Improving Related Entity Finding via Incorporating Homepages and Recognizing Fine-grained Entities

Youzheng Wu Chiori Hori Hisashi Kawai Hideki Kashioka
Spoken Language Communication Group,MASTAR Project
National Institute of Information and Communications Technology (NiCT)
2-2-2 Hikaridai, Keihanna Science City, Kyoto 619-0288, Japan
{youzheng.wu, chiori.hori, hisashi.kawai, hideki.kashioka}@nict.go.jp

Abstract

This paper describes experiments on the TREC entity track that studies retrieval of homepages representing entities relevant to a query. Many studies have focused on extracting entities that match the given coarse-grained types such as organizations, persons, locations by using a named entity recognizer, and employing language model techniques to calculate similarities between query and supporting snippets of entities from which entities are extracted to rank the entities. This paper proposes three improvements over baseline, i.e., 1) incorporating homepages of entities to supplement supporting snippets, 2) recognizing fine-grained named entities to filter out or negatively reward extracted entities that do not match the specified fine-grained types of entities such as a university, airline, author, and 3) adopting a dependency tree-based similarity method to improve language model techniques. Our experiments demonstrate that the proposed approaches can significantly improve performance, for instance, the absolute improvements of nDCG@R and P@1 scores are 8.4%, and 27.5%.

1 Introduction

Many user information needs would be better answered by presenting a ranked list of entities directly, instead of just a list of relevant documents. Based on this assumption, increasing attention has been devoted to related entity finding tasks that aimed at finding documents representing entities of a correct type that are relevant to a query. The TREC expert finding track (Nick, 2005), for example, focused on creating an ordered list of experts who have skills and experiments on a given topic. The INEX entity ranking task (Vries, 2007) studied at ranking Wikipedia entities given a query, in which target entity types are shifted from a single type of entity (person) to any Wikipedia category. The TREC related entity finding (REF) track (Balog, 2010) started in 2009, is defined as: Given an input entity, by its name and homepage, the type of the target entity[1], as well as the nature of their relation, described in free text, find related entities that are of a target type, standing in the required relation to the input entity. The REF task is also similar to a combination of the TREC list QA (Voorhees, 2003) and homepage finding (Hawking, 2001) tasks. In short, all these entity finding tasks generally aim at performing entity-oriented search tasks on the Web. This paper is concerned with the TREC REF track. Figure 1 shows an example of this.

```
<query>
<num>7</num>
<entity_name>Boeing 747</entity_name>
<entity_URL>clueweb09-en0005-75-02292</entity_URL>
<target_entity>organization</target_entity>
<narrative>Airlines that currently use Boeing 747 planes.</narrative>
</query>
```

Figure 1: Test query in TREC 2009 entity track.

The key challenge in the REF task involves entity ranking, that is, estimating the likelihood of the extracted entities being answer entities for a given query. Many related studies (Bron, 2010; Fang, 2010) have employed language model techniques to estimate the likelihoods of the extracted entities being answer entities via calculating similarities between query and supporting documents/snippets of entities. This technique may fail in cases where supporting documents/snippets of entities do not support their being answer entities.

[1]TREC 2010 limits the track's scope to searches for instances of the organizations, people, locations and product entity types.

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 174–182,
Chiang Mai, Thailand, November 8 – 13, 2011.

To improve the above approach, this paper first argues that candidate entities' homepages are important supplements to supporting snippets and should be effectively exploited. Homepage information is, however, ignored by many TREC participants' systems. Second, much of the work to date only extracts coarse-grained types of entities (such as people, organizations, locations and products specified in *target_entity* field as shown in Figure 1) by using entity repositories such as YAGO (Suchanek, 2007) or named entity recognizers (Ratinov, 2009), and then rank them. However, some queries specify fine-grained types of target entities in *narrative* fields, such as airlines in Figure 1. In these cases, fine-grained entity recognition is necessary and helpful for improving performance, which can recognize fine-grained named entities such as airlines, publishers, drivers, or newspapers. Third, a dependency tree-based similarity approach is implemented to substitute language model techniques, which proved superior to the latter.

The contributions of this paper include 1) incorporating homepages of entities, and 2) recognizing fine-grained types of entities for improving entity ranking. Furthermore, we propose an unsupervised method of generating training examples for fine-grained entity recognition and exploit multiple-contexts of entities as classification features. In related studies, only single-contexts of entities are employed. The experimental results in terms of the TREC 2010 entity track test data set demonstrate that the nDCG@R improvements of our three proposals, i.e., dependency-tree similarity, incorporating homepage and recognizing fine-grained named entity components, are 2.3%, 4.1%, and 2.1%, respectively. Compared with baseline, the accumulative improvements of our REF system in terms of nDCG@R, P@1 and P@5 scores are 8.4%, 27.5%, and 12.0%, respectively.

2 Related Work

The TREC REF task is highly related to a combination of the TREC list QA and homepage finding, INEX entity ranking, and TREC expert search tasks. The TREC list QA task (2001-2007) (Voorhees, 2003) required systems to assemble an unordered list of answer strings to factoid questions such as *Who are six actors who have played Tevye in "Fiddler on the Roof"?* The underlying information need is of a more informational nature. However, the REF task is situated in explorative search tasks. Moreover, the list QA task also does not require returning to the homepage for each answer string. In recent years, retrieval-based (Yang, 2003), pattern-based (Ravichandran, 2002), deep NLP-based (Moldovan, 2002; Harabagiu, 2003), and supervised/unsupervised machine learning based approaches (Ittycheriah, 2002; Wu, 2007) have been proposed. The TREC homepage finding task (2001-2003) assumes that incoming queries (like "IJCNLP 2011") are attempts to navigate to the homepage of a particular web site (`http://www.ijcnlp2011.org/`).

The TREC expert search task (2005-2008) (Nick, 2005) focused on creating an ordered list of experts who have skills and experiments on a specific topic with enterprise data. Most of the proposed approaches generally fall into two categories: generative language models and discriminative models. For example, Balog (2006) proposed profile-centric (directly models the knowledge of an expert from associated documents) and document-centric (locates documents on the topic and then finds the associated experts) generative language models (LMs). Cao (2005) proposed a two-stage language model consisting of a document relevance and co-occurrence model. There are many other generative probabilistic models such as (Fang, 2007; Serdyukov, 2008). Fang (2010) proposed a principled relevance-based discriminative model that integrates a variety of document evidence and document candidate association features for improving expert searching.

The INEX entity ranking task (2007-2010) (Vries, 2007) studies ranking of Wikipedia entities to a query topic. Apart from estimating similarities between Wikipedia pages and the given query topic, many systems (Pehcevski, 2008) have exploited Wikipedia link structure and Wikipedia categories, for instance, estimating overlap between the set of categories associated with target Wikipedia pages and the categories specified in a given query topic.

The TREC REF task (2009-2010) (Balog, 2010) aims at entity-oriented search on the Web. The most typical system is a cascade of the following components. (1) Document Retriever retrieves top relevant documents to a given query from the given Clueweb09 collection with 503 million English pages. (2) Entity Extractor extracts candidate entities that match the given target types from the

	Types of Anwers	Source of Answers	Entity Extraction	Main Models used
List QA	Noun phrase	Newspaper texts	Needed	IR-based, NLP-based, and machine learning-based models
Expert Search	Person only	W3C corpus	No	IR-based model
INEX Entity	Any Wikipedia category	Wikipedia data	No	IR-based model with Wikipedia category and link
TREC REF	Location, person, organization, and product	Clueweb09, a snapshot of the Web	Needed	IR-based model due to the task is originally situated in a search problem

Table 1: Comparison of entity ranking tasks. W3C corpus is a simulation of enterprise data crawled from public W3C (*.w3.org) sites in June 2004.

top relevant documents by using entity repositories such as Wikipedia, or using named entity recognizers. (3) Entity Ranker estimates the probabilities of the extracted entities being answer entities by using supporting documents and/or snippets in which entities and queries co-occur. A number of language modeling techniques borrowed from expert search systems were employed (Bron, 2010; Fang, 2010; Li, 2010). (4) Homepage Finder assigns primary homepages for the top ranked entity names by using entity names as queries, or homepage identifiers.

Table 1 compares these tasks from four aspects.

3 Our System

We can see that TREC entity ranking task is very complicated, and each component is an independent research topic in the fields of NLP and IR. This paper cannot cover all of them, and only focuses on Entity Ranker component, that is, given a query Q, and a list of extracted entities $E = \{e_i|i = 1, 2, ...n\}$ associated with their homepages $H = \{h_{e_i}|i = 1, 2, ..., n\}$, how to effectively rank these entities.

The other three components are beyond the scope of this paper. For better understanding of the REF system, we simply introduce them. Our Document Retriever first employs Yahoo! BOSS API[2] to search relevant pages from the Web and then map them to documents in Clueweb09. Since one lesson from TREC 2009 is that commercial search engines such as Yahoo! are generally superior in locating relevant documents for the search engine, we used the Indri tool for building. In Entity Extractor, an NER tool developed at UIUC (Ratinov, 2009)[3] is employed. In particular, phrases/words tagged with PER, ORG, LOC and MISC tags are extracted when the target entities are people, organizations, locations, and products, respectively. For Homepage Finder, the DBpedia homepage data[4] is used to train a binary classifier and features are similar to (Upstill, 2003). It is noted that we reverse the sequence of the Entity Ranker and Homepage Finder to enable incorporating homepage for ranking (introduced in section 3.3), that is, we first assign homepage for each entity, and then rank them.

3.1 Baseline

In the context of expert search, the task is to find out what is the probability of a candidate person being an expert to a query. The REF system can be simply regarded as the task of estimating $p(e_i|Q)$, the probability of an entity e_i being answer entity given a query Q. Therefore, approaches proposed in expert search can be used for entity finding. In TREC expert search, document model (referred as Model 2) (Balog, 2006) turned out to be one of the most prominent and effective models for estimating $p(e_i|Q)$. Model 2 is also used as our baseline, which can be expressed by,

$$p(e_i|Q) \propto \sum_{j=1}^{n} p(Q|es_{ij}) * p(e_i|es_{ij}, Q) \quad (1)$$

In (1) es_{ij} stands for the j-th supporting snippet from which entity e_i is extracted, n is the number of supporting snippets, $p(Q|es_{ij})$ denotes the relevance between query and supporting snippet, and can be relatively easy to determine using a language model (the KL-divergence language model used in this paper), $p(e_i|es_{ij}, Q)$ denotes the co-occurrence of the query and entity in the snippet. Because unique characteristics of the W3C corpus used in expert search, meta-based co-occurrence

[2] http://developer.yahoo.com/

[3] http://l2r.cs.uiuc.edu/~cogcomp

[4] http://dbpedia.org/About

model is commonly used. In entity ranking task, the KL-divergence language model is adopted to calculate $p(e_i|es_{ij}, Q)$. Model 2 can be further improved in the context of REF task as follows.

3.2 Improvement 1: Dependency Tree-based Similarity

To improve unigram KL-divergence language model that can not capture relations between query words, this paper adopts a dependency tree-based similarity algorithm to calculate $p(Q|es_{ij})$, which can be expressed by,

$$p(Q|es_{ij}) \quad \propto \quad \frac{DP_Q \cap DP_{es_{ij}}}{\sqrt{|DP_Q| \times |DP_{es_{ij}}|}} \qquad (2)$$

where DP_Q and $DP_{es_{ij}}$ stand for a set of sub-trees generated from dependency trees of query Q and text snippet es_{ij}, respectively. Dependency trees are obtained by parsing Q and es_{ij} using Lin's dependency parser, Minipar[5], and the subtree is defined as any node up to its two descendants and extracted with the Freqt toolkit[6].

3.3 Improvement 2: Incorporating Homepage

The goal of the REF task is to return homepages representing entities to a query, and homepages sometimes contain valuable information for ranking. Therefore, it is easy and necessary to incorporate homepages of entities in ranking. As input in entity finding, we receive a query Q, a list of candidate entities $E = \{e_i|i = 1, 2, ..., n\}$ associated with their homepages $H = \{h_{e_i}|i = 1, 2, ..., n\}$. The Entity Ranker can be reformulated to estimate a conditional probability $p(e_i, h_{e_i}|Q)$. The top k entities with their homepages are deemed the most probable answer entities.

By assuming entity e_i is independent of its homepage h_{e_i}, we obtain,

$$p(e_i, h_{e_i}|Q) \; = \; p(e_i|Q) \times p(h_{e_i}|Q) \qquad (3)$$

where $p(e_i|Q)$ stands for the probability of entity e_i being an answer given query Q, and can be calculated using Equation (1) and (2), $p(h_{e_i}|Q)$ stands for the probability of homepage h_{e_i} being an answer given query Q.

By applying the Bayes' Theorem and assuming that $p(h_{e_i})$ is uniform for all homepages h_{e_i}, we obtain,

$$p(h_{e_i}|Q) = \frac{p(Q|h_{e_i}) \times p(h_{e_i})}{p(Q)} \propto p(Q|h_{e_i}) \qquad (4)$$

In some cases, homepages such as that of racecar driver Michael Schumacher (`http://www.michael-schumacher.de/`) do not contain any valuable information but intend to greet visitors and provide information about the site or its owner. Thus, we retrieve text snippets hs_{e_i} from a homepage site using query Q to build a back-off model for $p(h_{e_i}|Q)$ built from the homepage (the opening or main page of the homepage site). Finally, we can obtain,

$$p(Q|h_{e_i})' = \alpha \times p(Q|h_{e_i}) + \beta \times p(Q|hs_{e_i}) \qquad (5)$$

where $\alpha + \beta = 1$, $p(Q|h_{e_i})$ and $p(Q|hs_{e_i})$ are estimated using the KL-divergence language model.

In short, conditional probability $p(e_i, h_{e_i}|Q)$ of the likelihood of entity e_i with its homepage h_{e_i} being answer is calculated by using Equation (3), (5), (1) and (2).

3.4 Improvement 3: Fine-grained Entity Recognition

As mentioned, entities are extracted by using NER tool. There exist two problems. First, the NER tool can only identify coarse-grained types of entities such as organizations or locations. However, users' queries sometimes specify fine-grained types of named entities such as airlines, universities, or actresses. Second, many incorrect entities are extracted. The main reason lies in: the NER tool is trained on newspapers, but we use it to tag web data. Therefore, it is necessary to filter out or negatively reward entities that do not match the fine-grained entity types if specified in queries. For example, this step can hopefully remove or negatively reward the extracted entities that are not airlines for the TREC 2009 test query shown in Figure 1.

Many semi-supervised methods have been proposed to recognize fine-grained types of entities. For example, Hearst (1992) used lexical patterns such as "X, such as Y". Fleischman (2002) employed a supervised learning method that considered the local context surrounding the entity as well as global semantic information. Etzioni (2005) started with a set of "*predicates*" and bootstrapped the extraction process from high-precision generic patterns. Oh (2009) exploited

[5] `http://webdocs.cs.ualberta.ca/~lindek`
[6] `http://chasen.org/~taku/software`

Wikipedia structure information and textual context to determine fine-grained types of Wikipedia entities. Generally, these methods mainly exploit the single-context of an entity as classification feature, which may result in errors in cases in which the relation between entity and its fine-grained type is not explicitly expressed.

Our proposal differs: 1) we utilize multiple contexts in which entities and their fine-grained types co-occur, 2) multiple contexts obtained by querying the Web with entities and their fine-grained types are helpful to disambiguate entities, 3) a dependency pattern-based approach is proposed for fine-grained classification of named entities. More specifically, our goal is to assign a class label ("yes" or "no") for each ⟨entity, fine-grained type⟩ pair. A "yes" means the entity belongs to the corresponding fine-grained type. Otherwise, it does not. The details are as follows.

3.4.1 Step-1: Preparation of Training Examples

A certain number of ⟨entity e, its fine-grained type fgt, multi-contexts they co-occur mc⟩ training triples are needed to build a classifier of fine-grained entities. The key challenge here is to prepare positive and negative $\langle e, fgt \rangle$ pairs.

A Wikipedia article usually starts with a definition sentence like "*Continental Airlines is an American airline based and headquartered in Continental Center I in downtown Houston, Texas.*" We find that it is practicable to automatically extract entity ("Continental Airlines" in this example) and its fine-grained type ("airline") from such well-formed sentences. To extract the pairs from these definition sentences, we first use Minipar to parse all Wikipedia definition sentences and then extract the pairs using heuristic rules such as *(be (Wikipedia-entity) (fine-grained-type))*. In this example, ⟨Continental Airlines, airline⟩ is extracted. Finally, 41,495 pairs are generated. These pairs will be used as positive instances. In order to construct negative training pairs, we first adopt the NER tool to recognize named entities in the Wikipedia definition sentences, and then pair the fine-grained type and the identified entities, except for the Wikipedia entity, as negative examples. For example, ⟨Continental Center I, airline⟩, ⟨Houston, airline⟩, and ⟨Texas, airline⟩ pairs are generated. Finally, 122,686 negative pairs are collected from Wikipedia.

Multi-context mc can be easily obtained by querying a search engine with the entity and its type and merging the first k snippets returned. Formally, $mc = \bigcup_{j=1}^{k} s_j$, where s_j denotes the j-th snippet. For ambiguous entities such as "Michael Collins", it is hard to recognize their fine-grained types with fewer frequencies from multiple contexts obtained by querying the Web with entities only. Multiple contexts learned with entities and their fine-grained types can partially solve this problem.

3.4.2 Step-2: Building Classifier

In order to handle long distance relations between words, dependency patterns are extracted as features for classification. First, textual contexts of $\langle e, fgt, mc \rangle$ triples are parsed using Lin's Minipar. Then, the shortest dependency paths between e and fgt are extracted as dependency patterns. Figure 2 shows two examples. To reduce the dimensionality of the feature space, we calculate the precision of each dependency pattern by using the equation, $precison = Cnt_p/(Cnt_p + Cnt_n)$, where, Cnt_p and Cnt_n denote total numbers of patterns occurring in positive and negative triples, respectively. We sort the extracted dependency patterns in deceasing order of precision and empirically select the top 500 patterns as classification features.

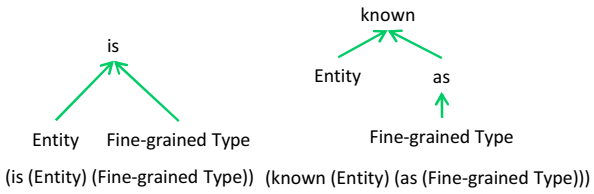


Figure 2: Examples of dependency patterns.

For the classifier, we employ multivariate classification SVMs that can directly optimize a large class of performance measures such as F_1-Score, prec@k and rec@k (the precision and recall of a classifier that predicts exactly k = 100 examples to be positive) (Joachims, 2005). For our experiment we held out 500 pairs from each of the positive and negative instances for testing. The remainder are used for training. Table 2 reports the results on the testing data. These results are quite promising. The classifier optimizing F_1-Score is finally used in our REF system.

3.4.3 Step-3: Using Classifier

To use the classifier in the REF system, we recognize fine-grained type fgt_Q of the target entity

	Rec@k	Prec@k	F1-score
Precision	80.8	89.3	86.4
Recall	97.4	83.6	91.6
F-measure	88.3	86.4	88.9

Table 2: Fine-grained named entity classifiers optimizing different measures.

from the $narrative$ field of query Q according to the predefined heuristic rules such as the head of the first non-stop noun phrase being fine-grained. For exmaple, $gallery$ in "*What art galleries are located in Bethesda, Maryland?*" is identified as fine-grained type.

For each entity e_i extracted via the Entity Extractor, the following steps are performed. (1) obtain textual contexts by querying the Yahoo! search engine with entity e_i and the identified fine-grained type fgt_Q, and merging the Yahoo! snippets returned. (2) parse contexts using Lin's Minipar and extract dependency patterns between e_i and fgt_Q. (3) employ the classifier to determine whether the entity e_i belongs to the fine-grained type fgt_Q, and remove or negatively reward the entities that are not fine-grained type identified from the query.

4 Experiments

Our experiments are conducted in the context of the TREC 2010 REF task. Relevance judgements in the TREC were performed in two stages. In phase one, all participant systems were pooled to a depth of the 20. The submitted homepages were judged on a three-point relevance scale: (2) primary homepage devoted to and in control of the entity, (1) relevant homepage devoted to the entity, but is not in control of the entity, and (0) non-relevant homepage that only mentions the entity but is not about the entity. Note that the Wikipedia page of a given entity is regarded as non-relevant by definition in TREC 2010. In phase two, homepages belonging to the same entity are grouped together. The test set used in the TREC 2010 entity track contains 50 test queries. In the official evaluation, only 47 test queries are used because no answers to the other three queries are found. Among 47 test queries, 31 are for organization, 7 for location, 8 for person, and 1 for product name. The average number of answered homepages per topic is 14 (Balog, 2010).

The TREC metrics are based on the homepages only because the ultimate goal of the REF system is to find the homepages of the entities. The main metric is nDCG@R; that is, the normalized discounted cumulative gain at rank R (the number of primary and relevant homepages for that topic) where a record with a primary gets a gain of 3, and a record with a relevant gets a gain of 1. We also report P@N, that is, the fraction of primary homepages in the first N ranks. Experimental results are computed using the *eval-entity.pl* script released by TREC.

4.1 Overall Performance

Table 3 reports the results of the four runs. Best$_T$ and Median$_T$ denote the best and median scores among all TREC 2010 participants' systems, respectively. Perfect$_O$ means the manual ranking system, which can indicate the performance ceiling that our Entity Ranker can achieve. Our$_{comb}$ denotes the proposed system that negatively reward all entities not belonging to the fine-grained entity type by simply putting them at the end of the ranking list.

	nDCG@R	P@1	P@5	P@10
Our$_{comb}$	.1865	.3404	.20	.1596
Perfect$_O$	.3564	.8298	.5872	.3787
Median$_T$	.12	-	-	-
Best$_T$	.38	-	-	-

Table 3: Comparison of four runs.

The results demonstrate that: i) Our$_{comb}$ significantly improves the median performance of the TREC participant systems from 12% to 18.65% in terms of nDCG@R. However, Perfect$_O$ (the performance ceiling) is much high than our automatic system, Our$_{comb}$. This means that there is still much room for improving the entity ranking component. ii) Figure 3 shows the performance of each target type. Product-type queries achieve a worse score due to poor product (PRO) name recognition of the NER tool. The best P@1 score is obtained for organization (ORG) type queries. The nDCG@R score for the person (PER) type is, however, better than that for ORG-type queries. This is because the average number of answer homepages for the ORG-type (29.5) is significantly larger that that of the PER-type (10.5), and the recall for ORG type queries is relatively poor. iii) The Best$_T$ (Yang, 2010)

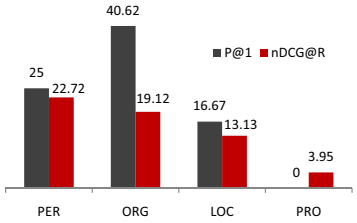


Figure 3: Results per topic type.

is even better than our Perfect$_O$. This indicates that the recall of our entity extraction is unsatisfactory. Note that this paper is mainly concerned with entity ranking, and entity extraction is not the scope of this paper. Yet the proposed methods can be incorporated into Best$_T$ and it can be expected to further improve its performance. Because Best$_T$ only use co-occurrence information between entities in ranking. For better understanding, Table 4 analyzes the recalls of the answer entities in the Document Retriever and Entity Extractor modules. #que represents the number of queries in which at least one answer entity is contained. #ent represents the number of answer entities of all test queries contained. This table indicates that the recall of the Entity Extractor is only 37% (= 266/715).

	Golden Answers [a]	Document Retriever [b]	Entity Extractor [c]
#que	47	45	40
#ent	715	384	266

[a] Golden answers are proved by TREC 2010 `http://trec.nist.gov/data/entity10.html`
[b] No answer entities are retrieved for queries 34 and 44.
[c] No answer entities are extracted for queries 28, 36, 37, 65, and 66.

Table 4: Recalls of answer entities.

The following sections mainly analyze the impacts of the dependency tree-based similarity, incorporating homepage and the fine-grained entity recognition to the REF system; thus, we exclude the queries for which no answer entities are extracted in the Entity Extractor, and the following experiments are based on 40 queries of the TREC 2010 test set.

4.2 Impact of Homepage and Fine-grained Entity Recognition

Table 5 shows the contributions of the dependency tree-based similarity (DTBS), incorporating homepage information (HP), and fine-grained named entity recognition (FG-NER). The baseline is Model 2 discussed in section 3.1. Significance tests are conducted. †: significantly better than the system without this component at the $p = 0.05$ level using two-sided t-tests; $^\flat$: significantly better at the 0.01 level.

	nDCG@R	P@1	P@5	P@10
Baseline	.1336	.125	.115	.1075
+DTBS	.1562†	.25†	.135	.1225
+HP	.1969$^\flat$	.20	.20$^\flat$	.175$^\flat$
+FG-NER	.2181†	.40$^\flat$	.235	.1875

Table 5: Contribution of each component.

The experimental results indicate that: i) the DTBS method can greatly improve Baseline, e.g., the nDCG@R and P@1 scores are significantly improved by 16.9% and 100.0%, respectively. We expect this because the DTBS method considers the relation between words. ii) homepage (HP) can positively impact the REF system in terms of nDCG@R, P@5, and P@10 metrics, which, however, leads to a lower P@1 score. When a non-homepage but one highly-related to the query is assigned as the homepage of an incorrect entity, incorporating homepage information will cause a negative influence. iii) the FG-NER can greatly improve the P@1 score from 20.0% to 40.0% and the P@10 score by 7.1% (not significant). This indicates that the FG-NER on TREC answer entities has nice precision and but poor recall. Table 2, however, shows that our FG-NER can achieve promising precision and recall. It is hard for TREC non-famous entities to retrieve snippets from the Web that conform to the dependency patterns extracted from snippets of Wikipedia entities, which results in poor performance.

In short, it is effective to use a dependency tree-based similarity, homepage information, and fine-grained named entity recognition in the REF system. Figure 4 shows the nDCG@R scores of Our$_{comb}$ and Baseline for each of the 40 queries.

4.3 Evaluation on Entity Names

The experiments above are based on homepages only in which correct entities with wrong homepages are not rewarded. This section discusses the performance based on named entities in which failures of finding homepage are ignored. In calculating nDCG@R, each answer entity gains 1 and a

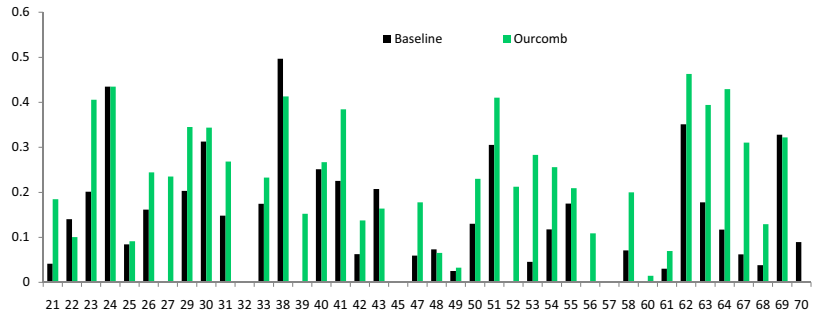
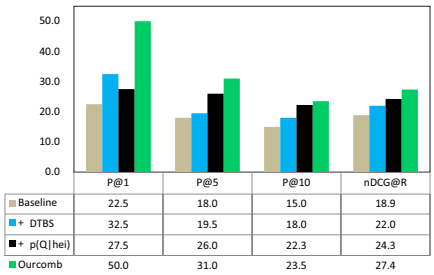


Figure 4: nDCG@R score for each test query.

non-relevant entity gains 0. Figure 5 shows the performance. This figure indicates that the improvements from each proposed component are more significant when errors from homepage finding are ignored. For example, the absolute enhancements of the FG-NER in terms of P@1 and nDCG@R scores are 22.5%, and 3.1%, respectively. This experiment indicates that the Homepage Finder component needs to be improved.



	P@1	P@5	P@10	nDCG@R
Baseline	22.5	18.0	15.0	18.9
+ DTBS	32.5	19.5	18.0	22.0
+ p(Q|hei)	27.5	26.0	22.3	24.3
Ourcomb	50.0	31.0	23.5	27.4

Figure 5: Metrics based on named entities.

5 Conclusion

This paper focused on developing a model for retrieving homepages of entities relevant to a query from a huge collection, and proposed three algorithms for improvements: a dependency tree-based similarity method, incorporating homepages of entities to supplement text snippets that the entities are from, and fine-grained classification of named entities. The comparison experiments on the TREC 2010 test data set showed that the proposed algorithms can significantly improve the system; e.g., the cumulative improvements of the nDCG@R, P@1, and P@5 scores over the Baseline reach 8.4%, 27.5%, and 12.0%, respectively. Moreover, our approaches can also be used in other tasks such as factoid QA. For example, in the TREC 2007 QA test set, about 50% questions (except for questions which answers are numeric and time expresses) contain fine-grained types of answers. Thus, our fine-grained entity recognition module can be expected to lead to improvements in QA systems.

In the future, we will work toward entity extraction and fine-grained named entity recognition. Table and list-based entity extraction may be essential due to the considerable number of answer entities scattered in tables, lists, and other structured forms. For example, answer entities to TREC 2010 query 29 (Find companies that are included in the Dow Jones industrial average.) are contained in a table at `http://www.1728.com/dowjone2.htm`. Li (2010) summarized the statistics of the TREC 2010 test queries in which answer entities are expressed in tables and lists. This means the NER tool trained by the newspaper corpus might fail at identifying the entities from tables and lists, and we have a great deal of work to do in order to correctly identify them. For fine-grained named entity recognition, more studies are needed on identifying fine-grained types of non-famous entities. For example, it is hard to determine whether "Rosenberg Gallery" is a gallery from the snippets relevant to the query "Rosenberg Gallery, gallery".

References

Abdessamad Echihabi and Daniel Marcu. 2003. A Noisy-Channel Approach to Question Answering. In *Proc. of ACL 2003*, Japan.

Abraham Ittycheriah, and Salim Roukos. 2002. IBM's Statistical Question Answering System-TREC 11. In *Proc. of TREC 2002*.

Arjen P. de Vries, Anne-Marie Vercoustre, James A. Thom, et al. 2007. Overview of the INEX 2007 Entity Ranking Track. In *Proc. of INEX 2007*.

Claudio Giuliano. 2009. Fine-Grained Classification of Named Entities Exploiting Latent Semantic Kernels. In *Proc. of CoNLL 2009*, pp. 201-209.

Dan Moldovan, Sanda Harabagiu, Roxana Girju, et al. 2002. LCC Tools for Question Answering. In *Proc. of TREC 2002*.

David Hawking, and Nick Craswell. 2001. Overview of the TREC 2001 Web Track. In *Proc. of TREC 2001*.

Deepak Ravichandran, and Eduard Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. In *Proc. of ACL 2002*.

Ellen M. Voorhees. 2003. Overview of the TREC 2003 Question Answering Track. In *Proc. of TREC 2003*, pp.54-68, USA.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *Proc. of WWW 2007*, pp.697-706.

Hui Fang and ChengXiang Zhai. 2007. Probabilistic Models for Expert Finding. In *Proc. of ECIR 2007*, pp.418-430.

Hui Yang, and Tat-Seng Chua. 2003. QUALIFIER: Question Answering by Lexical Fabric and External Resources. In *Proc. of EACL 2003*, pp.363-370.

John Lafferty and Chengxiang Zhai. 2001. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In *Proc. of SIGIR-2001*.

Jong-Hoon Oh, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Bilingual Co-Training for Monolingual Hyponymy-Relation Acquisition. In *Proc. of ACL 2010*, pp.432-440.

Jovan Pehcevski, Anne-Marie Vercoustre, and James Thom. 2008. Exploiting Locality of Wikipedia Links in Entity Ranking. In *Proc. of ECIR 2008*.

Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 2006. Formal Models for Expert Finding in Enterprise Corpora. In *Proc. of SIGIR 2006*, pp.43-50.

Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 2010. Overview of TREC 2010 Entity Track. In *Proc. of TREC 2010*.

Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proc. of CoNLL 2009*.

Marc Bron, Krisztian Balog, and Maarten de Rijke. 2010. Ranking Related Entities: Components and Analysis. In *Proc. of CIKM 2010*.

Marc Bron, Jiyin He, Katja Hofmann, et al. 2010. The University of Amsterdam at TREC 2010 Session, Entity, and Relevance Feedback. In *Proc. of TREC 2010*.

Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proc. of COLING-92*, pp.539-545.

Michael Fleischman and Eduard Hovy. 2002. Fine Grained Classification of Named Entities. In *Proc. of COLING-2002*.

Nick Craswell, Arjen P. de Vries, and Ian Soboroff. 2005. Overview of the TREC-2005 Enterprise Track. In *Proc. of TREC 2005*, pp.1-7.

Oren Etzioni, Michael Cafarella, Doug Downey, et al. 2005. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. *Artificial Intelligence*, Volume 165 Issue 1.

Pavel Serdyukov, Henning Rode, and Djoerd Hiemstra. 2008. Modeling multi-step relevance propagation for expert finding. In *Proc. of CIKM 2008*.

Qi Li and Daqing He. 2010. Searching for Entities: When Retrieval Meets Extraction. In *Proc. of TREC 2010*.

Qing Yang, Peng Jiang, Chunxia Zhang, and et al. 2010. Reconstruct Logical Hierarchical Sitemap for Related Entity Finding. In *Proc. of TREC 2010*.

Rianne Kaptein, Pavel Serdyukov, Arjen de Vries, and Jaap Kamps. 2010. Entity Ranking using Wikipedia as a Pivot. In *Proc. of CIKM 2010*.

Sanda M. Harabagiu, Steven J. Maiorano and Marius A. Pasca. 2003. Open-Domain Textual Question Answering Techniques. In *Natural Language Engineering* 9 (3): 1-38.

Thorsten Joachims. 2005. A Support Vector Method for Multivariate Performance Measures. In *Proc. of ICML 2005*.

Trystan Upstill. 2003. Query-Independent Evidence in Home Page Finding. In *ACM Transactions on Information Systems*, Vol21, No3, pp.286-313.

Yi Fang, Luo Si, and Aditya P. Mathur. 2010. Discriminative Models of Integrating Document Evidence and Document-Candidate Associations for Expert Search. In *Proc. of SIGIR 2010*, pp.683-690.

Yi Fang, Luo Si, Zhengtao Yu, et al. 2010. Purdue at TREC 2010 Entity Track: A Probabilistic Framework for Matching Types Between Candidate and Target Entities. In *Proc. of TREC 2010*.

Youzheng Wu, Ruiqiang Zhang, Xinhui Hu, Hideki Kashioka. 2007. Learning Unsupervised SVM Classifier for Answer Selection in Web Question Answering. In *Proc. of EMNLP-CoNLL 2007*.

Youzheng Wu and Hideki Kashioka. 2009. NiCT at TREC 2009: Employing Three Models for Entity Ranking Track. In *Proc. of TREC 2009*.

Yunbo Cao, Jingjing Liu, Shenghua Bao, and Hang Li. 2005. Research on expert search at enterprise track of TREC 2005. In *Proc. of TREC 2005*.

Yutaka Sasaki. 2005. Question Answering as Question-Biased Term Extraction: A New Approach toward Multilingual QA. In *Proc. of ACL 2005*.