TREC-8 Experiments at SUNY at Buffalo

B. Han R. Nagarajan R. Srihari M. Srikanth

Department of Computer Science and Engineering
State University of New York at Buffalo,
Amherst, NY 14228-2567

1 Introduction

For TREC-8, State University of New York at Buffalo(UB) participated in the ad-hoc task and the spoken document retrieval(SDR) track. This is our first year of participation at TREC. We submitted two runs for the Ad-hoc task. The first run was term vector-based using SMART[10]. The second run used the TROVE - Text Retrieval using Object VEctors - system. For the SDR Track, we participated in the IR component of the Quasi-SDR task.

2 Ad-hoc Task

2.1 Overview

The UB team submitted two runs for the Ad-hoc Task. In the first run (UB99SW) we used SMART for indexing and retrieval on expanded queries generated by WordNet[4]. For each topic SMART+WordNet generated 5000 top ranked documents, which were input to our second retrieval engine TROVE (Text Retrieval using Object VEctor), and the result was submitted (UB99T). TROVE is our first attempt at exploring the feasibility of employing natural language processing (NLP) techniques in IR tasks. The system is implemented in its entirety from basic principles.

For decades NLP has been a promise to improving IR preformance, yet different experiments have had varying degrees of success [11]. In our experiment we focus on extracting semantics of documents using NLP techniques. In particular we avoid the ambiguities in full syntactical parsing and only extract semantics of partial phrases. The similarity between two phrases is then determined by the concepts they convey, instead of by their mere surface forms. The approach is similar to [12], however we analyze and represent the relations between phrase constituents by using semantics rules based on the types of constituents.

In the proposed TROVE system syntactical groups (noun groups, verb groups, etc) are first identified as *objects*, then short phrases representing semantics are grouped on top of these objects. The semantics are extracted based on the type codes assigned to the objects, and represented as *relation vectors* between two objects. The matching proceeds as a two-level process. In the first level match (*node-level* match) we establish a one-one mapping between objects in documents and objects in queries by comparing their similarities based on their semantic distance in WordNet. The second level match (*arc-level* match) proceeds by comparing the relation vectors between the corresponding objects. A similarity score is finally computed based on a conditional probability formula relating the two levels. The overall system diagram is given in Figure 1, where SEM denotes the semantics files, OBJ denotes the object list files, and VEC represents the vector files.

Due to time constraints, we were unable to finish the implementation of the complete TROVE system before the TREC deadline date. In particular the Semantics Interpreter and the Named Entity Tagger module were not ready. UB99T was generated using a partial implementation, i.e., only the node level match has been achieved, and the run was not completed.

For UB99T, we were able to use only node level matching for retrieval. Also, due to computational requirements, we were not able to complete the procedures for all 50 queries of the ad-hoc task. However,

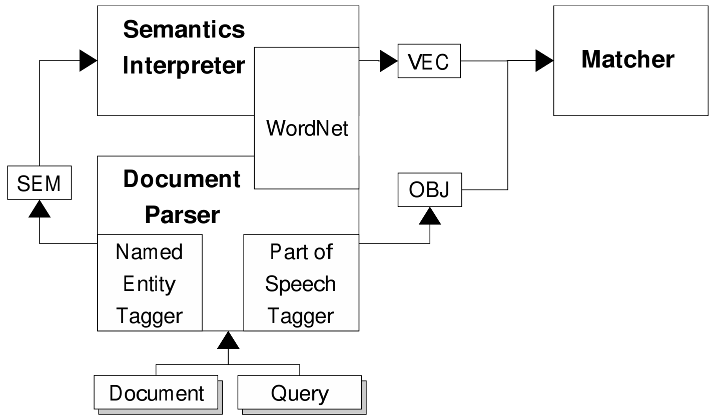

Figure 1: The TROVE system

in order to have our result evaluated, we decided to augment the incomplete portions of UB99T with the results from UB99SW. The two runs were submitted for comparison purposes.

2.2 Query Expansion via WordNet

In UB99SW we used WordNet for simple query expansion. For each topic we used the title and the description and filtered the stop-words out. Then for each noun and verb[1] we traversed upward and downward via hyponyms/hypernyms semantic links. The depth of traversal was arbitrarily set to 7 for ancestors and 6 for descendants. For all expansion words we encountered, only the first 5 of them were selected. A sample query expansion for the words **eliminate** and **border** is shown below:

eliminate destroy kill discharge expel eject get_rid_of do_away_with obviate rid_of annihilate

border boundary bound bounds boundary edge boundary_line borderline

2.3 Object Identification

To identify objects within a document, we first used a rule-based Part of Speech Tagger [2] to tag each term in the tokenized document. The identification is done using regular expressions involving the POS tags, and consists of 6 possible patterns:

1. Don't care (DC) patterns: These include existential 'there', list item marker, modal words (e.g. '*can*'), pronouns (including wh-pronouns) and wh-adverbs. DCs only serve as placeholders in order to correctly extract phrases at the later stages.

2. Auxiliary/stative verb group (A/SVG) patterns: These include various *progressive form* and *perfect tense* verb groups. For examples, "*is badly injured*", "*is charged with*", "*has amassed*", and "*have been considered seriously*". Whenever possible, we combine the ending preposition word with the head verb, e.g., "*is taken off*" is grouped as "*is taken_off*". This is done by looking up potential groupings in WordNet.

3. Simple verb group (VG) patterns: These cover the usual verb groups like "*accurately target*" and "*look at*". Similar to A/SVG the potential grouping with the ending preposition word is checked.

4. Noun group (NG) patterns: These cover complete noun groups like "*the three leftmost blue boxes*" and "*finishing touches*". Complex noun groups are identified by enumerating all possible combinations and looking them up in WordNet. For example, "*New York*" will be grouped as "*New_York*" and "*hot dog*" as "*hot_dog*".

[1] In SDR track a similar approach is used for query expansion, however we included adjective expansion as well in SDR track.

5. Function word (FW) patterns: These include all preposition words, conjunctions ('*or*' and '*and*') and the word '*to*'. FW is crucial at the stage of phrase extraction and semantics interpretation.

6. Punctuation mark (PM) patterns: These cover all punctuation marks, including possessive endings ('s) and symbols (e.g., parentheses). PMs only serve as placeholders in order to correctly extract phrases at the later stages.

Each object consists of a head and a list of modifiers. A type code is assigned to each word in an object according to 25 unique noun beginners [6] and 15 unique verb beginners [3], the type of an object is determined by the type code of its head.

In designing the patterns we try to avoid conflicts between different pattern matches. In case of conflicts the pattern with a smaller pattern number is favored. For example, "*is holding hands*" is identified as an A/SVG instead of a NG.

2.4 Phrase Extraction

Based on regular expressions involving the list of objects identified for each document, short phrases are extracted. There are 5 phrase patterns:

1. Complex nominals (CN): These group two NGs into one complex nominal, e.g. "*Chicago hot_dog*".

2. Possessive forms (PF): These cover the short phrases such as "*individual's personality*".

3. Complete phrases (CP): These include a complete short phrases like "*a man run across the street*".

4. NGs with situations (NS): These cover incomplete phrases such as "*man from Mars*". NGs are meant to capture parts of CPs when CP extraction is not possible.

5. Conjunctive groups (CG): These include noun and verb conjunctive groups. Examples: "*he moves into and holds the 2nd place in the competition*" and "*Mary and Bill's wedding*".

The patterns are applied in the order shown above, in particular CN and PF patterns are rewriting patterns in order to extract phrases like "*comments from stock exchange's listings division*". Each phrase is represented by 5 constituents: *subject*, *action*, *object*, *subject situations* and *situations*, of which each of the first three is an object, and the last two are two lists of FWs and objects. Thus "*the deal with BMW does not adversely affect Honda's policies in Europe*" is represented as:

- Subject: *the deal*
- Subject situations: *with BMW*
- Action: *does not adversely affect*
- Object: *Honda's policies*
- Situations: *in Europe*

The type of a particular constituent is determined by the type of the object involved. Therefore the semantics of a phrase can be inferred from the type information of its constituents.

2.5 Node-level Similarity

To obtain the node-level similarity score of a document, we compute the semantic distances of noun and verb objects between documents and in queries using WordNet. Since an object consists of a head and a list of modifiers, the similarity between two objects is taken as a linear combination of the similarity of the heads and that of the modifiers. Thus the problem is reduced to computing the similarity between two words.

The similarity of two words is determined by a slightly modified formula from [1]:

$$Sim(w_1, w_2) = \frac{idf_{w_1} \times idf_{w_2}}{Dist(w_1, w_2) + 1}$$

where $Dist(w_1, w_2)$ is the distance between word w_1 and w_2, and idf_{w_i} is the *inverse document frequency* of word w_i in the data collection. The distance between two words is determined by a weighted edge count in WordNet following hypernym/hyponym links. The weighting scheme adopted reflects the likelihood of a particular sense of a word, together with the specificity of the words along the semantic paths based on the notion of *basic-level lexicalized concepts* [9].

For words of different syntactic categories (i.e., noun vs. verb), a conversion to noun is attempted for the verb by trying to find if it has a noun entry in WordNet. The similarity is then computed between the two nouns, but it is panelized by a predefined factor.

The final node-level similarity score for a document is thus defined as

$$Sim = Sim_R \times C_Q \times c \times C_D$$

where Sim_R is the accumulated words similarity (*raw* similarity), C_Q and C_D are the percentage of terms being matched in the query and the document (*coverage*), respectively, and c is simply a constant for weighting C_D.

2.6 Discussions

Since UB99T run was incomplete and it was augmented with results from UB99SW, it is rather difficult for us to draw any significant conclusion at the moment. Moreover, the augmentation implies a performance upper bound set by UB99SW. However, the preliminary results from TREC-8 evaluation shows that out of 28 topics processed by TROVE, 12 have been improved over UB99SW in terms of average precision, on average however, the performance degenerated. A closer analysis reveals that the lack of query term weighting misled the system to target the wrong content words in the data collection. This will be addressed in future work.

3 SDR Task

3.1 System description

An in-house version of term vector based retrieval system was used for TREC SDR experiments. A combination of query term expansion and blind relevance feedback[10, 5] was used to obtain our submissions – **cedar-r1** and **cedar-b1**.

The following similarity measure was used for matching queries and documents.

$$sim(q, d) = \sum_{t \in q \cup d} w(d, t) * w(q, t) \tag{1}$$

with

$$\begin{aligned} w(d, t) &= \frac{1}{Z(d)} f(d, t) \cdot \log(\frac{N}{f(t)}), \quad f(t) \neq 0 \\ w(q, t) &= \frac{1}{Z(q)} f(q, t) \end{aligned}$$

where $f(d, t)$ and $f(q, t)$ are the frequencies of the term t in document d and query q, respectively. The value of N gives the total number of documents in the collection and $f(t)$ gives the number of documents in which the term t occurs. $Z(d)$ and $Z(q)$ are normalizers for $w(d, t)$ and $w(q, t)$, respectively to ensure that the weights are between 0 and 1.

The runs **cedar-r1** and **cedar-b1** that correspond to the R1 and B1 retrieval conditions of SDR were obtained using the following approach. After some preprocessing of the documents, index terms are extracted by filtering out stop words and reducing to word stems using the Porter stemming algorithm[7]. Around 540 stop words were used. The document retrieval is done in two phases. In the first phase, the query terms, which are stopped and stemmed, are used to rank the documents using the similarity measure in equation (1). In the second phase, the query is expanded using WordNet based on the query expansion scheme identified in section 2. The query expansion phase generated up to 5 terms for each query term in the original query. The original and expanded query terms are filtered and reweighted using the blind relevance feedback technique. The top 10 documents from the document ranking of the first phase are assumed to be relevant and the query term weight are adjusted accordingly. The reweighted set of query terms is used to rerank the documents in the collection. A final set of 1000 documents is retrieved using this document ranking.

3.2 SDR runs and Analysis

Table 1 gives the results for TREC-8 SDR submissions. It gives the precision values at different recall points as well as the average precision.

runid	5 docs	10 docs	20 docs	200 docs	Avg. Prec.
cedar-r1	0.4816	0.4551	0.3612	0.1247	0.3906
cedar-b1	0.4245	0.4000	0.3286	0.1127	0.3430

Table 1: TREC-8 SDR results

Table 2 gives a comparison of the performance of our system with respect to other participants. The table gives the number of queries that achieved the highest average precision, at least a median average precision or the lowest average precision.

runid	=Best	≥Median
cedar-r1	2	13
cedar-b1	0	7

Table 2: Average Precision comparisons with other TREC-8 participants

Some of the reasons for this average performance of the system are:

- The query expansion used WordNet and up to 5 additional terms were added to the initial query for each query term. This strategy for query expansion seems insufficient from the performance. More terms would have improved the performance of the system.
- We used blind relevance feedback to reweight the WordNet expanded query terms. Reweighting the query terms did change the ranking of the results. An alternate strategy is to use some of the terms from the top 10 relevant documents retrieved based on the initial query as additional query terms.

We plan to experiment in these directions as well as use the TROVE system for SDR task.

4 Conclusions

This was our first participation in TREC. Our performance is just about the average performance of systems. Our participation is TREC was a learning experience. Based on the results, query expansion is one of the main areas we plan to concentrate on to improve our results.

With respect to TROVE, only a part of the system is currently implemented. The following is in the works for the future:

1. Semantics Interpretation based on short phrases with type information: Ideally this should provide an elegant rule syntax so that one can specify semantics rules for converting a short phrase into an appropriate 3-valued predicate, e.g.,

   [location]/1 [location]/2 $\Rightarrow$ IN(%2, %1) // example: *new_york suburb*
   [time]/1 [event]/2 $\Rightarrow$ IN(%2, %1) // example: *yesterday's accident*
   [*]/1 [*]/2 [*]/3 $\Rightarrow$ AGENT(%2, %1), PATIENT(%2, %3)

2. More sophisticated query processing: This includes a more detailed syntactical analysis for queries in order to filter out the unrelated words. Also the query term weighting/expansion can be done based on *blind relevance feedback* so a set of most similar terms will be returned as expansion candidates and the weight of a term which receives the highest hit score earns the highest weight.

3. Sense disambiguation via *class-based probability*: In current implementation sense disambiguation is done by weighting semantic paths in WordNet for a word such that the more frequently used senses are preferred. We plan to incorporate class-based probabilities [8] to compute the confidence score for each sense of a noun based on the pivotal verb. For example, the word '*snow*' in "*snow is falling fast*" will have a much more preferred sense "*precipitation falling from clouds in the form of ice crystals*" instead of the sense for "*cocaine*", as hinted by the pivotal verb '*falling*'.

References

[1] Y. A. Aslandogan, C. Thier, C. T. Yu, J. Zou, and N. Rishe. Using semantic contents and wordnet(tm) in image retrieval. In *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, 1997.

[2] E. Brill. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing, ACL.*, 1992.

[3] C. Fellbaum. A semantic network of english verbs. In C. Fellbaum, editor, *WordNet: an Electronic Lexical Database*, chapter 3. MIT Press, 1998.

[4] C. Fellbaum, editor. *WordNet: an Electronic Lexical Database*. MIT Press, 1998.

[5] W. B. Frakes and R. Baeza-Yates, editors. *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall Inc., Englewood Cliffs, NJ, USA, 1992.

[6] G. A. Miller. Nouns in wordnet. In C. Fellbaum, editor, *WordNet: an Electronic Lexical Database*, chapter 1. MIT Press, 1998.

[7] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.

[8] P. Resnik. Wordnet and class-based probabilities. In C. Fellbaum, editor, *WordNet: an Electronic Lexical Database*, chapter 10. MIT Press, 1998.

[9] E. Rosch, C. B. Mervis, W. Gray, D. Johnson, and P. Boyes-Braem. Basic objects in natural categories. *Cognitive Psychology*, 8:382 – 349, 1976.

[10] G. Salton, editor. *The Smart retrieval system: experiments in automatic document processing*. Prentice-Hall, 1971.

[11] A. F. Smeaton. Using nlp or nlp resources for information retrieval tasks. In T. Strzalkowski, editor, *Natural Language Information Retrieval*. Kluwer Academic Publishers, 1999.

[12] T. Strzalkowski and et al. Natural language information retrieval: Trec-7 report. In D. K. Harman, editor, *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*, 1998.