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.

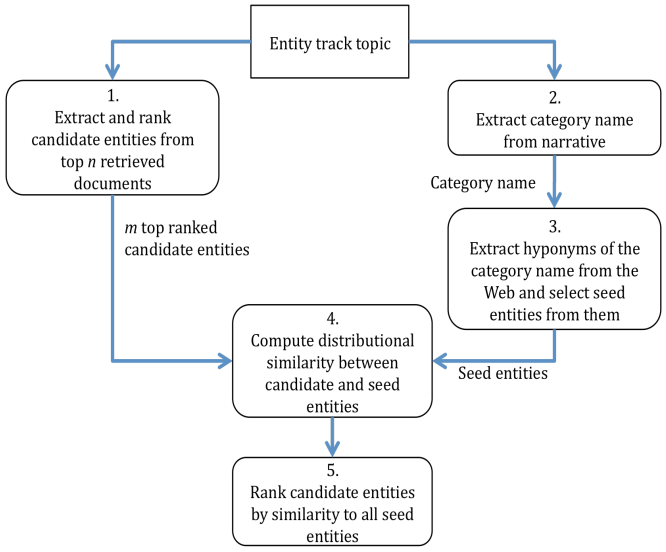
Entity track topic
1. Extract and rank candidate entities from top *n* retrieved documents
2. Extract category name from narrative
Category name
m top ranked candidate entities
3. Extract hyponyms of the category name from the Web and select seed entities from them
4. Compute distributional similarity between candidate and seed entities
Seed entities
5. Rank candidate entities by similarity to all seed entities

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.

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:

$$\begin{aligned} &TFIDFEntitySeedBM25 = \\ &= \alpha \times \log(TFIDF) + (1-\alpha) \times \log(EntitySeedBM25) \qquad (3) \end{aligned}$$

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.

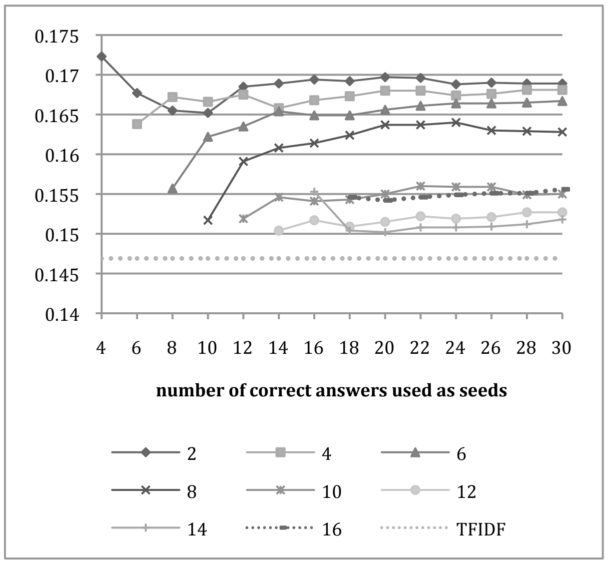

Figure 3. Effect of the number of correct answers as seeds and seed co-occurrence thresholds on nDCG@R (QA 2005).

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.

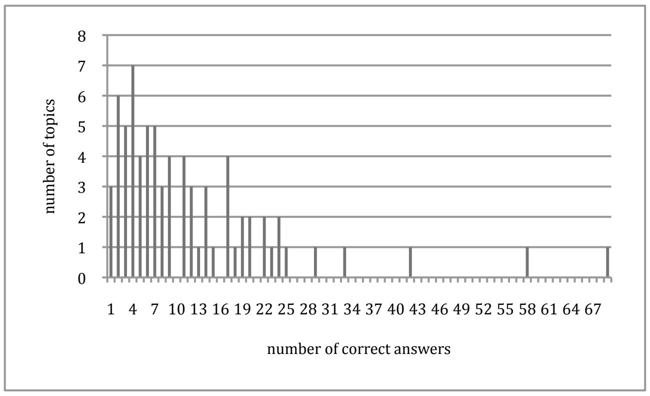

Figure 4. Distribution of correct answers for the QA 2005 list questions.

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.

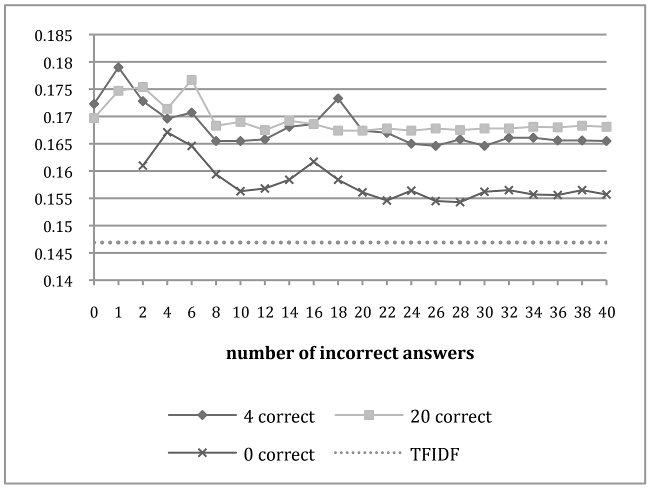

Figure 5. Effect of the number of incorrect answers on nDCG@R (QA 2005).

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.