Experiments on Related Entity Finding Track at TREC 2009

Qing Yang ,Peng Jiang, Chunxia Zhang, Zhendong Niu

School of Computer, Beijing Institute of Technology
{ yangqing2005,jp, cxzhang, zniu}@bit.edu.cn

Abstract. Our goal in participating in the TREC 2009 Entity Track is to study whether QA list technique can help improve accuracy of the entity finding task. Also, we take a looking for homepage finding to identify homepages of an entity by training a maximum entropy classifier and a logistic regression models for three types of entity respectively.

1. Introduction

This is Beijing Institute of Technology's first year participating in TREC. For related entity finding track, we mainly focus on employing pipeline architecture to model this track, indexing and retrieving by indri and making use of OpenNLP's ME classifier to identify extracted entities homepages.

2. Related Entity Finding Task

The related entity finding task is new to be proposed by NIST this year. This task is defined as the following:

Given an input entity, by its name and homepage, the type of the target entity, as well as the nature of their relation, described in free text, find related entities that are of target type, standing in the required relation to the input entity.

This task shares similarities with both expert finding (in that we need to return not "just" documents) and homepage finding (since entities are uniquely identified by their homepage). However, approaches to address this task need to generalize to multiple types of entities (beyond just people) and return the homepages of multiple entities, not just one. Also, the topic defines a focal entity to which returned homepages should be related. [1]

2.1. System Overview

We complete our experimental system architecture as pipeline architecture by devising from OpenEphyra's framework.

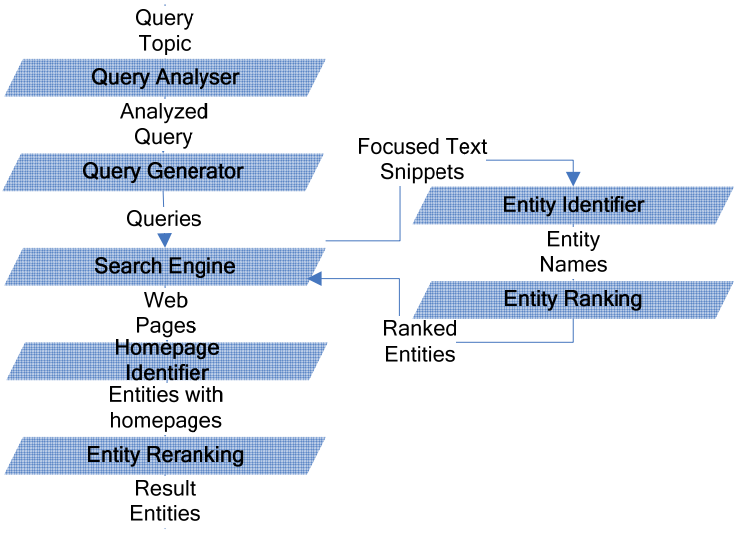

Fig. 1. The Related Entity Finding System Architecture.

We outline the retrieval framework as above. From TREC-supplied query topics, we first analyze the narrative of every query topic and extract keywords and terms. Second we employ BagofWordsGenerator and QueryReformationGenerator to rewrite query strings. Then we send query strings to the indri search engine, and get results. The granularity of results is focused text snippet rather than document. From the focused text snippets, we employ some OpenNLP components and Stanford's parser to identify target typed named entities. By counting number of occurrences in focused text snippets at sentence level, we rank entities by the reverse order of occurrences. We get the top 150 entities and post the top 150 entities' name to indri search engine respectively. Using Maximum entropy classifier to score the returned web pages as the related entity's homepage candidate, we rank the entity's homepage candidates by the scores in reverse order. Finally, we rerank those ranked entities by just filtering out those entities which have no homepages whose scores are above threshold.

2.2. Query Topic Parsing

As for a QA, Ephyra spends much effort to analyze question syntactically and semantically. To identify answer type, it employs machine learning scheme to train answer patterns and identify answer types. This tricky phase is not necessary for the related entity track because the target entity type is explicitly given. Also, OpenEphyra employs wordnet to expand query terms. From the expanded query string, it generates some irrelevant terms. Then we remove those irrelevant words manually to avoid topic drift. As for the explicit entity name, we just add it to the query string without expanding it. To take the first query topic for an example, OpenEphyra will generate two query strings. One is focus words i.e. blackberry

Carriers makes phones, weighted score 1.0. The query to send to indri is as following:

#combine [passage100:50](blackberry Carriers makes phones).

The passage length and increment size are set 100 and 50 separately experimentally. Variable lengths of window size will generate different results. It is challenging to decide the reasoned window size. The passage length constructed a context for locating related entities. It represents the proximity between the source entity and the target related entity.

The other is expanded terms, i.e. blackberry (Carriers OR toter OR bearer) makes (phones OR telephone OR "telephone set"). Obviously, toter and bearer are synonymous with carrier in wordnet. But it not suitable for this query topic. We just remove them manually. The converted query string is such as the following:

#combine [passage100:50](blackberry Carriers #or(phones telephone "telephone set")).

This query string has weighted score 1.5. The weight score is addresses as the degree how the generated query string matches the narrative of the query topic. It can be considered as a degree of proximity between target entity and input entity.

2.3. Named Entity Identification

In this task, the type of target entity is restricted in three types: person, organization and product. Generally speaking, the first two types are easier to identify from focused snippets. However, for the product type, it is rather difficult to be identified correctly. To deal with this issue, we resort to wikipedia online knowledge database whose pages always have a category label. We made a hardworking to find that those *introductions*, *productions*, *products*, *games*, *software*, *hardware* etc. category labels are almost classified into product type. It helps us to extract 43,393 product names. Also, by using the same method, we extracted 18,181 organization names and 118,002 person names.

In this experimental system, we employ OpenEphyra's NETagger to complete named entity identification. OpenEphyra's NETagger combines model-based, pattern-based and list-based named entity taggers. It is natural for us to add a product list to hope to improve product identification performance. As for the other easier to identify named entity types resort to StanfordNeTagger.

As for StanfordNeTagger, we load ner-eng-ie.crf-3-all2006-distsim.ser.gz serialized model which can label: PERSON, ORGANIZATION, and LOCATION entities. The model is trained on data from CoNLL, MUC6, MUC7, and ACE.

2.4. Related Entity Candidates Ranking

It is well known that the search results ranking is not necessary responding to those extracted entities ranking. We apply the following formula in a probabilistic model to rank related entity candidates.

$$Er = \alpha Qscore + \beta Rscore + \gamma Nredundancy$$

Q refers to Query String which represents the fitness of the generated query string to express the nature of related entity with the source entity. R refers to the search

Result, which represents the relevancy of searched result (in this context its granularity is passage) to the search string. N refers to Number of redundancy of the focused snippets which reside the same entity. α, β, γ are the coefficient respectively. In this experimental setting, it is simply to set all the coefficients 1. Besides that, the result's score are formalized to 1 if the result is in the top N. You may notice that the effect of the result ranking is taken into consideration implicitly for we extract entities from the result passages from top N. Those which are not in the top N are ignored definitely.

2.5. ME-Based Entity Homepages Classifier

We model the entity homepage identifying as a binary-class classifying problem. Our aim is to set the probability to represent the likelihood of one URL is a homepage rather than to make a binary decision – yes or no. [4] proposed a machine learning approaches to predict the correct homepage in response to a user's homepage finding query. He generates a binary decision tree to predict whether a URL is a homepage URL or not. Obviously, it is more suitable to employ probability than binary decision in this task. Table. 2.1 demonstrate those attributes.

Table. 2.1 Attributes vector

URL length	the number of characters in the URL
URL depth	the number of slashes in the UR
URL type	four types of URL: root, subroot, path, file. he type of URL which is proposed by UTwente/TNO in TREC-2001 homepage finding track
Entity in URL	whether the specified entity name is present in the URL
Variants of entity in URL	manually devise many variants of the entity name which are likely to be used by web designers and decide whether one of those variants exists in the URL
Position type	the position type refers to the above defined types of URL. The position type represents the entity name or its variants exists which part in the URL
Entity in page title	whether entity name or its variants exist in page's title
Keyword	whether page's title contains with a keyword; these keywords are "official", "home", "homepage"
Length of title	number of characters in title
Occurrence of entity in title	the number of entity name occurrence in the title

The ME model then is applied to the results returned by the mixture of context language models retrieval system, in hopes that we can filter out most of the irrelevant web pages in these returned webpage lists.

Additionally, we normalize URL by using BasicURLNormalizer which is extracted from nutch-0.9 before we extract type of URL. As for variants of entity name, we analyze homepage_en.nt from DBPedia and get the following rule to generate variants of the specified entity name.

Table. 2.2 Variant form rules

1.	Replace blank space with "_","+","%20|","" respectively
2.	Replace "'s" with "s", "", respectively
3.	Number of characters in Abbreviation is equal or greater then three
4.	Concatenate the first character of every word in the specified entity name including stopwords or excluding stopwords respectively.
5.	For two words in the specified entity name, combine first three to five characters of each word to generate abbreviations.

By using these variants of the specified entity name, it gets 99.9% in finding entity names in their homepages. It represents that web designers will always naming their homepage' URL from related entity names.

2.6. Logistic Regression Model for Homepage Finding

As for ME classifier, it is not easy to interpret the generated model from training materials. We leverage a logistic regression model for homepage finding also. To compare the effective performance the two models, the result is explained in the section 2.10.

2.7. Entity Homepage Finding

The procedure to identify the extracted entities' homepages involves two phrases. In the first phrase, we generated a Max entropy homepage classifying model to predict the probability of a URL is the specified entity's homepage. The second phrase is to employ a mixture of context language models, which can easily be expressed in the Indri query language to find which web page is most relevant to the specified entity. For example, for the entity name "BlackBerry", the following query will be constructed:

#wsum(5.0 #1(BlackBerry).(title) 3.0 #1(BlackBerry).(anchor) 1.0 #1(BlackBerry))

After send the constructed query to Indri, we get so many homepage candidates. Then, we employ beforehand generated homepage model to predict the probability of whether a webpage is the specified entity name's homepage or not. For the task is just

to return three URL as an entity name at most, we rank these homepage candidates according to the predicted probability in descending order and select the top 3.

2.8. Related Entity Reranking

In this phrase, we have already extracted related entities and their homepages. We take a simple approach to rerank the entity list by filtering whose homepages' probability scores are all below 0.5 which represent that the entity has no homepage at all. Naturally, by definition, every entity will have a homepage at least. Then the entities which are considered as no homepages will be dropped off. Of course, this decision depends on the precision of the homepage ME classifier and the coverage of the used corpus.

2.9. Experimental Setup

We ran our index builds and our queries on an IBM 366 server and data are located on SCSI disk made in RAID5. For conveniently handling we divide the index in six sections, which occupy 649G disk space totally. Index size is the total size of the index on disk including both the inverted file and compressed collection. All indexed documents are 50,220,423. For the slow index speed, we did not index anchor text but just title and heading fields. Documents are stemmed with the Krovetz stemmer and stopped using a standard list of 421 common terms. The metadata indexed include docno and url.

We used the full collection and simply handled all documents as HTML documents. That is, we did not resort to any special treatment of document types, nor did we exploit the internal document structure that may be present; instead, we represented all documents as plain text. We did not take special consideration on those wikipedia documents in the corpus.

Supporting documents When we extract context from a document we also store its document id and the entity the context belongs too. We return up to 3 supporting documents from this set for each entity.

2.10. Results and Discussions

The official test set contained only 20 queries. Given the facts that this is a new task, a new collection, and it supplies a relatively small number of topics, evaluation will primarily focus on analysis of the results and runs on a per-topic basis, rather than on average measures. The normalized discounted cumulative gain (nDCG) is an important measure in the official results. There are many methods to evaluate the system. For example, we can evaluate the system by the relevance of the normal homepages, Wikipedia homepages or names of entities. Table 1 lists the evaluation results of our two runs(we only submit BITDLDE09Run)that are evaluated with different output fields. NAME denotes the names of result entities, HP denotes the normal homepages of result entities and WP denotes the Wikipedia pages of result entities. The official results are the values in the first row. They evaluated the runs

according to the relevance of the primary homepages of entity and did not take the Wikipedia homepages or entity names into consideration. In other rows, we combine different output fields to test the change of performance. The results shows that the performance of each run improves when consider Wikipedia homepages in evaluation. The reason is that the Wikipedia pages are of high quality and make it easy for system to find the homepage. On the other hand, using entity names to evaluate decreases the performance. It shows that directly finding the entity names is more difficult than returning the whole web pages.

Table 2.3: nDCG and P@10 results of our runs with different output fields in Entity track

RunID	BITDLDE09Run		LogisticRegressionRun	
	nDCG	P@10	nDCG	P@10
HP	0.0416	0.0200	0.0499	0.0400
HP+NAME	0.0379	0.0200	0.0471	0.0400
HP+WIKI	0.0705	0.1250	0.0895	0.1150
HP+NAME+WIKI	0.0731	0.1250	0.0879	0.1150

We make a statistic for retrieval performance without discard any homepage for identified entities. It shows that the retrieval ratio is low (0.3024 for all relevant documents and 0.0898 primary homepages). Obviously, it needs to improve query formations to raise recall greatly. The low recall ratio makes a great effect for the low performance for the next step for homepage finding.

In future work, there are a number of things for us to explore. First, we will explore more efficient way to automatically construct queries to improve great recall. It may base the assumption described in Clarke: query terms are likely to appear in close proximity to each other within relevant documents. This technique is expressed by Donald Metzler etc. in TREC 2004 in the terabyte track which is used to evaluate Indri search engine. [2]Second is on homepage finding for a specified named entity. [3] found a prior based on the type of the URL to be a very effective source of information. Our Max Entropy Classifier will take the different distribution for different type entities into consideration. Third, we will employ language model to construct entity model from the selected homepage candidates to improve recall.

Acknowledgments. This work is supported by the grant from Chinese National Natural Science Foundation (No: 60705022).

3. References

[1] http://ilps.science.uva.nl/trec-entity/guidelines/

[2] Donald Metzler, T. S., Howard Turtle, W. Bruce Croft (2004). "Indri at TREC 2004: Terabyte Track."

[3] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In Proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02), 2002.