrase Clustering for Discriminative Learning. In *Proceedings of the ACL and IJCNLP 2009*.

[27] A. McCallum, D. Freitag and F. Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Proc. of ICML*.

[28] T. McIntosh. 2010. Unsupervised discovery of negative categories in lexicon bootstrapping, In *Proc of EMNLP*.

[29] S. Miller, J. Guinness and A. Zamanian. 2004. Name Tagging with Word Clusters and Discriminative Training. In *Proc. of HLT-NAACL*.

[30] L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL-09*.

[31] E. Riloff and R. Jones. 1999. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. In *Proceedings of AAAI-99*.

[32] B. Roark and M. Bacchiani. 2003. Supervised and unsupervised PCFG adaptation to novel domains. In *Proc. of HLT-NAACL*, pages 126–133.

[33] D. Shen, J. Zhang, J. Su, G. Zhou, and C. Tan. 2004. Multi-criteria-based active learning for named entity recognition. In *Proceedings of ACL*.

[34] A. Sun and R. Grishman. 2010. Semi-supervised Semantic Pattern Discovery with Guidance from Unsupervised Pattern Clusters. *In Proc. of COLING*.

[35] E. Tjong and F. D. Meulder. 2003. Introduction to the conll-2003 shared task: Language independent named entity recognition. In *Proceedings of Conference on Computational Natural Language Learning*.

[36] J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.

[37] V. Vyas, P. Pantel and E. Crestan. 2009. Helping Editors Choose Better Seed Sets for Entity Expansion. In *Proceedings of CIKM-09*.

[38] Y. Wong and H. T. Ng. 2007. One Class per Named Entity: Exploiting Unlabeled Text for Named Entity Recognition. In *Proc. of IJCAI-07*.

[39] D. Wu, W. S. Lee, N. Ye, and H. L. Chieu. 2010. Domain adaptive bootstrapping for named entity recognition. In *Proc. of EMNLP*.

[40] R. Yangarber, W. Lin and R. Grishman. 2002. Unsupervised Learning of Generalized Names. In *Proc. of COLING*.

Semi-supervised Statistical Inference for Business Entities Extraction and Business Relations Discovery

Raymond Y.K. Lau and Wenping Zhang
Department of Information Systems
City University of Hong Kong
Hong Kong SAR
{raylau, wzhang}@cityu.edu.hk

ABSTRACT

The sheer volume of user-contributed data on the Internet has motivated organizations to explore the collective business intelligence (BI) for improving business decisions making. One common problem for BI extraction is to accurately identify the entities being referred to in user-contributed comments. Although named entity recognition (NER) tools are available to identify basic entities in texts, there are still challenging research problems such as co-reference resolution and the identification of abbreviations of organization names. The main contribution of this paper is the illustration of a novel semi-supervised method for the identification of business entities (e.g., companies), and hence to automatically construct business networks. Based on the automatically discovered business networks, financial analysts can then predict the business prestige of companies for better financial investment decision making. Initial experiments show that the proposed NER method for business entity identification is more effective than other baseline methods. Moreover, the proposed semi-supervised business relationship extraction method is more effective than the state-of-the-art supervised machine learning classifiers when there are not many training examples available. Our research work contributes to advance the computational methods for the extraction of entities and their relationships from texts.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Text Mining; I.2.6 [**Artificial Intelligence**]: Learning; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms

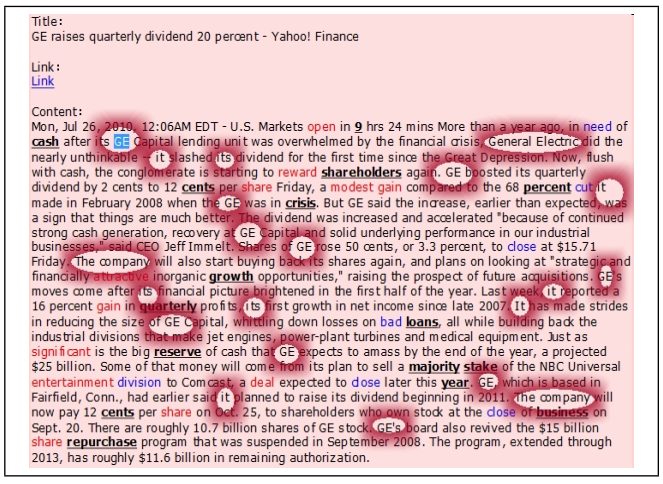
Algorithms, Performance, Experimentation

Keywords

Named Entity Recognition, Text Mining, Statistical Learning, Business Network Discovery

1. INTRODUCTION

The ubiquitous Web 2.0 applications have brought to organizations with unprecedented opportunities to exploit market intelligence and develop deep insights about their customers, business partners, and competitors [6, 7]. Due to the problem of information overload [13, 14], manual extraction of business intelligence from the flood of un-structured data coming from the Internet is not practical. Research effort has been devoted to the construction and analysis of social networks based on user-contributed data [25, 30], and hence to improve marketing effectiveness and reduce marketing costs [2, 17, 26]. Although some research work has been performed for the automatic construction and analysis of social networks [5, 25, 30], little work has been done for the discovery and analysis of business networks. Business networks can be seen as one kind of social network [17, 31]. The first step toward business network discovery is to identify the business entities referred to in the user-contributed free text. In fact, this ia a very challenging task because abbreviations and pronouns are often used to refer to business entities in natural languages.


Title:
GE raises quarterly dividend 20 percent - Yahoo! Finance

Link:
Link

Content:
Mon, Jul 26, 2010, 12:06AM EDT - U.S. Markets open in 9 hrs 24 mins More than a year ago, in need of cash after its GE Capital lending unit was overwhelmed by the financial crisis, General Electric did the nearly unthinkable -- it slashed its dividend for the first time since the Great Depression. Now, flush with cash, the conglomerate is starting to reward shareholders again. GE boosted its quarterly dividend by 2 cents to 12 cents per share Friday, a modest gain compared to the 68 percent cut it made in February 2008 when the GE was in crisis. But GE said the increase, earlier than expected, was a sign that things are much better. The dividend was increased and accelerated "because of continued strong cash generation, recovery at GE Capital and solid underlying performance in our industrial businesses," said CEO Jeff Immelt. Shares of GE rose 50 cents, or 3.3 percent, to close at $15.71 Friday. The company will also start buying back its shares again, and plans on looking at "strategic and financially attractive inorganic growth opportunities," raising the prospect of future acquisitions. GE's moves come after its financial picture brightened in the first half of the year. Last week, it reported a 16 percent gain in quarterly profits, its first growth in net income since late 2007. It has made strides in reducing the size of GE Capital, whittling down losses on bad loans, all while building back the industrial divisions that make jet engines, power-plant turbines and medical equipment. Just as significant is the big reserve of cash that GE expects to amass by the end of the year, a projected $25 billion. Some of that money will come from its plan to sell a majority stake of the NBC Universal entertainment division to Comcast, a deal expected to close later this year. GE, which is based in Fairfield, Conn., had earlier said it planned to raise its dividend beginning in 2011. The company will now pay 12 cents per share on Oct. 25, to shareholders who own stock at the close of business on Sept. 20. There are roughly 10.7 billion shares of GE stock. GE's board also revived the $15 billion share repurchase program that was suspended in September 2008. The program, extended through 2013, has roughly $11.6 billion in remaining authorization.

Figure 1: The Problem of Co-reference Resolution for Business Entity Identification

Figure 1 highlights the difficulties of business entity identification using the company "General Electric" as an example. As can be seen, "General Electric" is also referred to as GE (an abbreviation) or co-referenced by "it" or "the company". To deal with the first problem, we apply a semi-supervised learning method to automatically extract various abbreviations statistically associated with the particular business entity name in a training corpus. In addition, we propose a heuristic rule-based approach to deal with co-references

EOS, SIGIR 2011 workshop, July 28, Beijing, China

appearing in financial texts.

One of the main contributions of our research reported in this paper is the development of a semi-supervised statistical learning method for the identification of abbreviations of business entities in financial comments. The second contribution of our research work is the development of a novel algorithm, called proximity-based backward searching (PBS) to resolve the co-references of business entities. Finally, by successfully identifying business entities, we show our novel computational method for the automatic discovery of business networks. The main advantage of our proposed computational methods is that minimal human intervention is involved. As a result, the proposed method has better chance to be applied to real-world applications requiring the functionality of business entity identification.

2. RELATED RESEARCH WORK

There are two common approaches to conduct named entity recognition, namely, rule-based approach [4, 32] and learning-based approach [21, 29]. For rule-based approach, manually pre-defined heuristic or linguistic rules are applied to identify specific types of entities such as people, organizations, places, etc. [4, 32]. However, the main weakness of this approach is that the pre-defined rules may not be able to cover a variety of situations, particularly when they are applied to different domains. On the other hand, learning-based approach utilizes training corpora to automatically build classifiers to identify entities in unseen documents [21, 29]. This approach is quite flexible since it can be applied to different domains as long as the classifiers are re-trained using labeled training examples of a new domain. Nevertheless, the main weakness of this approach is the requirement of manually labeling a large number of training examples. In this paper, we propose a statistical inference-based method to address the low recall problem in NER while avoiding the time consuming process of labeling training examples.

Business relationship mining techniques can be broadly classified into content-based or link-based approach. In a previous study, the contents of online financial news were scanned to estimate the co-occurrence statistics of a pair of stock tickers [2]. The co-occurrence statistics were then used to predict the possible relationships among companies. Moreover, the membership of a company pertaining to an industry sector was estimated [2]. The CoMiner system made use of NLP techniques and several pre-defined syntactic patterns (e.g., company-A "versus" company-B) to identify competitive company relationships based on a Web corpus [1, 16]. Each syntactic pattern was assigned a weight and the Point-wise Mutual Information (PMI) measure was used to estimate the strength of competitive relationships between two companies. However, relationship mining according to pre-defined syntactic patterns may encounter the low recall problem since natural languages used in financial news are very flexible. One main innovation of the business relationship discovery method proposed in this paper is the development of a statistical inference based method to automatically expand some seeding syntactic patterns to address the low recall problem.

A link-based approach was developed to extract competitor relations from online financial news [18]. Company names identification was conducted based on stock tickers appearing in financial news. A weighted directed graph approach was adopted. If company y was mentioned in a financial news cataloged for company x, a directed business relationship from x to y is assumed. Twelve graph-based features such as in-degree, out-degree, total-degree, PageRank, etc. were taken as features and fed into four supervised classifiers. A similar link-based approach was also applied to predict the binary company revenue relations (e.g., whether company x has higher revenue than company y) based on online financial news [17].

A hybrid approach of both hyperlinks and contents of Web pages were exploited to discover hidden competitor relationships [20]. Both network-based and content-based features were used by supervised classifiers for competitive business relationships discovery. Instead of assuming that co-occurring companies in Web pages imply competitive relationship, our proposed approach examines different kinds of company relationships according to relational keywords found at the sentence level. The CopeOpi system was proposed to extract opinions and implicit business relationships of some targeting entities based on the Web corpus [10, 11]. Business associations among the target entities are discovered based on the opinion-tracking plots (i.e., the pattern of opinion scores exhibited over time).

The CoNet system employed shallow natural language processing (NLP) techniques to identify commercial entities and relationships from online financial news [31]. Simple linguistic rules were applied to identify the abbreviations of company names and resolve the co-references of company names. Commercial relationship mining was performed according to a set of pre-defined relationship keywords. However, such an approach may suffer from low recall due to the limited number of pre-defined relationship keywords. Latent semantic indexing (LSI) was also examined to identify relationships among entities of interests [3]. In particular, each entity was represented by an entity vector and mapped to the LSI space. For any two entities appearing in a document collection, the proximity of the corresponding entity vectors in the LSI space provides a quantitative measure of the degree of contextual association between the entities. The ArnetMiner system employed a supervised machine learning method, conditional random fields (CRF), to identify researcher and their relationships based on annotated Web pages [25]. Genetic algorithm was applied to discover communication networks based on the contents of email messages of the Enron corpus [30]. OpenCalais is a commercially available Web Service that employs NLP and machine learning techniques for NER, entity relationship identification, and event tagging (e.g., business events).[1] The Knowledge Base Population Track of the annual Text Analysis Conference (TAC) also explores novel methods for the identification of entity-linking [8].

3. THE COMPUTATIONAL MODELS

3.1 Business Entity Identification

The first step toward business relationship discovery is to identify the company names in the Web 2.0 documents (e.g., user-contributed financial comments). There are two main challenges of business name tagging. First, various abbreviations of a company name (e.g., "General Electronic" as "GE") should be identified. Second, the co-references (e.g., the company) related to a company name should be resolved.

[1] http://www.opencalais.com/documentation/

The procedures for business entity identification can be summarized as follows:

1. Extract business full names and stock ticker labels from Web sources (e.g., Yahoo! Finance);
2. Apply a mutual information based statistical learning method to extract abbreviations frequently co-occurring with the business full names or stock tickers in a training corpus;
3. Apply general business entity identification rules to identify the remaining organization names;
4. Apply the proximity-based backward searching algorithm to resolve the co-references in each document;

The business full names and the stock tickers of companies extracted from Yahoo! Finance are passed to our NER module to compose the basic business name dictionary for preliminary name identification. The general business entity identification rules are also passed to our NER module for the identification of business entities. For instance, the tokens with title cases and preceding "Inc", "Co. Ltd.", "Corp.", etc. are automatically labeled as organization names. To automatically extract the abbreviations of business entities, a mutual information based statistical learning method is proposed. The basic intuition of the proposed semi-supervised learning method is that a business name and its abbreviations tend to appear in adjacent sentences (as shown in Figure 1), and this kind of co-occurrence could appear frequently in a corpus.

More specifically, we extend the BMI measure [15], a variant of the standard mutual information (MI) measure [24], to conduct statistical inference for the abbreviations of business entities. The BMI measure has been successfully applied to context-sensitive text mining for IR [12] and automatic domain concept extraction in ontology discovery [15]. The distinct advantage of the BMI measure is that it can take into account both positive and negative evidence presented in text to infer the strength of association between two tokens. The BMI measure is extended to develop the proposed abbreviation extractor:

$$\begin{aligned} ae(t_i, t_j) = \quad & \alpha \times Pr(t_i, t_j) \log_2 \left(\frac{Pr(t_i, t_j)}{Pr(t_i) Pr(t_j)} \right) - \\ & (1-\alpha) \times [Pr(t_i, \neg t_j) \log_2 \left(\frac{Pr(t_i, \neg t_j)}{Pr(t_i) Pr(\neg t_j)} \right) + \\ & Pr(\neg t_i, t_j) \log_2 \left(\frac{Pr(\neg t_i, t_j)}{Pr(\neg t_i) Pr(t_j)} \right)] \end{aligned} \quad (1)$$

where abbreviation extractor $ae(t_i, t_j)$ is a function to "infer" the statistical association between two terms t_i and t_j. For our application, one of the terms is the seeding business full name. The parameter $\alpha \in [0, 1]$ was used to adjust the relative weight of positive and negative evidence respectively [15]. $Pr(t_i, t_j)$ is the joint probability that both terms appear in a text window, and $Pr(t_i)$ is the probability that a term t_i appears in a text window. According to previous studies in information retrieval and ontology discovery, a text window of 5 to 10 tokens is effective [12, 15]. This text window introduces a constraint (i.e., a proximity factor) to the inference process of business abbreviations such that only the tokens adjacent to the business full names or stock tickers are considered. Such a constraint aims at reducing the noise of the text mining process.

However, for business abbreviation extraction, the text window should be much larger since a business full name and its abbreviation tend to appear in adjacent sentences rather than occurring within the same sentence. Therefore, we set the window size of two paragraphs for this NER application. The probability $Pr(t_i)$ is estimated based on $\frac{|w_t|}{|w|}$ where $|w_t|$ is the number of text windows containing the term t and $|w|$ is the total number of text windows constructed from an unlabeled training corpus. Similarly, $Pr(t_i, t_j)$ is the fraction of the number of windows containing both terms out of the total number of text windows. Negation such as $Pr(t_i, \neg t_j)$ is interpreted in the way that t_i but not t_j appears in a text window. After computing the abbreviation extraction scores of the tokens with specific POS (e.g., no valid POS found) and specific properties (e.g., with title case and followed by "Inc", "Co. Ltd.", "Corp.", etc.), the top n tokens with the highest abbreviation extraction scores are selected as the abbreviations of the business full names. For co-reference resolution, the proximity-based backward searching algorithm depicted in Figure 2 is applied.

```
Algorithm ProximityBackSearch( d_in )
Input:  d_in   /* a document (i.e., financial news)
Output: d_out  /* a document with co-references resolved
Main Procedure:
1. d_out := d_in ;                          /* initialize the returned object
2. n := CountToken( d_out );                /* count # of tokens in document
3. FOR i := 1 to n
4.     t1 := ReadToken(i, d_out );
5.     IF Pos(t1) = "pn" or Pos(t1) = "company"   /* the POS of the token is pronoun
6.         j := i – 1;
7.         WHILE j >= 1 AND InWindow(i,j)   /* loop backward in document
8.             t2 := ReadToken(j, d_out );
9.             IF Type (t2) = Nil   /* not recognized entity
10.                j := j – 1;
11.            ELSE
12.                IF Type(t2) = "organization"   /* type of entity
13.                    d_out (i) := t2;   /* replace co-reference
14.                ENDIF
15.                j := 0                  /* quit backward search
16.            ENDIF
17.        END
18.    ENDIF
19. END
20. Return d_out ;
```

Figure 2: The Proximity-based Backward Searching Algorithm

Within each document d_{in}, all the business entities are first identified based on the company table (containing full names, stock tickers, and abbreviations). In addition, generic business entity identification rules (e.g., title cases followed by "Co. Ltd.") are also applied to identify business entities. The proximity-based backward search algorithm then locates the first ambiguous pronoun or token (e.g., "company") $t1 \in d_{in}$. Then, the algorithm works backward to search for the nearest business entity already identified by the previous NER procedure. The proximity is globally defined in advance; for our experiment reported in this paper, the proximity of two paragraphs are defined. If a nearest business entity is found and it is located within the two paragraphs boundary, the pronoun or ambiguous token is

replaced by the full name of the nearest business entity. The algorithm continues to process the next ambiguous pronoun in the document until no more ambiguous pronoun is found or the end of the document is encountered.

3.2 Business Relationship Mining

One unique feature of the proposed business relationship mining method is that it is based on an un-supervised statistical inference technique. The main advantage is that manually labeled training examples are not required for the proposed method, and hence it improves the chance of deploying the proposed methodology to support real-world applications. Similar to the ideas proposed by Turney and Littman [28] for automatic opinion indicators expansion, we develop a statistical inference method to automatically extract a set of domain-specific relationship indicators based on some seeding relationship indicators. In particular, 10 collaboration indicators (e.g., cooperate, ally, collaborate, joint, own, partner, coordinate, engage, partnership, agree) and other 10 competition indicators (e.g., compete, challenge, against, vie, contend, fight, contest, dispute, battle, accuse) were used as the seeding relationship indicators. The synonyms of these indicators are also extracted from WordNet [19] automatically. The initial set of relationship indicators is used by the statistical inference module to generate the relationship lexicon. Then, business relationship mining is conducted based on the system generated relationship lexicon. When compared to the method proposed by Turney and Littman [28], the improvements of our method includes using a shallow NLP method to identify specific POS for relationship indicators mining, and the application of WordNet to filter noisy relationship indicators generated by the mining process. Above all an enhanced mutual information based metric (Eq.1) is applied to identify additional relationship indicators statistically correlated with the seeding relationship indicators.

For business relationship mining, our system first utilizes the proposed NER method to identify the valid lexical patterns such as (organization, relation, organization) and (relation, organization, organization), or (organization, organization, relation). A proximity threshold ω_{prox} is applied to measure the proximity between a relationship indicator and a company name. If the distances (by words) between a relationship indicator and the respective companies are all below the proximity threshold ω_{prox}, a lexical pattern is considered a valid pattern. Then, each valid lexical pattern is classified as competitive or collaborative based on the relationship indicators captured in the relationship lexicon. The functions $Coll(x,y) = \frac{\text{Freq}_{coll}(x,y)}{\text{Freq}(x,y)}$ and $Comp(x,y) = \frac{\text{Freq}_{comp}(x,y)}{\text{Freq}(x,y)}$ are used to derive the collaborative and the competitive scores for each pair of companies (x,y) identified from the a financial news corpus. $\text{Freq}_{coll}(x,y)$ is the occurrence frequency of a collaborative relationship involving x and y; $\text{Freq}_{comp}(x,y)$ is the occurrence frequency of a competitive relationship involving x and y; $\text{Freq}(x,y)$ is the occurrence frequency of any recognized relationships involving x and y. To prune noisy business relationships, only the business relationship with $Coll(x,y)$ or $Comp(x,y)$ score greater than a relationship threshold ω_{Freq} will be extracted by our system. If $Coll(x,y) - Comp(x,y) > \omega_{gap}$ ($Comp(x,y) - Coll(x,y) > \omega_{gap}$) is true, the pair of companies are considered collaborative (competitive). The threshold ω_{gap} is used to filter significant collaborative (competitive) relationships. The set of significant relationships pertaining to each industry can then be used to build a business network for that industry.

We use Pajek[2], a shareware for graph plotting, to visualize the discovered business networks. Figure 3 shows a sample business network discovered by our system. The top financial institutions included in the 2010 Forbes 2,000 list (Banking industry) were used as the seeds, and the news collected from Reuters for the period from 2006 to 2009 were used as the source. Solid lines indicate collaborative relationships and dash lines indicate competitive relationships. It seems that the global banking sector operates in a collaborative rather than a competitive manner in general. Such a network helps a financial analyst or business manager quickly identify which company is the leader and which company is under keen competition in a specific business sector.

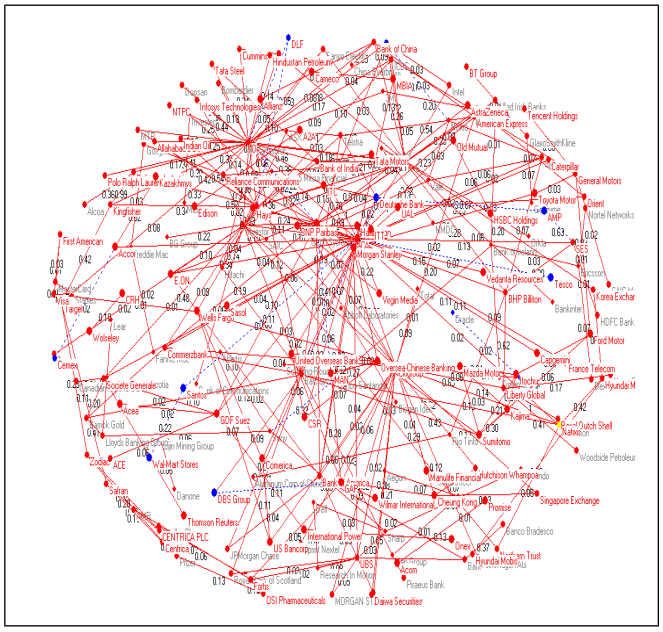


Figure 3: The Business Network Discovered Based on the Forbes Banking Industry

4. EXPERIMENTS AND RESULTS

Based on the financial news and company data collected from Reuters and Yahoo! Finance, the effectiveness of the proposed computational models for NER and business relationship mining were evaluated. Other baseline methods were also applied to carry out the same tasks. One of the main difficulties of evaluating the proposed computational models is the construction of a test dataset of reasonable size. It is quite labor-intensive to manually annotated companies and business relationships. A subset of companies from the 2010 Forbes 2,000 companies was used as our test cases. A total of 474,716 financial news were crawled from Reuters for the period from January 1, 2005 to December 31, 2009. A subset of the 2009 financial documents was manually annotated to build the evaluation dataset. For the evaluation of the proposed NER method, 314 financial news from among the downloaded financial news were annotated by two human annotators. When both human annotators

[2]http://pajek.imfm.si/doku.php?id=pajek

agreed on a business entity (i.e., a company), the particular token would be annotated and added to the evaluation dataset. There were 1,532 annotated business entities and all of them belonged to the Forbes 2,000 companies. For the evaluation of the proposed business relationship mining method, 2,074 sentences were annotated by other two human annotators. Similar to the annotation of business entities, when both annotators agreed on a business relationship, the annotated relationship would be added to the evaluation dataset. There were 839 collaborative relationships and 535 competitive relationships in the dataset. The evaluation measures such as Precision, Recall, and F-measure commonly used in information retrieval and opinion mining research were applied to our experiments [22].

For the first experiment, we examined the effectiveness of the proposed NER method (AE). The first baseline (BASE1) was the NER model developed based on the business full name and the generic business entity identification rules. In other words, automatic abbreviation extraction of business names (Eq.1) and co-reference resolution were not supported. The second baseline (BASE2) incorporate the proposed proximity-based backward searching co-reference resolution algorithm, but automatic automatic abbreviation extraction of business names was not supported in BASE1. The parameter $\alpha = 0.57$ was empirically established based on a subset of the evaluation dataset. The experimental results are reported in Table 1. It is obvious that the proposed NER method with semi-supervised abbreviation extraction and proximity-based backward searching co-reference resolution achieves the best recall while maintaining a comparable precision with the other baseline methods. AE outperforms BASE2 by 9.7 % in terms of recall and 4.5 % in terms of F-measure. As expected, the performance of the first baseline method is the worst in business entity identification because of the lack of capability of recognizing the abbreviations of business entities and resolving co-references in free text.

Table 1: Comparative Performance of Business Entity Identification

Method	Recall	Precision	F-measure	Accuracy
AE	0.898	0.839	0.868	0.885
BASE2	0.819	0.842	0.829	0.869
BASE1	0.631	0.881	0.735	0.852

For the second experiment, we examined the effectiveness of the proposed business relationship mining method. This task involves 3 classes (i.e., collaborative, competitive, neither). We assessed the performance of the collaborative relationship classification task and the competitive relationship classification task separately. The proposed statistical inference method (SI) (Eq.1) was used to construct the relationship lexicon for business relationship identification. One of the baseline systems (BASIC) used the 20 seeding relationship indicators alone. Both the SI and the BASIC methods employed the proposed business abbreviation and co-reference resolution methods. The second baseline method employed a state-of-the-art supervised classifier, the conditional random fields (CRF) classifier [9, 23] to label the sequences of company relationships. CRF is based on a discriminative probabilistic model (an undirected graph model) with each vertex representing a random variable, and the edge indicating the dependency between two random variables. The ultimate goal is to estimate the probability distribution of each vertex based on some training data. A publicly available Java-based implementation of the CRF classifier was used in this experiment.[3] Finally, the SVM-struct supervised classifier for sequence labeling was also applied [27].

Table 2: Comparative Performance of Business Relationship Classification

Classification of Collaborative Business Relationships				
Method	Recall	Precision	F-measure	Accuracy
SI	0.701	0.638	0.668	0.789
CRF	0.626	0.638	0.632	0.753
SVM-struct	0.618	0.633	0.625	0.747
BASIC	0.622	0.639	0.630	0.741
Classification of Competitive Business Relationships				
SI	0.688	0.639	0.663	0.775
CRF	0.631	0.656	0.643	0.756
SVM-struct	0.619	0.643	0.631	0.744
BASIC	0.625	0.654	0.639	0.747

For each kind of relationship classification, 70% of the annotated positive examples were used in the training set and the reaming 30% of the positive examples were used in the test set. The same number of negative examples was added to the training set and the test set, respectively. The results of our experiment were tabulated in Table 2. For both the collaborative and the competitive business relationship identification tasks, it is clear that the SI method outperforms the other baseline methods. More specifically, the SI method outperforms CRF and SVM-struct in terms of F-measure by 5.7%.and 6.9%, respectively for the collaborative relationship classification task. Surprisingly, both CRF and SVM-struct did not perform very well given a relatively small training set. However, the training processes for CRF and SVM-struct are realistic because a large number of labeled business relationships are rarely available in the real-world. The experimental results show that the proposed semi-supervised statistical inference method can be successfully applied to build a relationship lexicon, which eventually improves the performance of business relationship identification. The additional advantage of the proposed method is that labeled training examples are not required for business relationship mining.

5. CONCLUSIONS AND FUTURE WORK

Although some research work was performed for the automatic construction and analysis of social networks, little work has been done for the discovery and analysis of business networks. The main contributions of this paper include: (1) the development of a novel semi-supervised business entity recognition method; (2) the development of a novel semi-supervised business network discovery method; (3) the empirical evaluation of the proposed computational methods based on real-world datasets. Experimental results confirm that the proposed business entity identification and business network discovery methods are effective. More specifically, the proposed business relationship discovery method outperforms state-of-the-art supervised machine learning classifiers for the respective classification tasks. The distinct advantage of the proposed computational methods is that

[3]http://crf.sourceforge.net/

manually labeled training examples are not required. Accordingly, it facilitates the application of the proposed computational methods to real-world applications. Future work involves evaluating the effectiveness of the proposed methods using a larger dataset and comparing the performance of the proposed methods with other well-known systems. The properties of the discovered business networks will be examined to develop effective business prestige metrics to predict business performance.

6. REFERENCES

[1] Shenghua Bao, Rui Li, Yong Yu, and Yunbo Cao. Competitor mining with the web. *IEEE Transactions on Knowledge and Data Engineering*, 20(10):1297–1310, 2008.

[2] A. Bernstein, S. Clearwater, and F. Provost. The relational vector-space model and industry classification. In *Proceedings of Workshop on Learning Statistical Models from Relational Data*, 2003.

[3] R.B. Bradford. Relationship discovery in large text collections using latent semantic indexing. In *Proceedings of the Fourth Workshop on Link Analysis, Counterterrorism, and Security*, 2006.

[4] Indra Budi and Stephane Bressan. Application of association rules mining to named entity recognition and co-reference resolution for the indonesian language. *International Journal of BI and DM*, 2:426–446, December 23 2007.

[5] Chakrabarti and Faloutsos. Graph mining: Laws, generators, and algorithms. *CSURV: Computing Surveys*, 38, 2006. Article 2.

[6] H. Chen and D. Zimbra. Ai and opinion mining. *IEEE Intelligent Systems*, 3(25):74–76, 2010.

[7] Sanjiv Das and Mike Chen. Yahoo! for amazon: Sentiment extraction from small talk on the web. *Management Science*, 53(9):1375–1388, September 2007.

[8] N. Fernández, J.A. Fisteus, L. Sánchez, and E. Martín. Webtlab: A cooccurrence-based approach to kbp 2010 entity-linking task. In *Proceedings of the Second Text Analysis Conference*. NIST, 2010.

[9] Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. Efficient, feature-based, conditional random field parsing. In *Proceedings of the Annual Meeting of the ACL*, pages 959–967. ACL, 2008.

[10] Lun-Wei Ku, Hsiu-Wei Ho, and Hsin-Hsi Chen. Novel relationship discovery using opinions mined from the web. In *Proceedings of the 31 National Conference on AI*, pages 1357–1362. AAAI Press, 2006.

[11] Lun-Wei Ku, Hsiu-Wei Ho, and Hsin-Hsi Chen. Opinion mining and relationship discovery using copeopi opinion analysis system. *JASIST*, 60(7):1486–1503, 2009.

[12] R.Y.K. Lau. Context-Sensitive Text Mining and Belief Revision for Intelligent Information Retrieval on the Web. *Web Intelligence and Agent Systems An International Journal*, 1(3-4):1–22, 2003.

[13] R.Y.K. Lau, P. Bruza, and D. Song. Belief Revision for Adaptive Information Retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference*, pages 130–137, Sheffield, UK, July 25–29 2004. ACM.

[14] R.Y.K. Lau, P. Bruza, and D. Song. Towards a Belief Revision Based Adaptive and Context-Sensitive Information Retrieval System. *ACM Transactions on Information Systems*, 26(2):8.1–8.38, 2008.

[15] R.Y.K. Lau, D. Song, Y. Li, C.H. Cheung, and J.X. Hao. Towards A Fuzzy Domain Ontology Extraction Method for Adaptive e-Learning. *IEEE Transactions on Knowledge and Data Engineering*, 21(6):800–813, 2009.

[16] Rui Li, Shenghua Bao, Jin Wang, Yong Yu, and Yunbo Cao. Cominer: An effective algorithm for mining competitors from the web. In *ICDM*, pages 948–952. IEEE Computer Society, 2006.

[17] Z. Ma, O. Sheng, and G. Pant. Discovering company revenue relations from news: A network approach. *Decision Support Systems*, 4(47):408–414, 2009.

[18] Z. Ma, O. Sheng, and G. Pant. A network-based approach to mining competitor relationships from online news. In *Proceedings of the 2009 International Conference on Information Systems*, 2009. Article 59.

[19] G. A. Miller, Beckwith R., C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *Journal of Lexicography*, 3(4):234–244, 1990.

[20] G. Pant and O. Sheng. Avoiding the blind spots: Competitor identification using web text and linkage structure. In *Proceedings of the 2009 International Conference on Information Systems*, 2009. Article 57.

[21] J. Patrick, C. Whitelaw, and R. Munro. SLINERC: The sydney language-independent named entity recogniser and classifier. In *Proceedings of the Sixth Conference on Natural Language Learning*, pages 199–202, 2002.

[22] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, New York, 1983.

[23] Sunita Sarawagi. Efficient inference on sequence segmentation models. *Proceedings of the Twenty-Third International Conference on Machine Learning*, volume 148, pages 793–800. ACM, 2006.

[24] C. Shannon. A mathematical theory of communication. *Bell System Technology Journal*, 27:379–423, 1948.

[25] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. *Proceedings of the 14th ACM SIGKDD*, pages 990–998. ACM, 2008.

[26] M. Trusov, A. Bodapati, and R. Bucklin. Determining influential users in internet social networks. *Journal of Marketing Research*, 47:643–658, 2010.

[27] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[28] Peter D. Turney and Michael L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346, October 2003.

[29] N. G. Vincent. Semantic class induction and co-reference resolution. In *Proceedings of the 45th Annual Meeting of the ACL*, pages 536–543, 2007.

[30] Garnett Carl Wilson and Wolfgang Banzhaf. Discovery of email communication networks from the enron corpus with a genetic algorithm using social network analysis. In *IEEE Congress on Evolutionary Computation*, pages 3256–3263. IEEE, 2009.

[31] Y. Xia, N. Ji, W. Su, and Y. Liu. Mining commercial networks from online financial news. In *Proceedings of the 2010 IEEE International Conference on E-Business Engineering*, pages 17–23, 2010.

[32] G. Zhou and J. Su. A high-performance co-reference resolution system using a constraint-based multi-agent strategy. In *Proceedings of the 2004 International Conference on Computational Linguistics*, 2004. Article 522.

Unsupervised related entity finding

Olga Vechtomova
Department of Management Sciences
Faculty of Engineering
University of Waterloo
Waterloo, ON, Canada

ovechtom@uwaterloo.ca

ABSTRACT
We propose an approach to the retrieval of entities that have a specific relationship with the entity given in a query. An initial candidate list of entities, extracted from top ranked documents retrieved for the query, is refined using a number of statistical and linguistic methods. The proposed method extracts the category of the target entity from the query, identifies instances of this category as seed entities, and computes distributional similarity between candidate and seed entities.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Retrieval models.

General Terms
Algorithms, Design, Experimentation.

Keywords
Information retrieval, entity retrieval, related entity finding.

1. INTRODUCTION
Most information retrieval systems, including commercial search engines, respond to the user's query by retrieving documents. If a user is looking for entities that have a specific relationship to an entity already known to him, he has to manually find them in the documents retrieved by an IR system. Arguably, users with such information needs may prefer to use a system that retrieves entities, rather than documents, as this would eliminate the time-consuming process of reading through large amounts of text.

In this paper we propose a method for retrieving and ranking entities related to the entity given in the query by a specific relationship. The evaluation was done on the dataset of the Related Entity Finding (REF) task of the Entity track of TREC 2010 [1], as well as the "list" questions of the QA track of TREC 2005 [2]. The proposed approach is unsupervised and domain-independent, extracting entities from the texts of documents retrieved for the user's query. Our goal is to minimise the reliance on knowledge bases in the process.

The paper is organised as follows: Section 2 gives a detailed description of the proposed method, Section 3 presents evaluation, in Section 4 we analyse the effect of the major parameters on performance, in Section 5 we provide an overview of related work, and conclude the paper in Section 6.

2. METHODOLOGY
In the following subsections we describe our approach to entity finding in detail. Figure 1 provides an overview of the main components of the proposed method. In the first stage (rectangle 1), the system retrieves an initial set of documents for the query from the Web. Only the sentences containing query terms plus one preceding/following sentence are retained. Named Entity tagging is applied to these sentences, and candidate entities are extracted and ranked. In the second stage, the target category name is automatically identified from the topic narrative. In stage 3, the system finds hyponyms of this category name, and selects seed entities from the hyponyms. In stage 4, the entities (candidates and seeds) are represented as vectors of weighted grammatical dependency triples, and pairwise (candidate-seed) similarity is calculated. In stage 5, candidate entities are ranked by similarity to all seeds. Stage 1 is described in Section 2.1, while stages 2-5 are presented in Section 2.2.

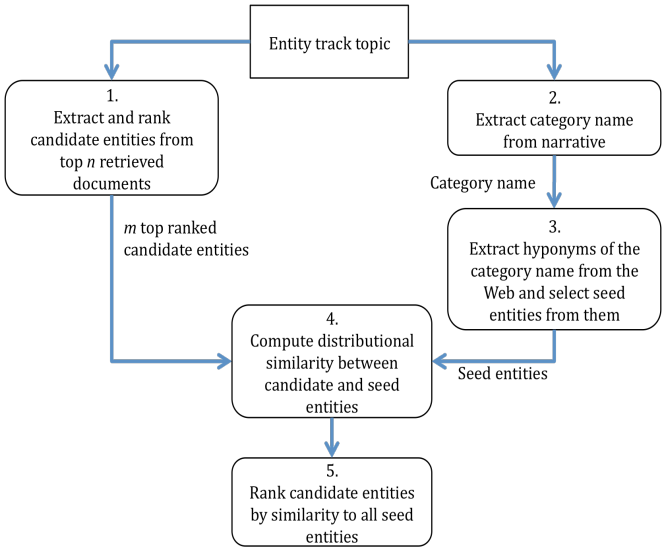


Figure 1. Components of the proposed method.

2.1 Extracting candidate entities
The components of the Entity track topics include the name of the entity known to the user (topic entity), the document ID of its homepage, the type of the sought (target) entities, which can be "organization", "person", "location" or "product", and a one-sentence narrative describing the relationship between the topic entity and the target entities. An example of a topic from the Entity track REF task of TREC 2010 is given in Figure 2.

```
<num>23</num>
<entity_name>The Kingston Trio</entity_name>
<entity_URL>clueweb09-en0009-81-29533</entity_URL>
<target_entity>organization</target_entity>
<narrative>What recording companies now sell the
Kingston Trio's songs? </narrative>
```

Figure 2. Entity track topic example.

EOS, SIGIR 2011 workshop, July 28, Beijing, China

As the first step, queries to retrieve top documents from the Web are generated from the "entity name" and "narrative" sections of the topics. The objective of the algorithm is to extract named entities and other noun phrases from the topic. For this purpose we use a Part-Of-Speech tagger [3], a Noun Phrase chunker [4], and a list of titles of Wikipedia pages. The algorithm is described in detail in [5]. The resulting queries are then used to retrieve the top 50 documents from a Web search engine. We did not evaluate alternative values for the number of top documents retrieved. Our motivation to use 50 is to keep the number of documents for subsequent in-depth analysis reasonably small, and at the same time have sufficient amount of text to extract entities from.

The retrieved documents are parsed to remove HTML tags, script and style sections, and broken into sentences. We then extract sentences that contain at least one query term. If a query term is a noun, the system attempts to match its singular and plural forms. For each such sentence, we also extract one sentence before and one after. The sentences are then processed by the LBJ-based Named Entity Recognizer [6]. The NER tagger only assigns the labels of "Location", "Organization", "Person" and "Miscellaneous". For topics with the target entity type of "organization", "person" and "location", we extract all entities tagged with the corresponding NER labels, while for topics of category "Product" we extract entities labelled as "Organization" and "Miscellaneous". After candidate entities are extracted, they are ranked by *TF*IDF*, where *TF* is the number of times the entity occurs in the 50 retrieved documents, while *IDF* is calculated using the number of documents containing the entity in ClueWeb09 Category B corpus.

2.2 Ranking candidate entities by the similarity to the target entity category

Since the chosen NER tagger can only be used to identify entities of few broad categories, such as organisations and people, the list of candidate entities can be noisy. This is further compounded by the NER tagger errors. To refine the list of entities, we apply the distributional similarity principle, which is based on the observation that semantically close words occur in similar contexts. If we have a small number of correct seed entities, we can rank the candidate entities by the distributional similarity to them. There are a number of semi-supervised methods (e.g., [7]) that use a small set of seed words to find other words that occur in similar contexts, and therefore, are likely to be semantically similar. The problem in our task is that the seed words are not given. However, the topic narratives have descriptions of the categories of entities that are to be retrieved. Our approach is to find seed entities based on the described categories. We developed a method to extract the category name from the narrative, e.g., "recording companies" from the topic in Figure 2, and adapted Hearst's method for the automatic acquisition of the hyponymy relation [8] to find entities that belong to this category. Seed entities are then selected from the hyponyms. We also developed a new method for computing the distributional similarity between seeds and candidate entities using BM25 with query weights [9], and ranking the entities by similarity to all seed entities.

The stages presented in Figure 1 as rectangles 2-5 are described in this section. As an input to stage 4, we use top *m* entities ranked by *TF*IDF* in stage 1. This set of entities will be subsequently referred to as "candidate entities". The value of *m* was determined to be 200 based on the training dataset (REF task of the Entity track in TREC 2009).

2.2.1 Extracting category names from topic narratives

To extract category names (stage 2 in Figure 1), the narratives are first processed using Brill's Part-of-Speech (POS) tagger [3] and a Noun-Phrase chunker [4]. Then a set of rules is applied to select one of the initial noun phrases (NPs) from the narrative. Commonly, the first noun phrase in the narrative is the correct category name, for example "recording companies", extracted from the following POS-tagged and NP-chunked narrative: "[What/WP] [recording/NN companies/NNS] now/RB sell/VBP [the/DT Kingston/NNP Trio's/NNP songs/NNS] ?/.".

2.2.2 Identifying seed entities

After the category name is identified, the next step is to find entities that belong to this category. We adapted the unsupervised hyponymy acquisition method proposed by Hearst [8]. Hearst's method consists of using six domain- and genre-independent lexico-syntactic templates that indicate a hyponymy relation. For each topic, six queries are constructed using these templates and the category name extracted from the topic narrative. For example, the query for template "*NP such as {NP,}* {(or|and)} NP*" and category name "recording companies" is: "recording companies such as". Each query is submitted to a commercial search engine as a phrase (i.e. quote-delimited). If the total number of pages retrieved by all six queries is fewer than 10, the first word in the category name is dropped and the search is repeated. If again it returned fewer than 10 pages, the first two words are dropped, and so on until at least 10 pages are retrieved, or the remaining category name is a unigram, in which case we use however many pages were found. Also, if a category name is a unigram, the query includes the topic title in addition to the template, in order to minimise the extraction of unrelated entities.

The documents retrieved for each query are processed to remove HTML tags, and split into sentences. The sentences containing the hyponymy lexico-syntactic patterns are then processed using the LBJ-based NER tagger [6]. Depending on the expected position of hyponyms in the lexico-syntactic pattern, NEs either immediately preceding, or following the pattern are extracted. If several NEs are used conjunctively, i.e., separated by a comma, "and" or "or", all of them are extracted. For each topic, we extract only NEs with the tags corresponding to the target entity type specified in the topic. An example of text retrieved for the above query and processed by the NER tagger is: "In large recording companies such as [ORG EMI], the mastering process was usually controlled by specialist staff...". The entity "EMI" is extracted as hyponym from this sentence as it has the correct entity type ("organization") for this topic.

One problem with using all found hyponyms as seed entities is that they can be unrelated to the topic. We need to ensure that we use only those hyponyms, for which there exists some evidence of relationship to the topic. For this purpose, we defined as seeds the intersection of found hyponyms and entities extracted from the top 50 documents retrieved for the initial query as described in Section 2.1. For example, for the above topic, the following hyponyms were identified as seeds: "Warner Bros", "Decca", "Columbia Records", "Capitol Records", "Bear Family Records". If only one seed word is identified as a result of this process, we do not perform entity re-ranking on this topic, and keep the original *TF*IDF* ranking order.

2.2.3 Computing distributional similarity between candidate and seed entities

Distributional similarity between entities is computed based on the commonality of their contexts of occurrence in text. In their simplest form, contexts could be words extracted from windows around entity occurrences. Alternatively, they could be grammatical dependency relations, with which an entity occurs in text. The use of grammatical dependency relations is more constraining in calculating entity similarity, and allows us to identify tightly related entities, which could be inter-substituted in a sentence without making it illogical and ungrammatical. In contrast if we only use co-occurring words in calculating similarity, we would get more loosely related entities. Several previous approaches to calculating distributional similarity between words use grammatical dependency relations, e.g., [10]. Since we are interested in identifying entities that are of the same semantic category as the seed words, we also use grammatical dependency relations.

For each seed and candidate entity we retrieve 200 documents from ClueWeb09 Category B using BM25 [9] implemented in Wumpus[1] search engine. Each document is split into sentences, and sentences containing the entity are parsed using Minipar[2] syntactic parser to extract grammatical dependency triples. Each dependency triple (e.g., "release *V:subj:N* Capitol Records") consists of two words/phrases and a grammatical relation that connects them. The dependency triples are transformed into features representing the context of each candidate and seed entity. To transform a triple into a feature, we replace the entity name in the triple with 'X', e.g., "release V:subj:N Capitol Records" is transformed into "release V:subj:N X". To avoid using features that are specific to one or few seed entities, only features that occur with at least 50% of all seed entities are used in computing entity similarity. For each seed and candidate entities we build a vector consisting of these features and their frequencies of occurrence with this entity.

In order to compute the similarity between the vectors of seed and candidate entities, we adapted BM25 with query weights formula, QACW (Query Adjusted Combined Weight) [9]. QACW is calculated for each seed and candidate entity combination. In the formula, the vector of the seed entity is treated as the query and the vector of the candidate as the document:

$$QACW_{c,s} = \sum_{f=1}^{F} \frac{TF(k_1+1)}{K+TF} \cdot QTF \cdot IDF_f \qquad (1)$$

Where: F – the number of features that a candidate entity c and a seed entity s have in common; TF – frequency of feature f in the vector of candidate entity; QTF – frequency of feature f in the vector of the seed entity; $K = k_1 \times ((1-b)+b \times DL/AVDL)$; k_1 – feature frequency normalisation factor; b – vector length normalisation factor; DL – number of features in the vector of the candidate entity; $AVDL$ – average number of features in all candidate entities.

We evaluated different combinations of b and k_1 values on the 20 topics from the Entity track of TREC 2009, with the best results in NDCG@R obtained with b=0.8 and k_1=0.8.

In order to calculate the *IDF* of a feature, we need to have access to a large syntactically parsed corpus, such as ClueWeb09 Category B. Since we do not have such a resource, and it is computationally demanding to produce one, we approximate *IDF* of a feature with the *IDF* of its component word. For example, for the feature "release V:subj:N X" we calculate the *IDF* of "release" by using its document frequency in the ClueWeb09 Category B collection.

Arguably, when calculating the similarity of candidate entities to seed entities, we should take into account how strongly each seed is associated with the original TREC topic. Candidate entities similar to the seeds, which have weak association with the topic, should be downweighted compared to those candidate entities, which are similar to seeds strongly associated with the topic. We propose to quantify this association by using *TF*IDF* entity weights calculated in the first stage of the method (Section 2.1). Thus, the matching score of a candidate entity with all seeds is calculated according to:

$$EntitySeedBM25_c = \sum_{s=1}^{S} w_s QACW_{c,s} \qquad (2)$$

Where: w_s – *TF*IDF* weight of the seed entity s.

Only those candidate entities that have EntitySeedBM25 greater than zero are retained. The final ranking of entities is achieved through a linear combination of *TF*IDF* and *EntitySeedBM25* according to the following equation:

$$TFIDFEntitySeedBM25 = \\ = \alpha \times \log(TFIDF) + (1-\alpha) \times \log(EntitySeedBM25) \qquad (3)$$

Values from 0.1 to 1 at 0.1 intervals were evaluated for α on the 20 topics from the Entity track of TREC 2009, with the best results in NDCG@R obtained with α=0.5.

3. EVALUATION

Our methods were evaluated on the dataset of the Related Entity Finding (REF) task of the Entity track of TREC 2010 [1] and on the "list" questions from the Question Answering (QA) track of TREC 2005 [2]. All parameters were tuned on the 20 topics from the REF task of the Entity track 2009.

3.1 Evaluation on the REF task of the Entity track of TREC 2010

The requirement in the REF task of the Entity track is to retrieve a ranked list of up to 100 entities for each topic. For each retrieved entity, the systems are required to retrieve one homepage, which must be represented as the ClueWeb09 Category A document ID.

Relevance judgements of entity homepages were done on a 3-point scale: 2 – primary page (i.e. homepage of the correct entity), 1 – descriptive page related to the correct entity, and 0 – all other pages. The two official evaluation measures are nDCG@R – normalised discounted cumulative gain at R, where R is the number of primary and relevant homepages for that topic, and P@10 – fraction of primary homepages among the documents retrieved for the top 10 entities. The Mean Average Precision (MAP) and Precision at R were also calculated for TREC 2010 topics[3]. In order to find homepages of entities, we developed a simple algorithm, which consists of retrieving the top 10 webpages for each entity from a commercial Web search engine, filtering out a small number of common URLs, such as "dictionary.com", "facebook.com" and "wikipedia.org", and using as homepage the top ranked page that also exists in the ClueWeb09 Category A collection. The evaluation procedure was the same for both training and test topics. The evaluation results

[1] www.wumpus-search.org

[2] http://webdocs.cs.ualberta.ca/~lindek/minipar.htm

[3] The evaluation script provided in TREC 2009 only calculates NDCG@R and P@10.

on the 20 training topics are given in Table 1, while the results on the 50 test topics are shown in Table 2. TFIDF is the baseline system that ranks entities by *TF*IDF* (Section 2.1), while TFIDFEntity-SeedBM25 is the system that uses distributional similarity of entities to seeds in entity ranking (Equation 3 in Section 2.2.3).

Table 1. Evaluation results on 20 REF training topics.

Run	nDCG@R	P@10	Rel. retr.	Prim. Retr.
TFIDF	0.1712	0.1450	86	63
TFIDFEntity SeedBM25	0.1705	0.1700	85	62

Table 2. Evaluation results on 50 REF test topics.

Run	nDCG @R	P@10	MAP	R-prec	Rel. retr.	Prim. Retr.
TFIDF	0.1226	0.0936	0.0588	0.1006	89	152
TFIDFEntity SeedBM25	0.1400‡	0.1043	0.0722‡	0.1140	91	157

3.2 Evaluation on the "list" questions from the QA track of TREC 2005

The "list" type of questions in the QA track of TREC 2005 are formulated differently from the Entity track REF topics. For each topic a target entity is specified, which is similar to the entity_name part of Entity track topics. Each topic has one or two list questions, formulated in a similar way as the narrative section of the Entity track topics. A major difference of the QA list questions from the Entity track REF topics is that target entity types are not given. Also, some list questions are looking for answers of types other than "Person", "Organization", "Location" and "Product". For our evaluation we only selected questions seeking entities of the above four types, as other types do not necessarily fall under the definition of an entity accepted in the Entity track, i.e. something that has a homepage. We also manually added target entity types (i.e., "Location", "Product", "Person" or "Organization") to make the questions conform to the Entity track topic format. In total, we used 74 out of 93 list questions in the QA 2005 dataset.

The official evaluation methodology for the list questions in the QA track required the participating sites to submit an unordered set of answer–documentID pairs, where answer is the entity string and documentID is the ID of a document supporting the entity as an answer. The document collection in the official QA track evaluation was AQUAINT. The evaluation measure was an F-measure, computed as F=(2*IP*IR)/(IP+IR), where Instance Recall (IR) is the number of distinct instances (entities) judged correct and supported by a document out of the total number of known correct and supported instances, and Instance Precision (IP) is the number of distinct instances judged correct and supported by a document out of the total number of instances returned. It is, however, not possible to use this evaluation methodology post-TREC since the number of judged supporting documents is very limited. In order to allow researchers to perform post-TREC evaluations, the track organisers released sets of patterns representing the correct answer strings extracted from the answer pool. The set contains only one pattern representing each correct answer, and takes the form of a regular expression, such as "(Holland|Netherlands)". Two major limitations of this pattern set are: it only contains correct answers from the pool, and may therefore be incomplete for some topics, and, secondly, the patterns themselves may not exhaustively cover all spelling and lexical variations of answers.

The evaluation reported in this section was performed using these patterns. F-measure as well as standard evaluation measures used in the Entity track of TREC 2010 were calculated. Since supporting documents are not used, Instance Recall is re-defined as the number of distinct instances that match patterns out of the total number of patterns for the question, and Instance Precision as the number of distinct instances that match patterns out of the total number of instances returned. Each pattern can only be matched once, in other words, any repeated matches on the same pattern are ignored. The results are summarised in Table 3.

Table 3. Evaluation results on 74 QA 2005 list questions.

Run	nDCG @R	P@10	MAP	R-prec	Rel. retr.	F-measure
TFIDF	0.1469	0.1432	0.1241	0.1362	349	0.0831
EntitySeed BM25	0.1561	0.1473	0.1299	0.1440	359	0.0854

4. DISCUSSION AND ANALYSIS

A major contribution of the proposed method is automatic identification of seed entities based on the category name extracted from the topic narrative. Most automatically identified seeds are not the correct answers. Table 4 shows the statistics for the 74 list questions in the QA 2005 dataset.

Table 4. Statistics for 74 QA 2005 list questions.

	seeds	correct answers	seeds ∩ correct answers
Total	1269	1005	217
Mean	17.15	13.58	2.93
Median	3.5	10.5	0

One question that we would like to investigate is how the performance of automatically identified seeds compares to the performance of the correct answers used as seeds. In order to do this, we performed (a) runs with different numbers of correct answers as seeds, and (b) runs with varying proportion of correct and incorrect answers used as seeds. The correct answers were randomly selected from the list of correct answer patterns (see Section 3.2), while as incorrect answers we used randomly selected automatically found seeds that are not in the list of correct answer patterns.

Another parameter that merits further analysis is the minimum number of seeds that a feature has to co-occur with to be considered in the computation of distributional similarity between a seed and a candidate entity. In the runs reported above we use only those features that co-occur with at least 50% of all seeds (Section 2.2.3). This may pose a problem if the number of seeds is large, which will mean that only few or even no features may satisfy this condition. In this section we report a systematic evaluation of different values for this parameter, referred to as "seed co-occurrence threshold".

Figure 3 shows the effect of the number of correct answers as seeds and the seed co-occurrence threshold on nDCG@R for 74 list questions in QA 2005. Each data series in the graph represents a different co-occurrence threshold and the X-axis represents the maximum number of correct answers used as seeds. The performance of the TFIDF system (Table 3) is also shown for reference here. Each run only uses up to *n* number of correct

‡ statistically significant improvement over TFIDF run at 0.01 level (2-tail paired t-test)

answers as seeds, where *n* values range from 4 to 30 in the increments of 2. The seed co-occurrence threshold values range from 2 to 16 in the increments of 2. All other parameters are kept the same as in TFIDFEntitySeedBM25 run reported in Table 3.

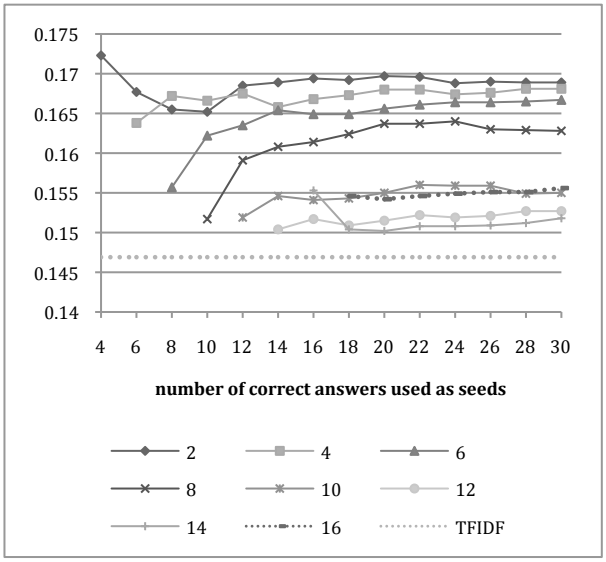


Figure 3. Effect of the number of correct answers as seeds and seed co-occurrence thresholds on nDCG@R (QA 2005).

The best performance in nDCG@R (0.1723) is achieved with 4 correct answers as seeds and seed co-occurrence threshold of 2. Interestingly, the co-occurrence threshold of 2 gives consistently good performance regardless of the number of seeds used. Another somewhat unexpected result is that the number of correct answers used as seeds does not seem to affect performance much. The highest nDGC@R is achieved with 4 and 20 seeds, used with the co-occurrence threshold of 2. Performance tends to increase a little with the initial increase in the number of seeds for runs using seed co-occurrence thresholds from 4 through 12, but then reaches a plateau at around 20 seeds. The plateau effect could be explained by the fact that only few topics have a large number of correct answers. As can be seen from Figure 4, only 13 out of 74 topics have 20 or more correct answers.

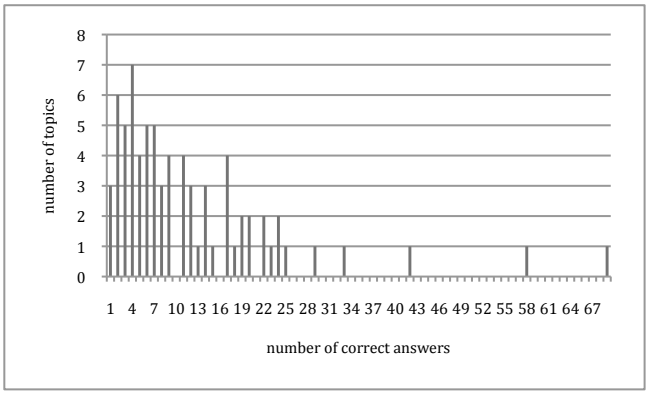


Figure 4. Distribution of correct answers for the QA 2005 list questions.

Next, we analyse how the presence of incorrect answers affects performance. We take *n* correct answers as seeds, and add to them *m* automatically identified seeds, which are not in the set of correct answers. The *n* is set to 4 and 20, which showed best results; for *m* we test values from 0 to 40 in the increments of 2. The results are given in Figure 5. Also plotted are the results for *n*=0. The TFIDF system is also shown for reference. Surprisingly, the best nDCG@R performance (0.1790) is achieved with 1 automatically identified seed added to 4 correct answers. This is 3.9% better than using only 4 correct answers as seeds. Similarly, adding 1 and 2 seeds to 20 correct answers leads to performance gains. This suggests that while it is useful to have correct answers as seeds, a small number of incorrect answers is not detrimental, and can even lead to small improvements. Also, adding larger number of incorrect answers has only a minor negative effect.

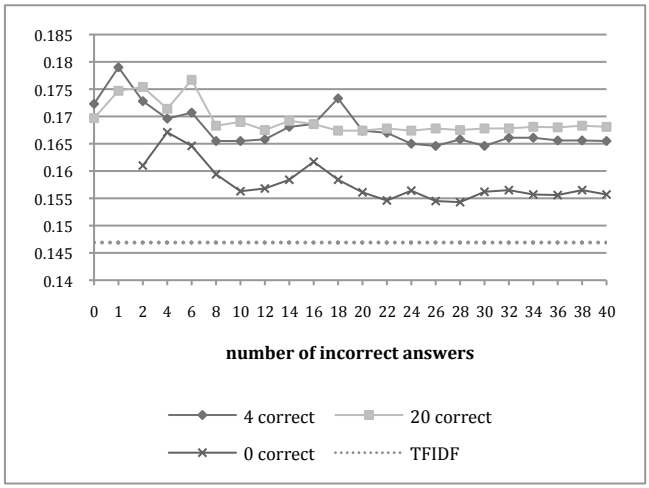


Figure 5. Effect of the number of incorrect answers on nDCG@R (QA 2005).

5. RELATED WORK

In this section we review some of the approaches to related entity finding in the Entity track of TREC. Most of the methods developed by participants of the Entity track start with the retrieval of some units of information (documents, passages, sentences) in response to the queries generated from the topic. The retrieved units are then used for extracting candidate entities. Below we discuss various approaches based on: (a) how the queries are constructed, (b) what units are retrieved (e.g., documents, passages, sentences), (c) how candidate entities are extracted and ranked.

5.1.1 Query construction

As an alternative to using "entity name" and "narrative" sections of topics as queries directly, some query structuring and expansion methods are explored. Vydiswaran et al. [11] model the information need as a triple (topic entity; relationship type; target entity), of which the first two are known. The words denoting the relationship are extracted from the narrative and expanded with WordNet synonyms. Fang et al. [12] expand the entities from narratives with their acronyms identified using dictionaries.

5.1.2 Retrieval units

Most approaches start with the retrieval of documents by using experimental IR systems and/or web search engines. For example, [11] use documents retrieved by Indri, from which they select snippets containing the query terms. McCreadie et al. [13] use the Divergence from Randomness model, a term proximity-based model, and the number of incoming links to the documents. Zhai et al. [14] use BM25, Fang et al. [12] use Google results filtered by the ClueWeb09 Category B documents, and Wu and Kashioka [15] compare the use of Indri and Google.

5.1.3 Candidate entity extraction and ranking

Vydiswaran et al. [11] extract candidate entities from the document snippets containing query terms, and rank them by a combination of the frequency of candidate entities in the retrieved snippets and their co-occurrence with the topic entity. McCreadie et al. [13] use DBPedia and US Census data to build representations of entities found in the ClueWeb09 Cat. B collection. Each entity representation includes alternative names (DBPedia aliases), DBPedia categories and documents in ClueWeb09 Cat. B containing the entity. They propose a voting model to rank entities. Zhai et al. [14] use titles and anchor texts in the retrieved documents as candidate entities. For each candidate entity a pseudo-document is built, consisting of top 100 sentences containing this entity. They experiment with ranking entities based on the similarity of their pseudo-documents and the pseudo-documents of the topic entity. Wu and Kashioka [15] use Wikipedia's hyperlink structure, to reduce the list of candidate entities. Entity scores are calculated based on the presence of hyperlinks between the Wikipedia pages of the candidate entity and the topic entity. They then retrieve snippets containing each candidate entity, and calculate a similarity score between the set of snippets and the topic entity, experimenting with a language modelling approach and Support Vector Machines. Kaptein et al. [16] calculate similarity between the topic entity and each candidate entity based on the co-citation information from the hyperlink graph constructed for the ClueWeb09 Cat. B collection. They also propose a method that extracts candidate entities from Wikipedia. Fang et al. [12] combine a number of approaches for ranking candidate entities, such as extracting entities from tables and lists in the retrieved web documents and using proximity in retrieved documents between a candidate and topic entities. One other method consists of extracting the first term from the narrative, which usually represents the category of the sought entities, and checking for each candidate entity if it occurs in the body or categories of its Wikipedia page. Bron et al. [17] select entities co-occurring with the topic entity, and propose a co-occurrence language model based on the contexts in which a candidate co-occurs with the topic entity.

6. CONCLUSION

We proposed an approach to finding related entities which relies primarily on statistical and linguistic methods. The approach was evaluated using the Entity track dataset of TREC 2010, as well as the QA track "list" questions from TREC 2005. Candidate entities are extracted using a NER tagger from documents retrieved in response to the query. As a separate step, target entity category names are automatically extracted from topic narratives, and are used to extract seed entities, i.e. entities that are likely to belong to this category. Top *m* entities ranked by TF*IDF are then re-ranked by their distributional similarity to the seeds. We developed a method for ranking candidate entities by the similarity to all seeds, whereby seeds are weighted by the strength of association with the topic. For computing the pairwise similarity between the vectors of the seed and candidate entities, we adapted BM25 with query weights. Evaluation results show that re-ranking of candidates by their similarity to seeds is effective, with some improvements being statistically significant.

7. REFERENCES

[1] Balog K., Serdyukov P., de Vries A.P. (2010) Overview of the TREC 2010 Entity Track. In Proc. of TREC 2010.

[2] Voorhees E.M. and Dang H.T. (2005) Overview of the TREC 2005 Question Answering Track. In Proc. of TREC 2005.

[3] Brill E. (1995) Transformation-based error-driven learning and natural language processing: a case study in part of speech tagging. Computational Linguistics, 21(4), 543-565.

[4] Ramshaw L. and Marcus M. (1995) Text Chunking Using Transformation-Based Learning. In Proc. of the Third ACL Workshop on Very Large Corpora, MIT.

[5] Vechtomova O. (2010) Related Entity Finding: University of Waterloo at TREC 2010 Entity Track. In Proc. of TREC 2010.

[6] Ratinov L. and Roth D. (2009) Design Challenges and Misconceptions in Named Entity Recognition. In Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL).

[7] Thelen, M. and Riloff E. (2002) A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In Proc. of EMNLP 2002.

[8] Hearst M. A. (1992) Automatic acquisition of hyponyms from large text corpora. In Proc. of the 14th Conference on Computational Linguistics, 539-545.

[9] Spärck Jones, K., Walker, S., & Robertson, S. E. (2000). A probabilistic model of information retrieval: Development and comparative experiments. Information Processing and Management, 36(6), 779–808 (Part 1); 809–840 (Part 2).

[10] Lin, D. (1998) Automatic retrieval and clustering of similar words. In Proc. of the 17th international Conference on Computational Linguistics, 768-774.

[11] Vinod Vydiswaran V.G., Ganesan K., Lv Y., He J., Zhai C.X. (2009) Finding Related Entities by Retrieving Relations: UIUC at TREC 2009 Entity Track. In Proc. of TREC 2009.

[12] Fang Y., Si L., Yu Z., Xian Y., Xu Y. (2009) Entity Retrieval with Hierarchical Relevance Model, Exploiting the Structure of Tables and Learning Homepage Classifiers. In Proc. of TREC 2009.

[13] McCreadie R., Macdonald C., Ounis I., Peng J., Santos R.L.T. (2009) University of Glasgow at TREC 2009: Experiments with Terrier. In Proc. of TREC 2009.

[14] Zhai H., Cheng X., Guo J., Xu H., Liu Y. (2009) A Novel Framework for Related Entities Finding: ICTNET at TREC 2009 Entity Track. In Proc. of TREC 2009.

[15] Wu Y., Kashioka H. (2009) NiCT at TREC 2009: Employing Three Models for Entity Ranking Track. In Proc. of TREC 2009.

[16] Kaptein R., Koolen M. and Kamps J. (2009) Result Diversity and Entity Ranking Experiments: Anchors, Links, Text and Wikipedia. In Proc. of TREC 2009.

[17] Bron M., He J., Hofmann K., Meij E., de Rijke M., Tsagkias M., and Weerkamp W. (2010) The University of Amsterdam at TREC 2010: Session, Entity and Relevance Feedback. In Proc. of TREC 2010.

Learning to Rank Homepages For Researcher-Name Queries

Sujatha Das*, Prasenjit Mitra†, C. Lee Giles†
*Department of Computer Science and Engineering
†College of Information Science and Technology
The Pennsylvania State University
gsdas@cse.psu.edu, pmitra@ist.psu.edu, giles@ist.psu.edu

ABSTRACT

Researchers constitute the 'kernel' entities in scientific digital library portals such as CiteSeerX [1], DBLP [2] and Academic Search [3]. In addition, to the primary tasks related to search and browsing of research literature, digital libraries involve other tasks such as expertise modeling of researchers and social network analysis involving researcher entities. Most information required for these tasks needs to be extracted using the publications associated with a particular researcher along with the information provided by researchers on their homepages. To enable the collection of these homepages, we study the retrieval of researcher homepages from the Web using queries based on researcher names. We posit that researcher homepages are characterized by specific content-based and structural features which can be effectively harnessed for identifying them. We use topic modeling as a means to identify features that are discipline-independent while learning the ranking function for homepage retrieval. On a large dataset based on researcher names from DBLP, we show that our ranking function obtains an increase in success rate from 3.2% to 21.28% at rank 1 and from 29.6% to 66.3% at rank 10 over the baseline retrieval model that uses a similarity function based on query-content match. We also obtain modest performance with our small set of (about 125) discipline-independent features on identifying the researcher homepages in the WebKB dataset.

1. INTRODUCTION

Navigational queries where the search intent is to reach a particular website constitute a large proportion of search engine traffic [4, 23]. Invariably, these websites pertain to entities such as persons, organizations, products and so on. This task was well-studied as part of the Homepage Finding track at TREC [4]. In particular, locating people's homepages and profiles by using their names as search engine queries has become common on the Web. Building an accurate module that automatically identifies people's homepages is not easy. Not only can several people with the same name have homepages, the same person can have multiple webpages associated with him/her. For instance, when 'Michael Jordan' the name of a famous researcher in machine learning is used as a query on Google [5], his homepage appeared at the 15^{th} position in the search results. In this case, there are several pages related to the basketball player in the remaining results. For most researchers in Computer Science, it is common to see several pages such as those from wikipedia, publication pages on DBLP, pages from networking sites such as linkedin, pages from book websites in search results when their names are used as queries on popular search engines. In most cases, accurate retrieval of the correct homepage, requires the user to specify additional information. For instance, the previous "Michael Jordan" query can be augmented by keywords such as "Berkeley" or "Computer Science" to improve the rank at which it appears in the search results. However, this query-specific augmentation requires additional information which is not always available apriori.

We refer to "researcher homepages" as "academic homepages" or simply 'homepages' in this paper. In general, a researcher's homepage contains information about the employment status of a researcher and his or her professional interests. Other information related to a researcher's publications, work and contact details might also be found on a homepage. Researcher homepages are essential resources in digital library access portals like CiteSeerX [14] and ArnetMiner [22]. Indeed, homepages are heavily used in these portals for performing backend tasks like crawling for new literature, metadata extraction of authors etc. For instance, the crawler in CiteSeerX uses author homepages as seeds for obtaining up-to-date publications for its research literature collection. Homepages also constitute the target of navigational queries based on researcher names that are common on digital library portals. Indeed, a study based on a month's traffic to CiteSeerX [6] indicates that about 9% of the queries constitute researcher names. Accessing homepages is also a natural second step after expertise search which involves ranking people based on their expertise on the queried topic [7].

In this paper, we focus on designing a ranking function for researcher homepage retrieval. We start by showing that traditional retrieval engines that use content-based similarity between the queried name and the document content per-

[1] http://citeseerx.ist.psu.edu/
[2] http://www.informatik.uni-trier.de/~ley/db/
[3] http://academic.research.microsoft.com/
[4] http://trec.nist.gov/

[5] search performed on 02/01/2011

EOS, SIGIR 2011 Workshop Beijing, China

form poorly with respect to homepage ranking. We therefore investigate the content and structural aspects of homepages that distinguish them from other webpages. The observations from this investigation are used to design features for learning a ranking function. Our focus is on learning a set of features that are discipline-independent. On a publicly-available dataset of 112550 webpages and 6367 researcher name queries, our ranking function obtains remarkable improvements with respect to the success rate and MRR (mean reciprocal rank) measures and beats both the retrieval method using language modeling and inference networks [6] and a second method that uses query expansion [13]. Our features can also be used for discriminating homepages from other type of pages in absence of name (query) information.

Previous research related to our contributions is briefly summarized in the `Related Work` section. We describe our proposed features for learning to rank homepages after that. The `Experiments` section covers the details of evaluation and observations whereas a summary and notes on future directions conclude the paper.

2. RELATED WORK

Homepage finding was addressed a TREC task in 2001. While the focus of this task was not specifically on people's homepages, the track served as a platform for addressing homepage finding in a systematic manner and provided a dataset for evaluation. Several researchers employed machine learning techniques for addressing homepage finding at TREC. Most top-performing systems of this track use query-independent features such as URL-type and PageRank along with features based on match with the query in the content and anchor-text [25, 23, 20].

Academic homepage identification was addressed for mining researcher profiles and integrating with the publication data in ArnetMiner [22]. Tang, et al. address homepage finding as a binary classification task and use URL-based features on the pages retrieved using Google API with researcher name queries. Collecting academic homepages in absence of name information was previously addressed in the Japanese domain using word features (from a manully-assembled dictionary) on the page and the surrounding pages that link to it [24]. Macdonald, et al. address homepage finding with the goal of building candidate profiles based on which on which experts are ranked [16]. They identify candidate homepages using the anchor-text based features, since the anchor text pointing to a homepage usually contains the candidate's name. Anchor-text features of incoming links to a homepage are not commonly available in the web-search scenario we address in this paper. It is also desirable to classify pages based on on-page HTML features and content.

Learning to rank is a relatively new area of research where the goal is to learn a ranking function based on partial preferences available as part of the training data [15]. Several algorithms for learning to rank were proposed such as RankSVM [11], RankNet [5] and PermuRank [26]. While each of these algorithms handle different aspects of the learning to rank problem (for instance, unbalanced number of preferences per example is addressed in RankNet), an essential component for all these methods is the design of features based on the task at hand. Most learning to rank algorithms are evaluated on clickthrough logs of real search engines where the type of query is not particularly addressed. In contrast, we solely address learning to rank for a homepage retrieval engine where the input queries are assumed to be "researcher name" queries. For our problem, we concentrate on designing good features that can be potentially used with any of the learning to rank algorithms. Our features are based on content-analysis using Latent Dirichlet Allocation models [2, 9, 10] and adaptations of HTML and URL features previously used in webpage genre identification [21, 12].

[6] http://www.lemurproject.org/indri/

3. PROPERTIES OF RESEARCHER HOMEPAGES

We start with an analysis of researcher homepages in order to get insights for feature design. The following observations are based on anecdotal analysis of homepages in our dataset (described in the `Experiments` section). It is evident that most researchers maintain their homepages to advertise their work and academic standing on the Web. As opposed to other types of webpages we found that certain categories of information are likely to appear more often than not on a researcher homepage. For instance, it is common to find the name and affiliation information along with a summary of educational background and current research activity on a researcher homepage. Additionally, most researchers either provide a list of representative publications or a link to the same on their homepage. Sometimes information related to a researcher's employment position might be found on the homepage. For instance, a faculty member might provide a list of courses s/he teaches or the students s/he advises. Prompted by these observations, we visualize the content of a researcher homepage as a mixture of different categories of information. Next, we analyze the content of researcher homepages with tools that support such a conjecture.

Table 1: Top words from Topics of Homepages

talk	page	students	member
slides	home	graduate	program
invited	publications	faculty	committee
part	links	research	chair
talks	contact	cse	teaching
tutorial	personal	student	board
seminar	list	undergraduate	editor
summer	updated	college	courses
book	fax	current	state
introduction	email	ph	activities
chapter	department	school	technical
group	interests	program	associate
workshop	phone	university	special
lectures	info	grant	education
presentation	homepage	news	present

3.1 Homepages as Topic Mixtures

Latent topic models posit that a document can be viewed as a mixture of a small number of latent topics and that each 'observed' word in the document can be attributed to one of those topics. If each 'category' of information that we described in the previous section, is treated as a `topic`, it is easy to map the creation of homepage by a researcher to the document generation process commonly used in topic models such as LDA (Latent Dirichlet Allocation). Consider for instance, the following steps of homepage creation by a researcher: The researcher first samples a mixture of topics that s/he would like to present on his or her homepage.

Each topic corresponds to a specific category of information such as contact information, publications etc. Depending on the person's preference, the homepage might contain more information related to one category (topic) than the others. Next, each category of information is laid out on the homepage depending on personal preferences. LDA uses the following generation process for each term position of the document:

1. Sample a topic for that position
2. Sample a word conditioned on the chosen topic

We refer the reader to [2, 9, 10] for gaining a deeper understanding of LDA including the plate notation, sampling equations and the estimation process. Previous research using LDA has shown its effectiveness as an unsupervised tool for analyzing text corpora. When LDA is used on the documents of a corpus, we obtain two important quantities as the by-product of the modeling process: [1] a topic-term association matrix, ϕ whose entries correspond to the predictive distributions of words given topics, and [2] the topic proportion vectors for each document. The $\phi_{w,i}$ values correspond to the probability of a word w given the topic i. After an LDA run, every term position in the document is randomly assigned a topic based on these probabilities. These assignments are used to express each document in terms of its topic proportion vector, θ_d. Thus, each document can now be embedded into a lower-dimensional topic space. From an analysis standpoint, obtaining the top words for a given topic (words with high probability values for that topic from ϕ) usually helps in discerning the underlying theme captured by that topic. Table 1 show the top words of topics indicative of homepages obtained after running LDA on known homepages in our DBLP dataset (`Experiments`). It is encouraging to see that these topics capture the "categories" of information that we described earlier based on anecdotal observations. We obtain the top words of the topics pertaining to homepages to form the dictionary of words for feature generation. The output from LDA also indicates other topic-term clusters that indicate other types of information. For instance, a sample list of topics that reflect subject-areas is shown in Table 2. In order, to learn a ranking function that works for different disciplines, it is important to avoid features that are discipline-specific. A simple manual examination of the topic-term associations from LDA provide an easy method to isolate homepage-specific topics thus enabling us to avoid the discipline-specific topics and related words. Note that there is no such easy way to isolate discipline-independent features in other feature selection methods such as mutual information.

3.2 Structural Aspects of Homepages

Our second set of observations pertain to the HTML structure and URLs of homepages. Though it is not always the case, it is very common for homepages to be hosted on the university or the institute domain the researcher is affiliated with. In addition, homepage URLs that belong to a certain university usually follow a particular naming convention for the URL (For example, a tilde followed by author's lastname after the department URL). URL conventions are less consistent across institutions as opposed to the HTML structure-based features that we now describe. It appears that it is a common convention to put the author's name, sometimes coupled with the word 'homepage' in the title tag of the HTML. Similarly, it is fairly uncommon for academic homepages to contains several tables or images embedded in them. Table 3 captures the list of URL, HTML and query dependent features we used for training our ranking function. The first three features in this table as were used previously in ArnetMiner for obtaining the homepages of authors and used for extracting expert profiles [22]. We show later that these features are indeed rather effective for homepage retrieval but large improvements are still possible with our list of content-based and structural features. Some of the features in Table 3 are homepage-specific variations of features previously used for genre identification.

Table 2: Top words from topics on subject areas

data	multimedia	systems	design
database	content	distributed	circuits
databases	presentation	computing	systems
information	document	peer	digital
management	media	operating	signal
query	data	grid	vlsi
systems	documents	storage	ieee
xml	based	middleware	hardware
acm	hypermedia	system	fpga
vldb	video	scale	implementation
sigmod	user	high	power
icde	adaptation	large	architectures

Table 3: Structural Features

	Feature Description
1	name match in HTML title
2	name match in HTML content
3	name match in url
4	depth of url
5	tilde in url
6	domain of the url (com, edu, ac or other)
7	term home in HTML title
8	number of links
9	number of links in the same domain
10	ratio of 8 and 9
11	number of images
12	number of tables
13	Number of links to pdf files
14	Links with anchor text containing research/publications

3.3 Researcher Homepage Retrieval

Our goal is to provide a re-ranking module on top of the results of a standard search engine (such as Google), that ranks academic homepages of researchers better than other webpages. Thus, as opposed to the generic search scenario, the target-type of our search (viz. researcher homepage) is known in advance. The previous section summarizes some common aspects of homepages w.r.t. their content and structure. We now explore the use of these features for learning a ranking function that works on top of the search results from a web search engine. Although we focus on using page-specific features, features external to the webpage such as PageRank or anchor text content can be folded into the ranking module when they are available. Content-based ranking methods typically use features based on match with the query for scoring documents. We show that using these features alone does not result in effective homepage ranking.

The Indri search engine uses a combination of inference networks and language modeling for content-based retrieval. The retrieval function in Indri has been shown to be effective

in a wide range of retrieval tasks at TREC. In addition to being a competitive baseline that scores documents based on query-document match, Indri provides the pseudo-relevance feedback feature as well as the provision to specify expanded queries [13, 19]. For learning the ranking function, we use the RankSVM [7] model proposed by Joachims [11]. During training, the RankSVM learns a retrieval function that maximizes the Kendall's τ measure based on the partial order information present in training examples. Kendall's τ is related to the Average Precision measure commonly used in ranked retrieval. It is intuitive to view a homepage, non-homepage pair as a preference pair and minimize the number of inversions in the training set while learning the ranking function, which is precisely what the RankSVM does. Our focus is on design on features for the ranking function and other learning to rank approaches (mentioned in Section 2) can as well be used for this purpose. We provide experimental results with the ranking functions learnt using RankSVMs with the following feature sets:

1. **Topic Proportions of Documents (tprop)** Latent Dirichlet Allocation embeds each document in the corpus as a vector in the topic space. We use the proportions of topics that pertain to homepages as features (real values between 0 and 1).

2. **Word Features (lda)** After identifying the topics from LDA that relate to homepages, we use the top-20 words from these topics to form a word dictionary. Each document can be expressed as a vector in this word space using their normalized term frequencies. These values are used as features for training the ranking function.

3. **Query-dependent Features (am)** This set comprises the features 1-3 and 7 from Table 3. These features were previously used by Tang, et al. for training a binary classifier that identifies academic homepages in ArnetMiner [22]. We include this set for comparison with our proposed features.

4. **All Features (lda+struct)** The structural features summarized in Table 3 are used together with the word features in this set.

We illustrate the performance benefits of learning a ranking function for homepage retrieval by comparing with the following methods:

1. **Baseline (bl)** The ranking function in Indri using language modeling and inference networks is used as the baseline with researcher names as queries.

2. **Pseudo-Relevance Feedback (prf)** We use the pseudo-relevance feedback mechanism of Indri to rank pages in response to researcher-name queries. This is a two-step process which involves, using the top words from documents retrieved with the original query to perform query expansion. This form of expansion was found to effective in certain settings where the query is precise [1, 17].

3. **Query Expansion (qexp)** Based on searching experience when the correct result is not retrieved using the "researcher name" as query, one expects better success by using an expanded query containing other discriminative words. For instance, for the "Micheal Jordan" example, mentioned in the introduction, one could try adding "machine learning" or "publications" in the query to filter out the basketball-player related results. In this baseline, we use the top-words from homepage-related topics(Table 1) to form the expanded query and use the expansion mechanism provided in Indri for retrieval.

[7] http://svmlight.joachims.org/

4. EXPERIMENTS

We now describe our experimental setup and results using the various methods listed out in the previous section. The following datasets are used in evaluation:

1. **DBLP Dataset** DBLP provides bibliographic information related to journals and proceedings in Computer Science areas like databases, networks and machine learning. Over one million articles are indexed in DBLP along with names of the authors of these articles. The DBLP dataset was used to estimate the number of researcher homepages on the Web with mark-recapture techniques [6]. The author names for which the correct homepage URL was also available in DBLP were used to search the Web with the Yahoo BOSS API [8]. If the listed homepage was found in the top 20 results of this search, the webpages that could be crawled from these 20 hits were added to the dataset. The resulting dataset contains 112550 webpages out of which 6367 are researcher homepages.

2. **WebKB** The WebKB dataset [9] comprises of 8,282 webpages from universities categorized into student, faculty, staff, department, course, project and other pages. This dataset is heavily used for evaluating document classification methods. For our experiments, we re-assigned the true labels for the pages in this dataset by categorizing student and faculty pages as researcher homepages and treating all remaining webpages (pages of the 'other' class were ignored) as non-homepages. Although the researcher name information is not available for this dataset, we can test the effectiveness of our query-independent features in discriminating homepages from other academic webpages.

We ran LDA on the homepages present in the DBLP dataset using the implementation provided with Mallet [18]. The LDA estimation process requires setting the number of topics and other parameters for priors of hyperparameters. We used the default settings along with the hyperparameter optimization option while running LDA. For the number of topics, we experimented with settings 10-200 in steps of 10 and evaluated the training data likelihood. The top-2 settings for which the best likelihood values were obtained were 70 and 80. However, manually examining the top words for each topic indicated that for settings greater than 70, several words are repeated under different topics indicating that multiple topics might be covering the same theme. We therefore set the number of topics to 70. The output from running LDA on a corpus includes clusters of terms that are highly probable for each topic. These words usually indicate the underlying theme covered by the given topic. For instance, the top words shown in Table 1 are indicative of themes like "contact information" (second column) and

[8] http://developer.yahoo.com/search/boss/

[9] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/

"professional activity" (last column). However, when LDA is run on homepages, other topics are also identified such as the ones pertaining to subject areas (Table 2). These topics indicate that it is common to find information related to "research interests" or "area of work" on researcher homepages. However, notice that these topics are an artifact of the dataset and one of our goals was to design a ranking function that is discipline-independent. The DBLP dataset is based on homepages from Computer Science and the words in the table clearly show topics corresponding to databases, multimedia, distributed systems and circuit design, the subfields in Computer Science. We manually examined the output from LDA to identify the topics that specifically pertain to homepages. Typically, about 10% of the identified topics (about 5 to 7 out of 70) correspond to homepages. Automatic identification of suitable topics via a suitable validation set and a greedy-scheme was previously studied by Das, et al [6].

4.1 Experimental Setup

We obtain the top-20 words corresponding to the homepage-related topics to form the word dictionary. The size of this dictionary is about a 100 words. Using a larger number of words per topic did not improve performance but using fewer words per topic did not perform as well. We use the normalized term counts of the words in this dictionary for generating word features. Other representations such as boolean, raw term counts and TFIDF performed worse than the normalized term count representation. For topic proportion vector, we directly use the mixture proportions output by LDA as features. These are real values between 0 and 1.

For the ranking experiments using DBLP we provide the results averaged across five random splits of the data. Since the names of the researchers whose homepage is present in this dataset is available, for the baseline methods, we use Indri to construct an index over this dataset and use the underlying search engine to check if the correct page is retrieved when the name is used as a query. For training the ranking function, for each example in the training folds, the correct homepage along with a randomly chosen example from the other pages obtained with Yahoo (check `Datasets` description) for this name, forms an ordered pair that is used to create a training instance for RankSVM. The ranking function learnt on the training folds is use to re-rank the top-100 results retrieved with Indri for queries in the test fold. The success rate (SR) and mean reciprocal rank (MRR) measures are used to evaluate performance on the test folds. For an example, if the correct homepage is found among the top-k results, this contributes a value of 1 to the aggregate score for SR@k and a value of $\frac{1}{k}$ for MRR@k. The aggregate score is normalized by the number of test queries. For the classification setting, we use the F1 measure to evaluate performance. More information regarding the use of these measures in ranked retrieval and classification can be found in Manning, et al. [17].

The performance for various settings is shown in the Table 4 and the associated plots. Since user studies indicate that people rarely browse beyond the second page of search results [8], we use success rate and MRR at $k = 1, 3, 5, 10, 20$ for evaluating the performance of our ranking function. We also present the F1 measure for the WebKB dataset.

4.2 Results and Observations

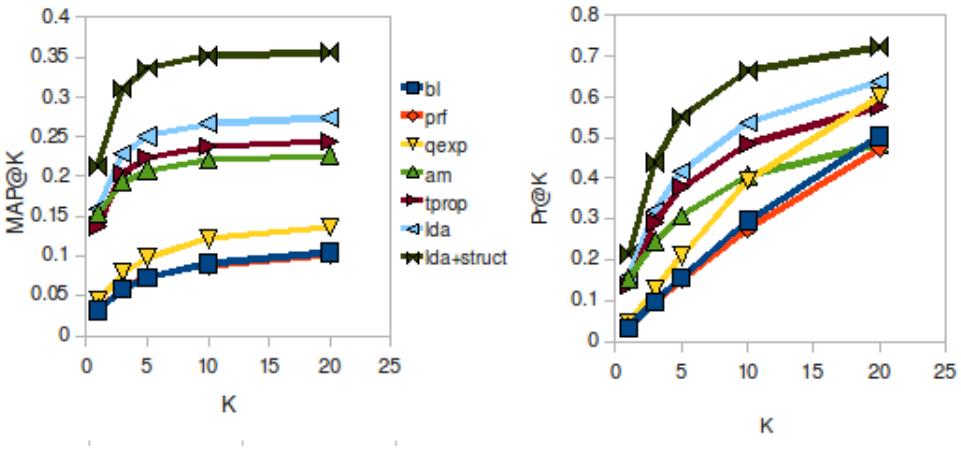
Table 4 and the associated plots capture the success rate and MRR variation with top-k for the different baseline and proposed methods. From these plots it is clear that when compared to the baseline methods that are typically how custom search applications are built, large improvements can be obtained by using ranking functions trained with the knowledge of the expected target page. Pseudo-relevance feedback was not found to be useful but query expansion using the LDA word features is useful. However, specifically training a ranking function using RankSVMs with these features is more effective than the straightforward query expansion. (compare the 'lda' curves with the 'qexp' curves in both the plots). The query-dependent URL features used for classifying webpages in ArnetMiner are indeed good indicators of homepages. However, both the topic proportion and word features individually yield larger improvements than using the ArnetMiner features alone. Word features are better than the topic proportion features but combining both these sets did not improve results (not shown). This is not surprising since the topic proportion vector can be viewed as an embedding of the webpage from the higher-dimensional word-space into a lower-dimensional topic space. Word features that are fine-grained that aggregated topic proportion value appear to be more beneficial w.r.t. ranking. The best performance is obtained when the structural features are used together with the word features. Indeed, compared to the Indri baseline, using a ranking function trained with the lda+struct features gives a remarkable improvement in the success rate from 3.2% to 21.28% at rank position 1 and from 29.55% to 66.31% at rank position 10.

Other combinations such as using expanded queries with structural features did not beat the performance with the lda+struct features. The margin trade-off parameter 'C' in RankSVM was set to 0.01 in the experiments in Table 4. We experimented with other values of C between 0.005 to 0.025 but found no considerable difference in the performance.

For the WebKB dataset, we evaluate classification performance using a classifier trained on the DBLP dataset with word and structural features. Only query-independent features were used since queries (researcher names) are not available for this set. The objective of this experiment is to evaluate the robustness of our features in identifying homepages in absence of names. This set up is common while building a focused crawler for accumulating a homepage collection in absence of names. We obtained an F1 score of 0.5738 on the WebKB dataset. Classification algorithms are sensitive to the imbalance of the dataset and the number of features used in training both in terms of performance and training time. We found decision trees to be the best performing among the other classifiers we tried for this task(viz., logistic regression, two-class and one-class SVMs). Given that the set of features used is small (about 125) and discipline-independent, this performance seems comparable to the results published on the WebKB dataset. Usually thousands of unigram and bigram [3] features are used and the reported accuracies range between $70 - 90\%$ on all the classes as opposed to homepage/non-homepage classes used by us.

5. CONCLUSIONS AND FUTURE DIRECTIONS

We focused the problem of homepage retrieval for researcher entities in digital libraries. We described the design



Method	SR@1	MRR@1	SR@3	MRR@3	SR@5	MRR@5	SR@10	MRR@10	SR@20	MRR@20
bl	0.0316	0.0316	0.0965	0.0588	0.1573	0.0725	0.2955	0.0902	0.5033	0.1046
prf	0.0344	0.0344	0.0970	0.0604	0.1517	0.0727	0.2745	0.0884	0.4740	0.1021
qexp	0.0448	0.0449	0.1262	0.0786	0.2094	0.0974	0.3946	0.1214	0.6023	0.1360
am	0.1540	0.1540	0.2456	0.1937	0.3066	0.2075	0.4058	0.2207	0.4839	0.2263
tprop	0.1377	0.1377	0.2925	0.2042	0.3777	0.2235	0.4840	0.2377	0.5757	0.2441
lda	0.1591	0.1591	0.3203	0.2285	0.4175	0.2505	0.5356	0.2664	0.6377	0.2736
lda+struct	**0.2128**	**0.2128**	**0.4388**	**0.3106**	**0.5512**	**0.3364**	**0.6631**	**0.3516**	**0.7225**	**0.3559**

Table 4: MRR and Success Rate(SR) for different methods

of a ranking function for this purpose using various content-based and structural aspects of researcher homepages. The effectiveness of our discipline-independent features was experimentally illustrated on two datasets.

Our interest in researcher homepages is due their resourcefulness for various tasks pertaining to digital libraries. Our future work is related to the aquisition of researcher homepages for building a homepage collection with one-one mappings between researchers (as represented in the digital library) and their homepages. As opposed to humans looking for researcher homepages where it might be sufficient to have the homepage among the top-10 results, generating such mappings automatically involves choosing a small number (one or two) of webpages for each researcher. Accuracy at this stage is very crucial since the performance of other tasks such as crawling for latest publications from homepages and researcher metadata extraction depend on the identified homepages.

6. ACKNOWLEDGMENTS

The first author would like to thank Sumit Bhatia and Shibamouli Lahiri for several insightful discussions on the topics addressed in this paper.

7. REFERENCES

[1] J. Allan. Relevance feedback with too much data. In *SIGIR*, 1995.
[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 2003.
[3] C. Boulis and M. Ostendorf. Text classification by augmenting the bag-of-words representation with redundancy-compensated bigrams. In *FSDM*, 2005.
[4] A. Broder. A taxonomy of web search. *SIGIR Forum*, 2002.
[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML*, 2005.
[6] S. Das, C. L. Giles, P. Mitra, and C. Caragea. Content-based methods for identifying academic homepages. In *JCDL*, 2011.
[7] H. Deng, I. King, and M. R. Lyu. Formal models for expert finding on dblp bibliography data. In *ICDM*, 2008.
[8] L. A. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www search. In *SIGIR*, 2004.
[9] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 2004.
[10] G. Heinrich. Parameter estimation for text analysis. Technical report, 2008.
[11] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002.
[12] I. Kanaris and E. Stamatatos. Learning to recognize webpage genres. *Inf. Process. Manage.*, 2009.
[13] V. Lavrenko and W. B. Croft. Relevance-based language models. In *SIGIR*, 2001.
[14] H. Li, I. G. Councill, L. Bolelli, D. Zhou, Y. Song, W.-C. Lee, A. Sivasubramaniam, and C. L. Giles. Citeseerx: a scalable autonomous scientific digital library. In *InfoScale '06*, 2006.
[15] T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, March 2009.
[16] C. Macdonald, D. Hannah, and I. Ounis. High quality expertise evidence for expert search. In *ECIR*, 2008.
[17] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. 2008.
[18] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.
[19] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Inf. Process. Manage.*, 2004.
[20] R. Nallapati. Discriminative models for information retrieval. In *SIGIR '04*, 2004.
[21] M. Santini. Automatic identification of genre in web pages. *PhD Thesis*, 2007.
[22] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *KDD '08*, 2008.
[23] T. Upstill, N. Craswell, and D. Hawking. Query-independent evidence in home page finding. *ACM Trans. Inf. Syst.*, 2003.
[24] Y. Wang and K. Oyama. Web page classification exploiting contents of surrounding pages for building a high-quality homepage collection. In *Digital Libraries: Achievements, Challenges and Opportunities*. 2006.
[25] W. Xi, E. Fox, R. Tan, and J. Shu. Machine learning approach for homepage finding task. In *String Processing and Information Retrieval*. 2002.
[26] J. Xu, T.-Y. Liu, M. Lu, H. Li, and W.-Y. Ma. Directly optimizing evaluation measures in learning to rank. In *SIGIR*, 2008.

An Evaluation Framework for Aggregated Temporal Information Extraction

Enrique Amigó
UNED NLP & IR group
Madrid, Spain
enrique@lsi.uned.es

Javier Artiles, Qi Li, Heng Ji
Computer Science Department
Queens College and Graduate Center (CUNY)
New York, USA
javier.artiles@qc.cuny.edu

ABSTRACT

This paper focusses on the representation and evaluation of temporal information about a certain event or entity. In particular, we consider temporal information that can be normalized to specific dates. This task requires the aggregation of temporal relations between events and dates extracted from multiple texts. Given that the resulting temporal information can be vague, it is necessary that an evaluation framework captures and compares the temporal uncertainty of system outputs and human assessed gold-standard data. Current representation models and measures are not suitable for this scenario. In this paper, we propose a novel representation model and assess its properties and limitations. In order to compare extracted information against a gold standard, we define an evaluation metric based on a set of formal constraints. Finally, we present experiments that show the behavior of the proposed metric. The task setting and the evaluation measure presented here have been introduced in the TAC 2011 Knowledge Base Population evaluation for the Temporal Slot Filling task.

1. INTRODUCTION

Recent research on Information Extraction (IE) has seen an evolution from single document to cross-document extraction tasks in an effort to build structured knowledge bases from large collections of unstructured texts. Among the most successful efforts in this direction is the Knowledge Base Population (KBP) track, which has gathered researchers around the problem of extracting of information about entities linked to an external knowledge source [13, 9]. Another example is the Web People Search (WePS) task, where systems are requested to cluster multiple search results about the same person and extract biographical attributes [3, 4, 2].

Several recent studies have stressed the benefits of using information redundancy on estimating the correctness of IE output [7], improving disease event extraction [22] and MUC event extraction [12, 14]. However, the information obtained on IE tasks such as KBP or WePS is viewed as static, ignoring the temporal dimension that is relevant to many types of attributes. While this is a reasonable simplification in many situations, it is unsatisfactory for applications that require some awareness of the time span when a fact was valid. Furthermore, considering a temporal dimension on a cross-document task adds the challenging problem of managing vague or incomplete information about the temporal boundaries of a particular fact.

This work focuses on the development of an evaluation framework for extracting temporal information about a particular event in order to estimate its specific temporal boundaries. Temporal information can be scattered across documents (e.g. in one document "John joined Microsoft in Sept. 1990" and in another document "Microsoft resigned his contract yesterday "), and expressed with different granularities (e.g. "He started working for Microsoft in the 90s", "He began his contract in September of this year"). Several evaluation frameworks for temporal information extraction have been defined; although, most of the previous work has focused on temporal relations extracted from single texts. Other work addresses information aggregation, but they do not provide a formal representation and evaluation framework.

In this paper, we present a model for representing temporal information that has been derived from the task requirements. After this, we present a novel evaluation measure that allows the comparison of system outputs against human assessed solutions. Given that the representation model is new, it is not possible to compare our approach to a previously stablished standard evaluation measure. However, we do define a set of formal constraints which state the properties that any measure should satisfy on this task. In addition, we describe some experiments that empirically assess the measure behavior and properties. This work has motivated the introduction of the Temporal Slot Filling task in the TAC 2011 Knowledge Base Population (KBP2011) evaluation campaign. In this task, systems are required to add temporal boundaries to IE slots such as "employee of", "country of residence" or "spouse".

In Section 2 we formally describe our proposed evaluation framework. In Section 3 we compare our proposal with previous related work. Section 4 describes our experiments and in Section 5 we discuss our conclusions.

2. THE EVALUATION FRAMEWORK

2.1 Time Representation

According to our task definition, we assume that: (i) events are not discontinuous in time; (ii) the temporal information is distributed across several documents; and (iii) both the gold standard and system outputs can contain uncertainty. This uncertainty can be due to the variable levels of granularity of temporal information (e.g. years, months) or to the

EOS, SIGIR 2011 workshop, July 28, Beijing, China

reasoning based on temporal order relations ("He worked for Microsoft before working for Apple").

Given the previous assumptions the representation model should consider temporal ranges for both the beginning and ending points. For simplicity, we assume that uncertainty follows a uniform distributions over time ranges. Our representation model consists of a tuple $< t_1, t_2, t_3, t_4 >$, which represents the set S of possible beginnings and endings of an event such that:

$$S = \{< t_{init}, t_{end} > | (t_1 < t_{init} < t_2) \wedge (t_3 < t_{end} < t_4)\}$$

In other words, t_1 and t_3 represent the lower bounds for the beginning and ending points respectively, while t_2 and t_4 represent the upper bounds.

Interesting properties of this temporal representation model include:

Aggregation: Each tuple represents a set of possible pairs $< t_{init}, t_{end} >$, allowing a straightforward aggregation of temporal information:

$$S \cap S' \equiv < max(t_1, t'_1), min(t_2, t'_2), max(t_3, t'_3), min(t_4, t'_4) >$$

Temporal relations event-time: This model is able to represent Allen's relations [1] between an event and a time interval. For instance, if an event S happens before a certain time interval $< t_a, t_b >$, then the latest time boundaries in S must be previous to the beginning of the time interval (t_a)[1] (see [1] for a graphical depiction of these relations):

$$S_{(BEFORE < t_a, t_b >)} \equiv < -\infty, t_a, -\infty, t_a >$$

Temporal relations event-event: Although our model does not capture directly time relations between events, it allows the reduction of time uncertainty when these relationships are detected. Given two events represented by the tuples S and S' and an Allen relation, there exists a simple operation over tuples that reduces the temporal uncertainty for both events. For instance, if S happens before S', the upper bounds for S must be previous to the S' beginning upper bound:

$$S \ BEFORE \ S' \rightarrow$$
$$(S =< t_1, min(t_2, t'_2), t_3, min(t_4, t'_2) >)$$
$$\wedge (S' =< max(t'_1, t_3), t'_2, max(t'_3, t_3), t'_4 >)$$

Temporal inconsistencies: The representation model provides a straightforward method to detect inconsistencies when aggregating temporal information in a tuple. An event tuple is inconsistent if a lower bound exceeds its corresponding upper bound:

$< t_1, t_2, t_3, t_4 >$ is inconsistent if and only if

$$t_1 > t_2 \vee t_3 > t_4 \vee t_1 > t_4$$

For instance, according to the BEFORE operator previously defined, given to (consistent) events S and S', a temporal relation $S \ BEFORE \ S'$ is inconsistent if the boundary lower bounds in S exceeds the S' beginning upper bound:

$S \ BEFORE \ S'$ is inconsistent if $t_1 > t'_2 \vee t_3 > t'_2$

[1]Due to the space limitations we do not list all relations.

The main limitation of assuming that events are continuous is that our representation model is not able to capture some relations such as regularly recurring events ("each Friday") or some fuzzy relations ("lately", "recently") that are encoded with the SET type in TimeML [15].

In short, besides providing the starting point for our evaluation framework, this representation model allows us to: (i) aggregate temporal information about an event; (ii) represent temporal relations between events and dates (time interval); (iii) reduce the temporal uncertainty about events by considering temporal relations between them; and (iv) detect errors during the aggregation process by checking for inconsistencies between tuples.

2.2 Formal Constraints for an Evaluation Measure

In order to complete the evaluation framework, we need to define a metric $Q(S)$ that compares a system's output S against a gold standard tuple $S_g =< g_1, g_2, g_3, g_4 >$. We now define a set of formal constraints that any quality metric based on our representation model should satisfy.

Best System Constraint: The maximum score should be achieved only by a tuple that exactly matches the gold standard:

$$Q(S) = 1 \leftrightarrow t_i = g_i \ \forall i \in \{1..4\}$$

Worst System Constraint: When the gold standard tuple does not contain any ∞ or $-\infty$ component, an uninformative tuple, such that all elements are ∞ or $-\infty$, always achieves the minimum score. This constraint prevents over-scoring tuples that do not produce any useful information.

$$t_i \in \{\infty, -\infty\} \ \forall i \in \{1..4\} \rightarrow Q(S) = 0$$

Quality Decrease: Any increment of the distance between a tuple component and the corresponding gold standard component implies a quality decrease. This constraint ensures that the metric is sensitive to any change in the tuple.

$$\triangle |t_i - g_i| \rightarrow \triangledown Q(S)$$

Temporal Boundary Independence: The correct estimation of any of the gold standard components g_i has an intrinsic value with no dependency on the rest of components. In other words, estimating correctly any of the components implies a quality higher than 0. The practical implication is that we cannot infinitely penalize an error in one tuple component.

$$(\exists i \in \{1..4\}.t_i = g_i) \rightarrow Q(S) > 0$$

Parameterization Constraint: According to the *Aggregation* property of our representation model, the more we aggregate temporal information, the less the resulting tuple is vague. However, we risk including incorrect information that is over-constraining the upper and lower bounds in the tuple. Depending on the task, the relative penalization for both aspects should be parameterizable in the metric. A parameter α should determine if a certain amount of *vagueness* is worse or better than a certain amount of *over-constraining*.

Formally, given two tuples S and S' with finite values such that $S \subset S_g \subset S'$, then:

$$\exists v/(\alpha > v) \leftrightarrow (Q_\alpha(S) > Q_\alpha(S'))$$

2.3 The Proposed Evaluation Metric

A simple approach to the evaluation metric would be to estimate the precision and recall between S and S_g sets. However, this approach would not satisfy the *Quality Decrease* constraint when S_g and S are not overlapped; for instance, when it is applied over events that happen at a single point in time ($g_1 = g_2 = g_3 = g_4$).

Our proposal consists of measuring absolute distances between components t_i and g_i.

$$Q(S) = \frac{1}{4} \sum_i \frac{c}{c + |t_i - g_i|}$$

Q is bounded between zero and one. An error of c time units produces a 0.5 score. As the value of c decreases errors are penalized to a greater extent. The c value can be fixed for all the events found in a corpus test set, or it can be set dynamically by considering a certain variable (e.g. event length in the gold standard).

According to the *Best System* constraint, the maximum score is achieved when $|t_i - g_i| = 0$ forall i. As required by the *Temporal Boundary Independence* constraint, estimating correctly any of the tuple components results in a quality higher than zero:

$$t_j = g_j \rightarrow Q(S) = \frac{1}{4}(\frac{c}{c + |t_j - g_j|} + \sum_{i \neq j} \frac{c}{c + |t_i - g_i|}) =$$

$$\frac{1}{4}(1 + \sum_{i \neq j} \frac{c}{c + |t_i - g_i|}) > 0$$

In addition, the metric is sensitive to any difference between tuples, satisfying the *Quality Decrease* constraint. For this, it is enough to demonstrate that $Q(S)$ is strictly decreasing regarding $|t_i - g_i|$. If we derive $Q(S)$ regarding any $|t_i - g_i|$, we obtain:

$$\frac{d\ Q(S)}{d\ |t_i - g_i|} = -\frac{c}{(c + |t_i - g_i|)^2} < 0$$

Finally, when all components in the reference tuple are finite, a non finite value in all the evaluated tuple component implies a minimum quality (*Worst System* constraint).

$$Q(< -\infty, \infty, -\infty, \infty >) = \frac{1}{4} \sum_i \frac{c}{c + \infty} = 0$$

In order to satisfy the *Parameterization* constraint, we just have to refine the c parameter by cases. *Vagueness* ($S \supset S_g$) represents to what extent systems tend to produce longer uncertainty ranges. *Over-constraining* ($S \subset S_g$) represents to what extent the constraints are false or the specificity of the evaluated tuple is higher than that of the gold standard tuple. Both quality criteria are affected by the following time mismatches:

$$S \supset S_g \longrightarrow t_1 \leq g_1 \wedge t_2 \geq g_2 \wedge t_3 \leq g_3 \wedge t_4 \geq g_4$$
$$S \subset S_g \longrightarrow t_1 \geq g_1 \wedge t_2 \leq g_2 \wedge t_3 \geq g_3 \wedge t_4 \leq g_4$$

S_g	**1950**	**1960**	**1970**	**1980**	
S	1950	1960	1970	1980	$Q = 1$
S	1945	1955	1975	1985	$Q = 0.5$
S	1940	1950	1980	1990	$Q = 0.33$
S	0	10000	1970	10000	$Q = 0.25$
S_g	**0**	**1960**	**1970**	**10000**	
S	0	1960	1970	10000	$Q = 1$
S	1950	1960	1970	1980	$Q = 0.5$

Table 1: Evaluation examples ($c_{vag} = c_{cons} = 5$).

These two quality aspects are complete. The two possible ordering relations between g_i and $t : i$ are represented. We can satisfy the *Parameterization* constraint by setting a different c value for vagueness and over-constraining.

$$Q(S) = \frac{1}{4} \sum_i \frac{c_i}{c_i + |t_i - g_i|}$$

Notice that an individual component of a tuple can only have vagueness or over-constraining error, but not both. Since the tuple is comprised of four components the overall tuple can have both of those types of errors.

$$c_i = \begin{cases} c_{vag}, & \text{if } (i \in \{1,3\} \wedge t_i \leq g_i) \vee (i \in \{2,4\} \wedge t_i \geq g_i) \\ c_{cons}, & \text{otherwise} \end{cases}$$

The demonstration for the *Parameterization* constraint is straightforward. If $S \subset S_g$ then:

$$Q(S) = \frac{1}{4} \sum_i \frac{c_{cons}}{c_{cons} + |t_i - g_i|}$$

If $S_g \subset S'$ then:

$$Q(S') = \frac{1}{4} \sum_i \frac{c_{vag}}{c_{vag} + |t_i - g_i|}$$

Therefore, depending on c_{cons} and c_{vag} both tuples can achieve any score between zero and one. Thus, there exists a parameter that determines if $Q(S) > Q(S')$.

Table 1 shows some examples of evaluated tuples[2]. In the first case the tuple is an exact match for all components. In the second case, there is an error of 5 years for each component, producing a final score of 0.5. In the next case, the error is greater (ten years) and the score decreases to 0.33. In the fourth case the system is not able to estimate three out of four component, obtaining a score of 0.25. In the last two examples, we have considered the situation in which there is no estimation in the gold standard for the earliest possible beginning and for the latest possible ending points. We approach the infinite values as extremely small and big values (0 and 10000). As the table exemplifies, if the tuple introduces a false lower bound for the beginning and a false upper bound for the ending, and the other two components are well estimated, the score is 0.5.

[2]For simplicity we consider years in our examples, although more fine-grained dates can be used.

3. RELATED WORK

The TempEval campaigns [21, 17] have evaluated the classification of temporal relations between events, time expressions and document creation dates, without considering cross-document relations. Most current research efforts are focused on these tasks [6, 20, 8, 10], which are evaluated using precision and recall measures over the TimeML [16] annotation scheme.

Other works have considered the aggregation of temporal information from multiple documents. In [23] the logical constraints of different temporal relations are considered jointly; although, evaluation is carried separately over each type of temporal relation. Thus, they do not provide an integrated representation and evaluation measure for the temporal information associated to a certain event. [18] represents and aggregates temporal information using a fuzzy version of Allen's Interval Algebra [1]. In this case the evaluation is carried indirectly, in terms of precision and recall of the classification of overlapped or non-overlapped events. Although they represent the uncertainty of temporal relations, they do not offer an evaluation measure that consider the vagueness of system outputs. In [19, 5] the time span of events is estimated based on the distribution of relevant terms across documents, but only a manual evaluation is provided.

Among the previous work [11] is the closest to our proposal. It estimates the ending and beginning time of an event across documents and their representation model consists of a pair indicating the starting and ending points. However, this representation model does not capture temporal uncertainty. They propose two evaluation measures. The first is a discrete measure of the precision and recall of correctly predicted constraints across sentences. Therefore, the *Quality Decrease* constraint is not satisfied. The second measure represents the tightness of the temporal bounds induced by a system. Although this measure is able to penalize long time intervals in responses, it does not give information about the amount of overlap between the estimated and actual bounds and does not satisfy the *Parameterization* property.

To the best of our knowledge, the previous evaluation frameworks are unable to capture and evaluate the uncertainty of temporal information derived from the aggregation of information from multiple documents.

4. EXPERIMENTS

In order to assess the behavior of the metric, we automatically generate a set of gold standard and "system" tuples. We generate 100 gold standard tuples according to the following procedure: r_{init} and r_{end} are two random values from (0..100) and (100..200):

$$g_1 = r_{init} - rand(0..50) \quad g_2 = r_{init} + rand(0..50)$$
$$g_3 = r_{end} - rand(0..50) \quad g_4 = r_{end} + rand(0..50)$$

4.1 Parameterization Experiment

In order to check the ability of the Q measure to weight the relative relevance of *vagueness* against *over-constraining*, we automatically produce two tuples for each gold standard; one vague tuple and one over-constrained tuple.

The components for the over-constrained and the vague tuples are:

$$S_{Overcons} =< t_1 + a_1, t_2 - a_2, t_3 + a_3, t_4 - a_4 >$$

$$S_{Vague} =< t_1 - a_1, t_2 + a_2, t_3 - a_3, t_4 + a_4 >$$

where $a_1 \in (0..(r_{init} - g_1))$,$a_2 \in (0..(g_2 - r_{init}))$, $a_3 \in (0..(r_{end} - g_3))$, and $a_4 \in (0..(g_4 - r_{end}))$. The limits for the a_i values avoid inconsistent tuples (e.g. $t_1 > t_2$). We have computed the average Q score for both approaches across the 100 gold standard cases. Figure 1 shows that depending on the c_{con} and c_{vag} parameters in the Q measure one system can improve the other and *vice versa*.

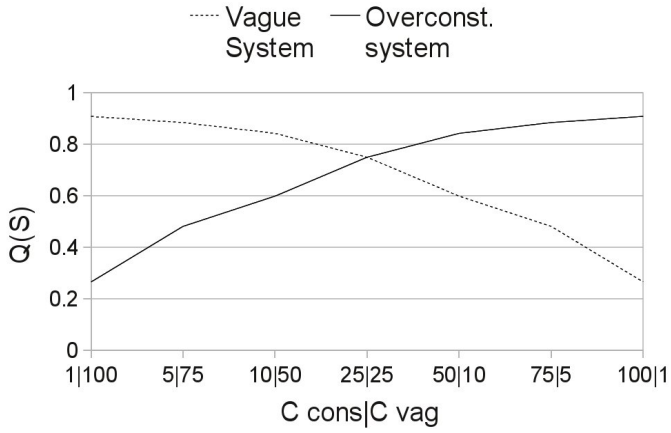


Figure 1: Parameterizing the c_{cons} and c_{vag} values in the Q measure over a vague and an over-constrained approach

4.2 Increasing the Error

The next experiment consists of producing tuples that start as a perfect match with the gold standard, and progressively incorporate a certain amount of error β. First, we generate a HOLD tuple assuming that the system correctly locates points in the time interval, but with an error β in the upper and lower bounds.

$$S_{HOLD} =< g_1 - \beta, g_2, g_3, g_4 + \beta >$$

Next, the BEFORE tuple detects a point before the event and correctly estimates the time lower bound. Analogously, we define the AFTER tuple:

$$S_{BEFORE} =< g_1, g2 + \beta, g_3 + \beta, g_4 + \beta >$$

$$S_{AFTER} =< g_1 - \beta, g2 - \beta, g_3 - \beta, g_4 >$$

The vague tuple progressively increases the uncertainty of the ending and beginning time points:

$$S_{VAGUE} =< g_1 - \beta, g2 + \beta, g_3 - \beta, g_4 + \beta >$$

Notice that it is not possible to emulate an infinitely over-constrained tuple, given that it would not satisfy the basic format constraints (e.g. $t_1 < t_2$). Also, we include a RAND tuple that estimates a single point in the time range with an error β around the center of the event.

$$S_{RAND} \equiv (t_1 = t_2 = t_3 = t_4 = \frac{r_{init} + r_{end}}{2} \pm \beta)$$

Figure 2 shows the Q score for these approaches when β is increased. In this case we have set the constants c_{con} and c_{vag} at 5. As the figure shows, the HOLD tuples approach a 0.5 score in the limit (note that, by definition, this approach captures a half of the tuple component). The AFTER and BEFORE tuples approach 0.3 in the limit, since they capture less information. When the VAGUE tuples are extremely vague their score approaches zero. Finally, the RAND tuples over-constrain the gold standard, achieving a low score even for small values of β.

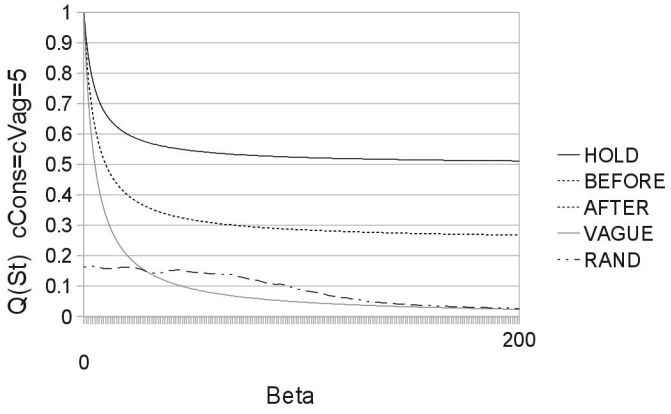


Figure 2: Several synthetic approaches with a increasing error beta

5. CONCLUSIONS

In this work we have proposed a novel framework to represent, aggregate and evaluate temporal information distributed in multiple documents. This framework allows the inclusion of operators that reduce the uncertainty when finding new time relations (events-date or event-event). The second contribution of this paper is the definition of an evaluation metric for this task grounded on a set of formal constraints. The empirical results over synthetic data show that the proposed metric: (i) can effectively assign a relative weight to vagueness vs. over-constraining; and (ii) correctly discriminates the quality of a set of controlled outputs.

One of the straights of the proposed model is its simplicity. Most temporal relations can be captured with a four element tuple or with a simple operator over tuples. Nevertheless two main limitations of our research are that our model can not represent discontinuous events such as regularly recurring events (e.g. "every weekend") and also that the assumption of a uniform distribution for uncertainty ranges does not allow us to capture fuzzy relations (e.g. "it happened at the beginning of the year 2000"). Our future work is to apply the representation and evaluation model over a collection of real data. The ongoing TAC KBP2011 Temporal Slot Filling task will provide a collection of manual assessments as well as testing the proposed metric on a variety of systems.

6. ACKNOWLEDGMENTS

This research was partially supported by the Spanish Ministry of Science and Innovation (Holopedia Project, TIN2010-21128-C02) and the Regional Government of Madrid and the European Social Fund under MA2VICMR (S2009/TIC-1542).

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053, the U.S. NSF CAREER Award under Grant IIS-0953149 and PSC-CUNY Research Program. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

7. REFERENCES

[1] J. F. Allen. Maintaining Knowledge about Temporal Intervals. In *Communications of the ACM, November 1983, Volume 26, Number 11*, volume 26, pages 832–843, New York, NY, USA, 1983.

[2] J. Artiles, A. Borthwick, J. Gonzalo, S. Sekine, and E. Amigó. WePS-3 Evaluation Campaign: Overview of the Web People Search Clustering and Attribute Extraction Tasks. In *2nd Web People Search Evaluation Workshop (WePS 2010), CLEF 2010 Conference, Padova Italy*, 2010.

[3] J. Artiles, J. Gonzalo, and S. Sekine. The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task. In *Proc. of the 4th International Workshop on Semantic Evaluations (Semeval-2007)*, 2007.

[4] J. Artiles, J. Gonzalo, and S. Sekine. WePS 2 Evaluation Campaign: overview of the Web People Search Clustering Task. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[5] H. L. Chieu and Y. K. Lee. Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 425–432, New York, NY, USA, 2004. ACM.

[6] L. Derczynski and R. Gaizauskas. Usfd2: Annotating temporal expresions and tlinks for tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 337–340, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[7] D. Downey, O. Etzioni, and S. Soderland. A Probabilistic Model of Redundancy in Information Extraction. In *Proc. IJCAI 2005*, 2005.

[8] E. Y. Ha, A. Baikadi, C. Licata, and J. C. Lester. Ncsu: Modeling temporal relations with markov logic and lexical ontology. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 341–344, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[9] R. G. H. T. D. Heng Ji and K. Griffitt. An Overview of the TAC2010 Knowledge Base Population Track. In *Proc. Text Analytics Conference (TAC2010)*, 2010.

[10] A. K. Kolya, A. Ekbal, and S. Bandyopadhyay. Ju_cse_temp: A first step towards evaluating events, time expressions and temporal relations. In

Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10, pages 345–350, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[11] X. Ling and D. S. Weld. Temporal information extraction. In *Proceedings of the Twenty Fifth National Conference on Artificial Intelligence*, 2010.

[12] G. Mann. Multi-document Relationship Fusion via Constraints on Probabilistic Databases. In *Proc. of HLT-NAACL 2007*, pages 332–339, 2007.

[13] P. McNamee and H. Dang. Overview of the TAC2009 knowledge base population track. In *Text Analysis Conference (TAC)*, 2009.

[14] S. Patwardhan and E. Riloff. A Unified Model of Phrasal and Sentential Evidence for Information Extraction. In *Proc. EMNLP 2009*, pages 151–160, 2009.

[15] J. Pustejovsky, J. M. Castaño, R. Ingria, R. Sauri, R. J. Gaizauskas, A. Setzer, G. Katz, and D. R. Radev. TimeML: Robust Specification of Event and Temporal Expressions in Text. In *Proc. of IWCS-5, Fifth International Workshop on Computational Semantics (IWCS-5)*, 2003.

[16] J. Pustejovsky, J. CastaÃśo, R. Ingria, R. Saura, R. Gaizauskas, A. Setzer, and G. Katz. Timeml: Robust specification of event and temporal expressions in text. In *in Fifth International Workshop on Computational Semantics (IWCS-5*, 2003.

[17] J. Pustejovsky and M. Verhagen. Semeval-2010 task 13: evaluating events, time expressions, and temporal relations (tempeval-2). In *Proc. of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 112–116, 2009.

[18] S. Schockaert, M. De Cock, and E. E. Kerre. Reasoning about fuzzy temporal information from the web: towards retrieval of historical events. *Soft Comput.*, 14:869–886, June 2010.

[19] R. Swan and J. Allan. Automatic generation of overview timelines. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, New York, NY, USA, 2000. ACM Press.

[20] N. UzZaman and J. F. Allen. Trips and trios system for tempeval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 276–283, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[21] M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 75–80, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[22] R. Yangarber. Verification of Facts across Document Boundaries. In *Proc. International Workshop on Intelligent Information Access*, 2006.

[23] K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 405–413, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Entity Search Evaluation over Structured Web Data

Roi Blanco
Yahoo! Research
Diagonal 177
Barcelona, Spain
roi@yahoo-inc.com

Harry Halpin
University of Edinburgh
10 Crichton St.
Edinburgh, UK
H.Halpin@ed.ac.uk

Daniel M. Herzig
Institute AIFB
Karlsruhe Institute of Technology
76128 Karlsruhe, Germany
herzig@kit.edu

Peter Mika
Yahoo! Research
Diagonal 177
Barcelona, Spain
pmika@yahoo-inc.com

Jeffrey Pound
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Canada
jpound@cs.uwaterloo.ca

Henry S. Thompson
University of Edinburgh
10 Crichton St.
Edinburgh, UK
ht@inf.ed.ac.uk

ABSTRACT

The search for entities is the most common search type on the web beside navigational searches. Whereas most common search techniques are based on the textual descriptions of web pages, semantic search approaches exploit the increasing amount of structured data on the Web in the form of annotations to web-pages and Linked Data. In many technologies, this structured data can consist of factual assertions about entities in which URIs are used to identify entities and their properties. The hypothesis is that this kind of structured data can improve entity search on the web. In order to test this hypothesis and to consistently progress in this field, a standardized evaluation is necessary. In this work, we discuss an evaluation campaign that specifically targets entity search over Linked Data by the means of keyword queries, including both queries that directly mention the entity as well as those that only describe the entities. We also discuss how crowd-sourcing was used to obtain relevance assessments from non-expert web users, the participating systems and the factors that contributed to positive results, and how the competition generalizes results from a previous crowd-sourced entity search evaluation.

Categories and Subject Descriptors

H.3.3 [**Information Storage Systems**]: Information Retrieval Systems

General Terms

Performance, Experimentation

Keywords

entity search, search engines, retrieval, evaluation

1. INTRODUCTION

In any new information retrieval task, one of the most common components needed is regular and standardized evaluation in order to determine if progress is being made, and this is increasingly important as large-scale amounts of structured data about entities is being introduced to the Web. While common information retrieval search technique used by Web search engines rely primarily on information implicit in the textual descriptions of web-pages and the structure of the links between web-pages, an increasing amount of data is available on the Web using the RDF (Resource Description Format) standard, which attempts to make explicit information about entities and the relations between them in a structured form as to create what has been termed the "Semantic Web" by Tim Berners-Lee, the inventor of the World Wide Web [2]. A number of entity-centric search engines have independently arisen that search and index this data, but they have not been systematically evaluated using information retrieval metrics until the SemSearch competition in 2010 [9]. However, this competition was criticized as being too simplistic by focusing only on keyword queries for named entities. We focus here on the second round of the competition, which broadened the kinds of queries tested from simple keyword-based queries for entities to complex queries that contained criteria that matched multiple entities. The process of how crowd-sourced judges were used to determine entity-based relevance is detailed, and the systems that participated are described, with a focus on the factors that led to their success in searching for entities.

EOS, SIGIR 2011 workshop, July 28, Beijing, China

2. RDF AND THE SEMANTIC WEB

Semantic Web languages based on RDF identify entities by assigning URIs (Uniform Resource Identifiers), such as *http://www.example.org/Paris*, to both entities and their relationships to other entities. So, one could describe the fact that 'Paris is the capital of France' in RDF by stating the triple *(http://example.org/Paris , http://example.org/capital, http://example.org/France)*. The first URI is the *subject*, the second URI is the *predicate*, and the third (either another URI or a string with a data-value) is *object*. RDF is then a graph model where the vertices may be either URIs that name entities or text, and the edges between vertices are also labelled with URIs. When this RDF data is accessible over HTTP at these URIs, and so accessible to search engines, this RDF data is called "Linked Data"[1]. Overall, the amount of Linked Data accessible to search engines has

[1] `http://www.linkeddata.org`

grown to 10s of billions RDF statements. Further technologies, such as reasoning about RDF entities and classes using schemas, are also available. The hypothesis of the Semantic Web is that by creating a clearly defined format for sharing structured data across the Web a number of common tasks can be improved for the benefit of end users.

2.1 Entity Search on the Semantic Web

The search for entities is the most common search type on the Web after navigational searches [15]. However, searching entities in the predominant textual content of the web is hard, because it requires error-prone and expensive text processing techniques such as information extraction, entity identification, entity disambiguation over large collections of documents. One of the main goals of the Semantic Web is to make information available in a structured form that is easier and more reliable for machines to process than documents. The hypothesis is that this kind of structured data can improve entity search on the Web. In order to test this hypothesis and to consistently progress in this field, a standardized evaluation is necessary.

3. OVERVIEW OF THE CHALLENGE

As noted earlier, while there is an increasing amount of data about entities on the Web encoded in RDF and accessible as Linked Data, and a growing number of independent 'Semantic search' engines that specialize in crawling and searching RDF such as Sindice [13]. Assessing the relevance of the results provided by these *semantic search engines* require an information retrieval evaluation methodology over realistic data-sets. The first large-scale evaluation of these Semantic Search engines took place in 2010 [9], focusing on the task of entity search. This choice was driven by the observation that over 40% of queries in real query logs fall into this category [15], largely because users have learned that search engine relevance decreases with longer queries and have grown accustomed to reducing their query (at least initially) to the name of an entity. However, the major feedback and criticism of the 2010 SemSearch Challenge was that by limiting the evaluation to keyword search for named entities the evaluation excluded more complex searches that would hypothetically be enabled by semantic search over RDF. Therefore, the 2011 SemSearch competition introduced a second track, the "List Search" track, that focussed on queries where one or more entities could fulfill the criteria given to a search engine.

The Semantic Search challenge differs from other evaluation campaigns on entity search. In comparison to the TREC 2010 Entity Track [1], the SemSearch Challenge searches over structured dat a in RDF rather than text in unstructured web-pages and features more complex queries. Likewise, in comparison to the INEX Entity-Ranking task [7], SemSearch focusses on RDF as opposed to XML as a data-format, and searches for relevance over entire RDF descriptions, not passages extracted from XML. Unlike the QALD-1 Question Answering over Linked Data [16] task, our queries were not composed of hand-crafted natural language questions built around particular limited data-sets such as DBPedia and MusicBrainz (i.e. RDF exports of Wikipedia and music-related information), but of both simple and complex real-world queries from actual query logs. The use of queries from actual Web search logs is also a major difference between our competition and all aforementioned competitions such as TREC and INEX. Keyword search over structured data gets also more attention in the database community [10] and an evaluation framework was recently proposed [6], but an standardized evaluation campaign is not yet available. The Semantic Search Challenge comprised two different tracks, which are described in the next section.

3.1 Entity Search Track

The Entity Search Track aims to evaluate a typical search task on the web, keyword search where the keyword(s) is generally the name of the entity. Entities are ranked according to the degree to which they are relevant to the keyword query. This task has been the same as defined for the 2010 Semantic Search Challenge [9].

3.2 List Search Track

The List Search Track comprises queries that describe sets of entities, but where the relevant entities are not named explicitly in the query. This track was designed to encourage participating systems to exploit relations between entities and type information of entities, therefore raising the complexity of the queries. The information need is expressed by a number of keywords (minimum three) that describe criteria that need to be matched by the returned results. The goal is to rank higher the entities that match the criteria than entities that do not match the criteria. Examples of the queries used in the two tracks are shown in Table 1 and described in the next section.

4. EVALUATION METHODOLOGY

Evaluating different approaches against each other requires a controlled setting in order to achieve comparability as well as repeatability of the evaluation's results. For the evaluation of ranked results, the *Cranfield* methodology [5] is the de-facto standard in information retrieval. The Cranfield methodology measures retrieval effectiveness by the means of a fixed document collection, a set of queries, a set of relevance assessments denoting which documents are (not) relevant for a query, and a number of well-understood and defined metrics, such as precision and recall. As we are evaluating search over entities in the RDF data format, our evaluation setting requires as the result of each participating system a ranked list of entities from an RDF data collection in response to each queries. Thus, the units of retrieval are individual entities identified by URIs (Uniform Resource Identifier), not documents. In the following sections, we describe how we created a standardized setting consisting of a RDF data collection, a set of keyword queries garnered from real-world Web search logs, and crowd-sourced relevance assessments.

4.1 Data Collection

A standard evaluation data collection should be not biased towards any particular system or towards a specific domain, as our goal is to evaluate general purpose entity search over RDF data. Therefore, we needed a collection of documents that would be a realistically large approximation to the amount of RDF data available 'live' on the Web and that contained relevant information for the queries, while simultaneously of a size that could be manageable by the resources of a research groups. We chose the 'Billion Triples Challenge' 2009 data set, a data-set created for the

Semantic Web Challenge [3] in 2009. The dataset was created by crawling data from the web as well as combining the indexes from several semantic web search engines. The raw size of the data is 247GB uncompressed and it contains 1.4B RDF statements describing 114 million entities. The statements are composed of *quads*, where a quad is a four tuple comprising the four fields *subject*, *predicate*, *object*, as is standard in RDF, but also a URI for *context*, which basically extends a RDF triple with a new field giving a URI that the triples were retrieved from (i.e. hosted on). Details of the dataset can be found at `http://vmlion25.deri.ie/` and it is available for download at `http://km.aifb.kit.edu/ws/dataset_semsearch2010`. There was only a single a modification necessary for using this data-set for entity search evaluation which was to replace RDF blank nodes (an existential variable in RDF) with unique identifiers so that they can be indexed.

4.2 Real-World Query Sets

A realistic search scenario requires queries that approximate user needs. Therefore, we created a set of queries based on the Yahoo! Search Query Tiny Sample v1.0 dataset, which contains over four thousand real queries from Yahoo's US query log of January, 2009. Each query in the log is asked by at least three different users and long numbers have been removed for privacy reasons. The query log is provided by the Yahoo! Webscope program[2].

For the entity search task, we selected 50 queries which name an entity explicitly and may also provide some additional context about it, as described in [15]. In the case of the list search track, we hand-picked 50 queries from the Yahoo query log as well as from TrueKnowledge 'recent' queries[3]. The queries describe a closed set of entities, have a relatively small number of possible answers (less than 12) which are unlikely to change.

Although many competitions use queries generated manually by the participants, it is unlikely that those queries are representative of the kinds of entity-based queries used on the Web. For example, queries around religious beliefs are quite a high percentage of queries in real web search engine logs. Therefore, we manually selected queries by randomly selecting from the query logs and then manually checked that at least one relevant answer existed on the current web of linked data.

Table 1 shows examples from the query sets for both tracks. The entire query sets are available for download[4].

08 toyota tundra	gods who dwelt on Mount Olympus
Hugh Downs	Arab states of the Persian Gulf
MADRID	astronauts who landed on the Moon
New England Coffee	Axis powers of World War II
PINK PANTHER 2	books of the Jewish canon
concord steel	boroughs of New York City
YMCA Tampa	Branches of the US military
ashley wagner	continents in the world
nokia e73	standard axioms of set theory
bounce city humble tx	manfred von richthofen parents
University of York	matt berry tv series

Table 1: Examples queries from the Entity Query Set (left) and List Query Set (right).

[2] `http://webscope.sandbox.yahoo.com/`
[3] `http://www.trueknowledge.com/recent/`
[4] `http://semsearch.yahoo.com/datasets.php`

4.3 Crowd-Sourced Relevance Judgments

We used Amazon Mechanical Turk to obtain the relevance assessments. This has been shown to be a reliable method for evaluation purposes, producing a rank ordering of search systems equivalent to the ordering produced by expert human assessors for this task [4]. The human assessors were presented with a simple, human readable rendering of RDF triples about the entity shown as attributes and values in a HTML table, with URIs being truncated to their rightmost hierarchical component. The URI of the entity itself was not shown. The rendering showed a maximum of ten attribute-value pairs with RDF attributes given in the specification and text values in English language being given preference. Based on this presentation and the keyword query, the assessors had to decide on a 3-point scale, whether the entity is irrelevant, somewhat relevant, or relevant. For the List Search track, the workers were presented additionally with a reference list of correct entities in addition to the criteria itself, which was obtained through manual searching by the organizers. This was done as the queries were of such difficulty that many assessors may not know the answers themselves.

First, the top 20 results of all runs for each query were pooled. Despite a validation mechanism in the submission process, we encountered problems with lowercased or N-triple encoded URIs, which required additional manual cleanup. URIs that did not appear as a subject in the data collection were discarded. Each result is evaluated by 5 workers and a majority vote yields the final assessment. The pooled evaluation procedure resulted in 3887 assessments for track 1 and 5675 assessments for track 2, which are available at `http://semsearch.yahoo.com/results.php`. The workers were paid $0.20 per batch of 12 assessments, which took them typically one to two minutes to complete. This results in an hourly wage of $6-$12.

The payment can be rejected for workers who try to game the system. To assure the quality of the assessments, we mixed *gold-win* cases, which are considered perfectly relevant by experts, and *gold-loose* cases, which are considered irrelevant, into a batch of 12 tasks presented to a worker. Thereby, we could estimate the quality of the assessments. All workers missing a considerable amounts of the gold-cases were rejected and their tasks put back into the pool to be done by others. Furthermore, we measured the deviation from the majority and observed such factors as the time to complete a batch. Workers obviously too far off were rejected as well. A lesson learnt here was that workers could choose how many batches they complete, which made it hard to measure the deviations for workers, who completed only few batches. In the future, we will require a minimum number of batches to be completed by each worker before being paid, in order to increase the quality assurance.

5. ENTITY SEARCH TRACK EVALUATION

Four teams participated in both tracks. These teams were University of Delaware (**UDel**), Digital Enterprise Research Institute (**DERI**), International Institute of Information Technology Hyderabad (**IIIT Hyd**), and Norwegian University of Science and Technology (**NTNU**). Dhirubhai Ambani Institute of Information and Communication Technology (**DA-IICT**) participated additionally in the List Search Track.

Each team was allowed to enter up to three different submissions ('runs') per track, in order to experiment with different system configurations. A submission is an ordered list of URIs for each query in the *trec format* allowing us to use the *trec_eval* software[5] to compute the metrics. In total, 10 runs were submitted for the Entity Search Track and 11 runs for the List Search Track.

In the following sections, we briefly describe and characterize the systems for each track and report on their performance. Detailed system descriptions are available at the Challenge website[6].

In order to categorize the systems and illustrate their different approaches to the entity search task, two major aspects can be distinguished: (1) the internal model for *entity representation*, and (2) the *retrieval model* applied for matching, retrieval, and ranking. Before, we characterize the systems, we discuss these two major aspects.

Entity representation.

teams used a *quad* having the same subject URI as the representation of an entity. Only DERI deviated from this representation and took all quads having the same subject and their contexts as the representation as the representation of an entity. The applied representations of an entity can be characterized by four aspects, which describe how the specifics of the data are taken into account. The RDF data model makes a distinction between object and datatype properties. Datatype properties can be seen as *attribute-value* pairs, where the value is a literal value, usually a text string. In contrast, object properties are typed *relations* in the form of attribute-object pairs, where the object is the URI identifier of another entity rather than a literal value. Since URIs are used as identifiers, each URI has a *domain name*, which can be seen as one kind of provenance. Another provenance aspect is the *context*, which describes the source of the triple in the BTC dataset. The domain is different from the context because URIs with the same domain can be used in different contexts. Whether these aspects are considered, is illustrated in Table 2 as follows:

- *attribute-value:* Are the attribute-values of the triples used in the entity representation (yes + / no −)?
- *relations:* Are the relations to other entities considered (yes + / no −)? The relations are potentially exploitable for ranking, because they form the data graph by linking to other entities. If this information is not taken into account, the relations usually treated as additional attribute-value pairs.
- *domain:* Is the domain information used (yes + / no −)? Entities of a certain domain are some times boosted, because certain domains are considered a-priori as relevant or of high quality. Often entities from *dbpedia.org* are considered for a-priori boosting.
- *context:* Is the context information included in the entity representation (yes + / no −)? This information can be used as well to favor certain sources.

[5] http://trec.nist.gov/trec_eval/
[6] http://semsearch.yahoo.com

Retrieval model.

All participating systems used inverted indexes to manage their data. Still, the different approaches can be characterized by three main aspects, although a specific systems could use a combination of them, which are: (1) purely *text based* approaches using a 'bag-of-words' representation of entities and common ranking techniques build on TF/IDF or language models[11]. The main notion of these approaches are term statistics calculated from the text representation. (2) The second type are *structured based* approaches which consider the structure of the data and weighted properties differently. In contrast to the text-based approaches, entities are not seen as flat text corpora, but as structured retrieval units. (3) The third aspect denotes whether the structure of the entire data graph is used to derive query independent scores, e.g. by graph analytics like PageRank. Since this aspect uses the structure for a-priori query scores, we refer to them as 'query-independent structure-based' (Q-I-structured-based) approaches.

Table 2 gives an overview of the systems based on the characteristics introduced above.

5.1 Overview of Evaluated Systems

		UDel		DERI			NTNU		
	Run	VO	Prox	1	2	3	Olav	Harald	Godfrid
Entity representation	attribute-value	+	+	+	+	+	+	+	+
	relations	-	-	-	-	+	-	-	+
	domain	+	-	-	+	+	+	+	+
	context	-	-	+	+	+	-	-	-
Retrieval model	Text-based	+	+	+	+	+	+	+	+
	Structure-based	-	-	+	+	+	-	+	+
	Q-I-structure	-	-	-	-	+	-	-	-

Table 2: Feature overview regarding system internal entity representation and retrieval model

UDel:

Entity representation: All quads having the same subject URI constituted one entity. Terms extracted from these quads are simply put into one 'bag-of-words' and indexed as one document.
Retrieval model: An axiomatic retrieval function was applied by University of Delaware [8]. For run **UDel-Prox**, query term proximity was added to the model, which favors documents having the query terms within a sliding window of 15 terms. The third run **UDel-VO** promotes entities whose URI has a direct match to a query term.

DERI:

Entity representation: In contrast to the other systems, the Sindice system from DERI took all quads having the same subject and the same context as the description of an entity. Only entity descriptions comprising more than 3 quads were considered. This entity description is internally represented as a labeled tree data model with an entity node as the root, and subsequent attribute and value nodes. In addition, run **DERI-3** used the entire graph structure, so exploiting the relationships of any given entity when ranking.
Retrieval model: BM25MF, an extension of BM25F,

which allows fields to have multiple values was used by Sindice to rank entities for all runs. The second and winning run, **DERI-2**, applied additionally query specific weights, namely query coverage and value coverage. These weights indicate how well the query terms are covered by a root node, respectively value node, in the internal data model. The more query terms are covered by a node, the more weight is contributed to this node. In addition, query independent weights were assigned to attributes, whose URI contain certain keywords, e.g. *label*, *title*, *sameas*, and *name*. Run **DERI-3** used additionally the relations to compute query independent scores based on the graph structure.

IIIT Hyd:

Did not provide a system description.

NTNU:

Entity representation: NTNU used the *DBPedia* dataset in addition to the BTC to represent entities. An entity is represented by three sub-models, the first comprises all name variants of this entity in DBPedia, the second considers several attributes from DBPedia for this entity, and the third uses the data from BTC about this entity. On the syntactic level, all triples having the same subject URI were used for the models based on DBPedia. For run **NTNU-Olav**, the model based on the BTC used only literal objects and regarded them as one flat text representation. For the runs **NTNU-Harald** and **NTNU-Godfrid**, the model had two fields, the name field which contained values of attributes that mentioned the name of the entity, while all other attributes were put into the content field.

Retrieval model: Mixture language models were used to incorporate the different entity models in the retrieval function, while weights were applied for specific attributes of DBPedia. Run **NTNU-Godfrid** used *sameAs* (an equivalence link on the Semantic Web) relations to propagate scores, in order to rank directly related entities higher.

5.2 Entity Search Track Results

The retrieval performance for the submitted runs are given in Table 3. The numbers on precision at cutoffs (P5, P10) give an impression about the top of the returned result lists. Mean Average Precision (MAP) takes the entire ranked list into account and is based on the complete assessment pool. On average, there are 9.4 relevant entities per query with a standard deviation of 11. For four queries no system could deliver relevant entities, which shows that some queries were really hard. These were queries with number *q18*, *q24*, *q25* and *q29*, e.g. *q25*: "holland bakery jakarta".

Discussion of the Entity Search Track.

The semantic search task of finding entities in an large RDF graph has been addressed by a spectrum of different approaches in this challenge as shown by the diversity of the results. The basis for most system are still the well known Information Retrieval techniques, which yields acceptable results. However, the winning system from DERI is a specialized system, which adapted IR methods and tailored them to RDF. The key feature for success, shared by the two top ranked systems in this years challenge, is to take the proximity or coverage of query terms on individual attribute values into account. This is a consequent development step over last year's challenge, where weighting properties individual was the key feature for success. The general observation is that considering the particular pieces of the structured data yields higher performance over unstructured, text-based retrieval shows that search can benefit from more structure.

Participant	Run	P10	P5	**MAP**
DERI	2	0.260	0.332	**0.2346**
UDel	Prox	0.260	0.337	**0.2167**
NTNU	Harald	0.222	0.280	**0.2072**
NTNU	Godfrid	0.224	0.272	**0.2063**
NTNU	Olav	0.220	0.276	**0.2050**
UDel	VO	0.194	0.248	**0.1858**
DERI	1	0.218	0.292	**0.1835**
DERI	3	0.188	0.252	**0.1635**
IIIT Hyd	1	0.130	0.148	**0.0876**
IIIT Hyd	2	0.142	0.132	**0.0870**

Table 3: Results of the Entity Search Track.

Similar to last year, one of the main and promising features of the RDF data model, namely the ability to express and type the relations between entities was only used by one run from DERI, which did not exceed the other runs. Whether relations are actually not helpful for entity search on large scale datasets or whether the usage of the relations is not yet understood remains to be investigated in the future. The List Search Track was designed with the intention in mind to get the systems to consider the relations as well. How the systems addressed this task is described in the next section.

6. LIST SEARCH TRACK EVALUATION

In general the teams participated with the same systems in the List Search Track and adapted them only slightly to this new task, although the most high-performing system was specially designed for the List Track. The adaptions are mostly on query analysis and interpretation, because the queries were not just keywords but more complex descriptions in natural language, as described in Section4.2. The modifications as well as the additional system are described in the next section followed by the results for this track. The modifications as well as the additional system are described in the next section followed by the results for this track.

6.1 Overview of Evaluated Systems

Delaware:

The team from Delaware applied an NLP parser to process the queries for run **UDelRun1**, in order to find the target type of the entities. Only entities belonging to this type were considered as results. For the runs **UDelRun2** and **UDelRun3** the type information was manually expanded, because the automatic processing failed in some cases. Instead of the axiomatic retrieval function, model-based relevance feedback was applied for run **UDelRun3** [17].

DERI:

DERI participated with an identical system configuration in the List Search Track.

NTNU:

NTNU participated with a system especially designed for this track. The system used only the Wikipedia

dataset and mapped the results to entities in the BTC collection. The queries were analyzed and potentially reformulated using the Wikipedia Miner software [12], in order to find the primary entity of the query. The query was run against an index of Wikipedia abstracts to get a candidate list of Wikipedia articles. The outgoing links from these articles were expanded and the resulting articles were also added to the candidate list. Scores are added if an article occurs multiple times and articles with a direct relation to the principal entity are boosted. In contrast to run **NTNU-1**, the runs **NTNU-2** and **NTNU-3** used an additional boosting for articles belonging to a Wikipedia set that had more than a certain fraction of its set of members in the candidate list. Run **NTNU-3** also applied an additional boost based on *sameAs* links.

DA-IICT:

The system by DA-IICT used a text-based approach build on Terrier [14] which favored entities according to the number of query terms present in their textual description. Due to data loss, the queries were only run against a part of the BTC data collection.

6.2 List Search Track Results

The retrieval performance for the submitted runs are shown in Table 4. The metrics were computed the same ways as for the Entity Track. There are on average 13 relevant entities per query with a standard deviation of 12.8. The participating systems could not find relevant entities for 6 queries. These were the queries with numbers *q15*, *q23*, *q27*, *q28*, *q45* and *q48*, for example *q15: "henry ii's brothers and sisters"*.

Participant	Run	P10	P5	**MAP**
NTNU	3	0.354	0.356	0.2790
NTNU	2	0.348	0.372	0.2594
NTNU	1	0.204	0.200	0.1625
DERI	1	0.210	0.220	0.1591
DERI	3	0.186	0.216	0.1526
DERI	2	0.192	0.216	0.1505
UDel	1	0.170	0.200	0.1079
UDel	2	0.162	0.152	0.0999
IIIT Hyd	1	0.072	0.076	0.0328
IIIT Hyd	2	0.072	0.076	0.0328
DA-IICT	1	0.014	0.012	0.0050

Table 4: Results of the List Search Track.

Discussion of the List Search Track.

The List Search Track proved to be a hard task and may require different techniques compared to the Entity Search Track. Since this track was new, most teams participated with their systems built for the Entity Search Track and adapted to the task mainly by analyzing and interpreting the query. Still, the performances show that solutions can be delivered, although there is obviously room for improvement. The winning system by NTNU did not use the BTC data collection, but was built on the Wikipedia corpus and exploited the links between articles. Obviously, the plain links between articles are a valuable resource for search. Ideally, such algorithms could eventually be adopted to more general-purpose RDF structured data outside that of Wikipedia.

7. CONCLUSIONS

The Semantic Search Challenge started in 2010 with the task of (named) entity retrieval from RDF data crawled from the Web. Though this task is seemingly simple, because the query contains the name of the entity, it features many of the problems in semantic search, including the potential ambiguity of short-form queries, the varying degrees of relevance by which an entity can be related to the one named in the query and the general quality issues inherent to Web data. The List Search Track introduced this year presented an even harder problem, i.e. queries that don't explicitly name an entity, but rather describe the set of matching entities.

The general direction of our work will continue toward exploring search tasks of increasing difficulty. In addition, there are a number of open questions that may impact the end-user benefits of semantic search engines and would still need to be investigated. For example, the retrieval engines above do not attempt to remove duplicates, and may return different, redundant descriptions of the same entity multiple times. A semantic search engine should remove such duplicates or merge them. Similarly, the user experience is largely impacted by the explanations given by the search engines. Similar to how current text search engines generate summaries and highlight keyword matches, a semantic search engine should attempt to summarize information from an RDF graph and highlight why a particular result is an answer to the user's query.

8. ACKNOWLEDGMENTS

We acknowledge Yahoo! Labs for hosting the Semantic Search Challenge 2011 website and sponsoring the prizes. The costs of the evaluation have been sponsored by the European SEALS project, `http://www.seals-project.eu`.

9. ADDITIONAL AUTHORS

Thanh Tran Duc, Institute AIFB, Karlsruhe Institute of Technology, 76128 Karlsruhe, Germany ducthanh.tran@kit.edu

10. REFERENCES

[1] K. Balog, P. Serdyukov, and A. de Vries. Overview of the trec 2010 entity track. In *TREC 2010 Working Notes*, 2010.

[2] T. Berners-Lee and M. Fischetti. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper San Francisco, 1999.

[3] C. Bizer and P. Mika. The semantic web challenge, 2009. *Journal of Web Semantics*, 8(4):341, 2010.

[4] R. Blanco, H. Halpin, D. M. Herzig, P. Mika, J. Pound, H. S. Thompson, and D. T. Tran. Repeatable and reliable search system evaluation using crowd-sourcing. In *SIGIR*. ACM, 2011.

[5] C. Cleverdon and M. Kean. Factors Determining the Performance of Indexing Systems, 1966.

[6] J. Coffman and A. C. Weaver. A framework for evaluating database keyword search strategies. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 729–738, New York, NY, USA, 2010. ACM.

[7] G. Demartini, T. Iofciu, and A. P. De Vries. Overview of the inex 2009 entity ranking track. In *Proceedings of the Focused retrieval and evaluation, and 8th international conference on Initiative for the evaluation of XML retrieval*, INEX'09, pages 254–264, Berlin, Heidelberg, 2010. Springer-Verlag.

[8] H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In *SIGIR*, pages 480–487, 2005.

[9] H. Halpin, D. M. Herzig, P. Mika, R. Blanco, J. Pound, H. S. Thompson, and D. T. Tran. Evaluating Ad-Hoc Object Retrieval. In *Proceedings of the International Workshop on Evaluation of Semantic Technologies (IWEST 2010)*. 9th International Semantic Web Conference (ISWC2010), 2010.

[10] *Proceedings of the Second International Workshop on Keyword Search on Structured Data, KEYS 2010, Indianapolis, Indiana, USA, June 6, 2010*. ACM, 2010.

[11] C. D. Manning, P. Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[12] D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. In *Proc. New Zealand Computer Science Research Student Conf.*, volume 9, 2009.

[13] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello. Sindice.com: a document-oriented lookup index for open linked data. *IJMSO*, 3(1):37–52, 2008.

[14] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.

[15] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc Object Ranking in the Web of Data . In *Proceedings of the WWW*, pages 771–780, Raleigh, United States of America, 2010.

[16] C. Unger, P. Cimiano, V. Lopez, and E. Motta. QALD-1 Open Challenge, 2011. http://www.sc.cit-ec.uni-bielefeld.de/sites/www.sc.cit-ec.uni-bielefeld.de/files/sharedtask.pdf.

[17] C. Zhai and J. D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, pages 403–410, 2001.

Author Index

NOTES