Differences in Document Retrieval and Entity Retrieval: Finding Support Documents with a Learning to Rank 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". In the entity retrieval, people usually work in this way: find a small set of support documents which contain answer entities, and then further detect the answer entities in this set. In most cases, people treat the support document findings as the conventional document retrieval problem. That is, support documents are relevant documents. In this work, we indicate support documents and relevant documents, although similar, have important differences. Further, we propose a learning to rank approach to find support documents. The results show that the learning to rank method runs significantly better than the baseline systems which 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 INTRODUCTION
Traditional search engines return a sequentially ranked list of documents as results according to a user's information needs. However, in some cases, people would like to know the exact entity answer for a query, like "what is the product of Medimmune Inc.", instead of a document containing the answers. This scenario enforces the study of entity retrieval. The difference of retrieval unit between conventional document retrieval and entity retrieval not only causes the variations of the results, but also causes divergences on assumptions and relevant judgments. Although search engines analyze hyperlinks and anchor texts, they are still based on the assumption of the "bag of words" model in the document units. Moreover, the relevance judgments are also on the document level. If any piece of the document is relevant (regardless of how small that piece is in relation to the rest of the document), retrieval systems will mark it as some sort of relevancy. This kind of search engine eschews analyses on answer entities with a user's information needs, since the identification of entities has not occurred yet. Entity retrieval, on the other hand, assumes the answer entities have some kinds of relationships with the topic entities, and is evaluated with a different unit, which will be a useful alternative for document retrieval on a large and diversity Web environments.

SIGIR Workshop on the Entity-Oriented Search, July 23, 2011, Beijing, P.R. China.

Entity retrieval systems, as conventional information retrieval tasks, require the effective and efficient return of the entity answers from a large unstructured corpus (e.g., the Web) or a semi-structured corpus (e.g., Wikipedia). In order to effectively and efficiently search entities, word-independent factors and word-dependent factors should be separated into two stages. In the word-independent stage, the assumption of the "bag-of-words" is applied to efficiently find the support documents on the assumption of word co-occurrence. In the word-independent stage, further complicated analyses are applied to the small set of support documents to effectively detect answer entities.

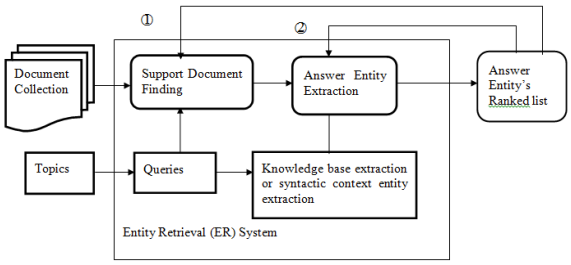

Figure 1. A Two-Layer Retrieval and Extraction Probability Model (TREPM).

With this consideration, we propose a Two-layer Retrieval and Extraction Probability Model (TREPM) to decouple entity retrieval tasks into two layers: support document finding and answer entity extraction, as seen in Figure 1 [1]. The inputs of the system include documents—HTML pages or plain texts—and users' information needs—the search task description with required entity type. The output answers are ranked lists of entities. The first layer, support document finding, is to retrieve a very small subset of the support document collection which contains the answer entities for further extractions; and the second layer, answer entity extraction, extracts the answer entities from the documents. The support document finding 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. It is easy to understand that it should contain as many answer entities as possible. Moreover, its size should be as small as possible since the answer entity extraction is to detect the answer entities with complicated analysis which will be a time-consuming task. The smaller the support document set, the more efficient the retrieval process is.

With the probability model, we describe the entity retrieval problem in the TREPM model as $p(e|q,t)$, 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

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

However, it is not efficient to calculate all document similarities, and detect the answer entities from all documents. Therefore, we choose support documents $d_{support}$ to estimate this probability.

$$p(e|q,t) \approx \sum_{d_{support}} p(d_{support}|q,t)\,p(e|d_{support},q,t)$$

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

Support documents are slightly different from the conventional relevant documents because support documents need to meet two criteria: being as small as possible, and containing as many answer entities as possible. I use "support" documents instead of "relevant" documents to distinguish them. 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 we expect the documents which directly answer the query ranking higher than the pages with miscellaneous information. In support document finding task, however, the expected rank list is reversed because a small set of support documents are preferred for further detection tasks, instead of exploring a huge number of documents.

In this study, we propose a learning to rank method for the support document finding. That is, with the 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 learning to rank 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 needs (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 learning to rank method for support document findings.

3 FINDING SUPPORT DOCUMENTS WITH THE LEARNING TO RANK APPROACH

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 probability of a document to be the support document according to the training data.

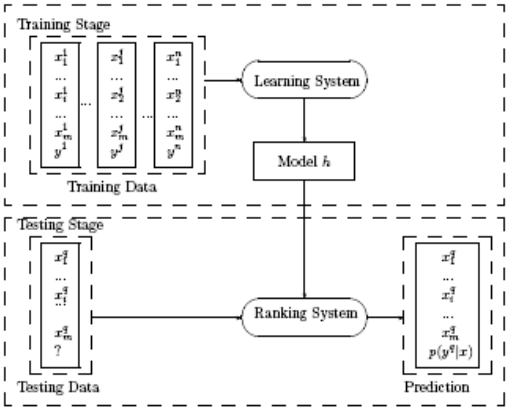

Figure 2. the Learning to Rank Framework

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.

3.1 Learning to rank 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 or. 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 degree 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 learning-to-rank method

Applying the learning to rank 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 learning to rank 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 of 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:

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

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 learning to rank 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 learning to rank 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 learning to rank 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.

4.1 Results

Figure 2 shows the results of the baseline systems and the learning to rank method for the support document findings. Precision, recall, and f-measure at rank 1 to 16 are reported. The logistic regression method is applied to learning based method, and the results are 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 learning to rank 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.

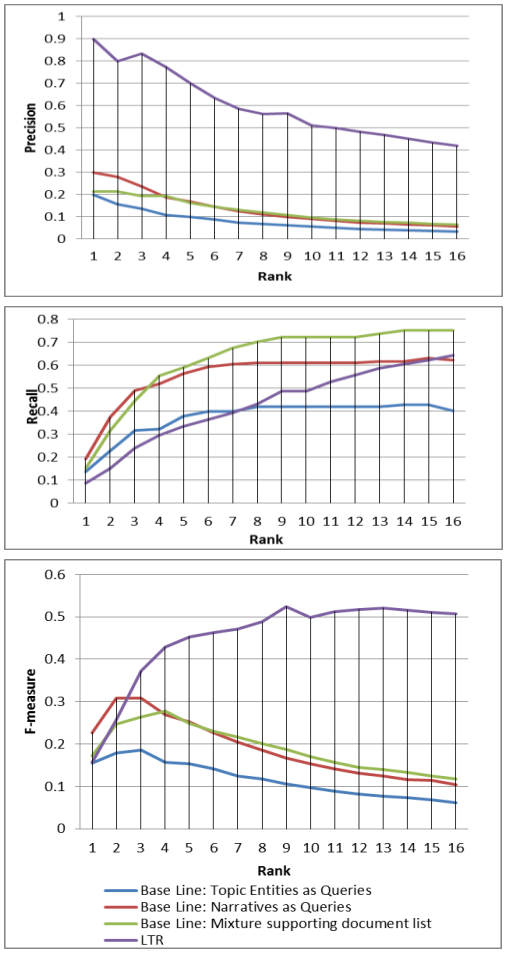

Figure 2. Precision, Recall, and F-measure for the Baseline Systems and the Learning to Rank method

With Figure 2 of the error rates on four topic types, we found that for the learning to rank 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 2: 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 learning to rank 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. The learning to rank method is applied for this task, and the various features are discussed. 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 learning to rank method is significantly better than the three baseline systems which treat the support document finding as a conventional document retrieval problem. Although the learning to rank 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.

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.