Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004), Geneva, Switzerland, pp. 1360-1366, 2004.

Learning to Identify Single-Snippet Answers to Definition Questions

Spyridoula MILIARAKI and **Ion ANDROUTSOPOULOS**
Department of Informatics, Athens University of Economics and Business
Patission 76, GR-104 34, Athens, Greece

Abstract

We present a learning-based method to identify single-snippet answers to definition questions in question answering systems for document collections. Our method combines and extends two previous techniques that were based mostly on manually crafted lexical patterns and WordNet hypernyms. We train a Support Vector Machine (SVM) on vectors comprising the verdicts or attributes of the previous techniques, and additional phrasal attributes that we acquire automatically. The SVM is then used to identify and rank single 250-character snippets that contain answers to definition questions. Experimental results indicate that our method clearly outperforms the techniques it builds upon.

1 Introduction

Since the introduction of the TREC QA track (Voorhees, 2001), question answering systems for document collections have attracted a lot of attention. The goal is to return from the collection text snippets (eg., 50 or 250 characters long) or exact answers (e.g., names, dates) that answer natural language questions submitted by users.

A typical system first classifies the question into one of several categories (questions asking for locations, persons, etc.), producing expectations of types of named entities that must be present in the answer (location names, person names, etc.). Using the question terms as a query, an information retrieval (IR) system identifies possibly relevant passages in the collection, often after query expansion (e.g., adding synonyms). Snippets of these passages are then selected and ranked, based on criteria such as whether or not they contain the expected types of named entities, the percentage of question words in each snippet, the percentage of words that also occur in other candidate snippets, etc. The system reports the most highly-ranked snippets, or, in the case of exact answers, named entities of the required type therein.

Unfortunately, the approach highlighted above falls short with questions that do not generate expectations of particular types of named entities and contain very few non-stop-words. Definition questions (e.g. "What is a nanometer?", "Who was Duke Ellington?") have both properties, and are particularly common. In TREC-2001, where the distribution of question types reflected that of real user logs, 27% of the questions were requests for definitions. Hence, techniques to handle this category of questions are very important.

We propose a new method to answer definition questions, that combines and extends the technique of Prager *et al.* (2001, 2002), which relied on WordNet hypernyms, and that of Joho *et al.* (2001, 2002), which relied on manually crafted lexical patterns, sentence position, and word co-occurrence across candidate answers. We train an SVM (Schölkopf and Smola, 2002) on vectors whose attributes include the verdict of Prager *et al.*'s method, the attributes of Joho *et al.*, and additional phrasal attributes that we acquire automatically. The SVM is then used to identify and rank 250-character snippets, each intended to contain a stand-alone definition of a given term, much as in TREC QA tasks prior to 2003.

In TREC-2003, the answers to definition questions had to be lists of complementary snippets (Voorhees, 2003), as opposed to single-snippet definitions. Here, we focus on the pre-2003 task, for which TREC data were publicly available during our work. We believe that this task is still interesting and of practical use. For example, a list of single-snippet definitions accompanied by their source URLs can be a good starting point for users of search engines wishing to find definitions. Single-snippet definitions can also be useful in information extraction, where the templates to be filled in often require short entity descriptions; see Radev and McKeown (1997). Experiments indicate that our method clearly outperforms the techniques it builds upon in the task we considered. We sketch in section 6 how we plan to adapt our method to the post-2003 TREC task.

2 Previous techniques

Prager *et al.* (2001, 2002) observe that definition questions can often be answered by hypernyms; for example, "schizophrenia" is a "mental illness", where the latter is a hypernym of the former in WordNet. Deciding which hypernym to report, however, is not trivial. To use an example of Prager *et al.*, in "What is a meerkat?" WordNet provides hypernym synsets such as {"viverrine",

"viverrine mammal"} (level 1), {"carnivore"} (level 2), {"placental", …} (level 3), {"mammal"} (level 4), up to {"entity", "something"} (level 9). In a neutral context, the most natural response is arguably "mammal" or "animal". A hypernym on its own may also not be a satisfactory answer. Responding that an amphibian is an animal is less satisfactory than saying it is an animal that lives both on land and in water.

Prager *et al.* identify the best hypernyms by counting how often they co-occur with the definition term in two-sentence passages of the document collection. They then short-list the passages that contain both the term and any of the best hypernyms, and rank them using measures similar to those of Radev *et al.* (2000). More precisely, given a term to define, they compute the level-adapted count (LAC) of each of its hypernyms, defined as the number of two-sentence passages where the hypernym co-occurs with the term divided by the distance between the term and the hypernym in WordNet's hierarchy. They then retain the hypernym with the largest LAC, and all the hypernyms whose LAC is within a 20% margin from the largest one. To avoid very general hypernyms (e.g., {"entity", "something"}), Prager *et al.* discard the hypernyms of the highest 1 or 2 levels in WordNet's trees, if the distance from the top of the tree to the definition term is up to 3 or 5, respectively; if the distance is longer, they discard the top three levels. This ceiling is raised gradually if no co-occurring hypernym is found.

Prager *et al.*'s method performed well with the definition questions and documents of TREC-9 (Prager *et al.*, 2001). In 20 out of the 24 definition questions they considered (83.3%), it managed to return at least one correct response in the five most highly ranked two-sentence passages; in all 20 questions, the correct response was actually the highest ranked. In TREC-2001, however, where there were more definition questions mirroring more directly real user queries, this percentage dropped to 46% (Prager *et al.*, 2002). In 44 of the 130 definition questions (33.8%) they considered, WordNet did not help at all, because it either did not contain the term whose definition was sought, or none of its hypernyms were useful even as partial definitions. Even when WordNet contained a hypernym that constituted a self-contained definition, it was not always selected.

In work that led to an alternative method, Hearst (1998) sketched a method to identify patterns that signal particular lexical semantic relations, in effect a bootstrapping approach. Applying this process by hand, Hearst was able to identify four high precision hyponym–hypernym patterns that are common across text genres. The patterns are shown below in the slightly modified form of Joho and Sanderson (2000). Here, *qn* (query noun) and *dp* (descriptive phrase) are phrases containing hyponyms and hypernyms, respectively.

(1) (*dp* such | such *dp*) as *qn*
e.g., "injuries such as broken bones"
(2) *qn* (and | or) other *dp*
e.g., "broken bones and other injuries"
(3) *dp* especially *qn*
e.g., "injuries especially broken bones"
(4) *dp* including *qn*
e.g., "European countries including England, France, and Spain"

Compared to Prager *et al.*'s method, Hearst's patterns have the advantage that they may identify hyponym–hypernym relations that are not present in WordNet, which is often the case with domain-specific terminology and proper names.

Joho and Sanderson (2000) observe that co-occurrences of hyponyms and hypernyms are often indicative of contexts that define the hyponyms. Instead of using WordNet, they identify hyponym–hypernym contexts using Hearst's patterns, to which they add the following ones.[1] Here, *qn* is the term to define, and *dp* is a phrase that contains a definition of *qn*; *dp* no longer necessarily contains a hypernym of *qn*. Each pattern is assigned a weight, which is its precision on a training set (the number of sentences it correctly identifies as containing a definition, over the number of sentences it matches).

(5) *qn* "(" *dp* ")" | "(" *dp* ")" *qn*
e.g., "MP (Member of Parliament)"
(6) *qn* (is | was | are | were) (a | an | the) *dp*
e.g., "Tony Blair is a politician"
(7) *qn* , (a | an | the) *dp*
e.g., "Tony Blair, the politician"
(8) *qn* , which (is | was | are | were) *dp*
e.g., "bronchitis, which is a disease of…"
(9) *qn* , *dp* , (is | was | are | were)
e.g., "Blair, Prime Minister of Britain, is…"

Joho and Sanderson first locate all the sentences of the document collection that contain the term to define, and then rank them using three attributes. The first one (*KPW*) is the weight of the pattern the sentence matched, if any. The second attribute (*SN*) is the ordinal number of the sentence in its document, ignoring sentences that do not contain the term; sentences that mention first the term in a document are more likely to define it. The third attribute (*WC*) shows how many of the words that

[1] We ignore a variant of (7) that ends in ".", "?" or "!".

are common across candidate answers are present in the sentence. More precisely, to compute *WC*, Joho and Sanderson retrieve the first sentence of each document that contains the definition term, and retain the 20 most frequent words of these sentences after applying a stop-list and a lemmatizer. *WC* is the percentage of these words that are present in the sentence being ranked. The sentences are ranked using the weighted sum of the three attributes, after hand-tuning the weights of the sum on a training set. In Joho *et al.* (2001), this method was evaluated with 50 definition questions and the top 600 documents that Google returned for each definition term. It returned a correct definition in the five top-ranked sentences in 66% of the questions.[2] As with Prager *et al.*'s method, then, there is scope for improvement.

3 Our method

Prager *et al.*'s approach is capable of answering a large number of definition questions, though, as discussed above, it does not always manage to locate the most appropriate hypernym, and the appropriate hyponym-hypernym relation may not be present in WordNet. Joho *et al.*'s technique does not rely on a predetermined ontology, which makes it less vulnerable to domain-specific terminology and proper names. Its limited pattern set, however, cannot capture definitions that are phrased in less typical ways, and the fact that the weights of the three attributes are hand-tuned raises doubts as to whether they are optimal. Our method overcomes these limitations by combining the two approaches in a common machine learning framework, and by using a larger attribute set.

We assume that a question processing module that separates definition from other types of questions is available, and that in each definition question it also identifies the term to be defined. When such a module is not available, the user can be asked to specify explicitly the question type and the term to be defined, via a form-based interface as in Buchholtz and Daelemans (2001). Hence, the input to our method is a (possibly multi-word) term, along with the most highly ranked documents that an IR engine returned for that term (in the experiments of section 4, we used the top 50 documents). The goal is to identify five 250-character snippets in these documents, such that at least one of the snippets contains an acceptable definition of the input term. As an example, we show below the two snippets that configuration 4 of our method (to be discussed) considered most appropriate in the case of "What are pathogens?". The first snippet defines pathogens as hazardous microorganisms. The second one provides instances of pathogens, but does not actually state what a pathogen is.

…considerations as nutrient availability. In particular, the panel concluded that the fear of creating hazardous microorganisms, or pathogens, is overstated. "It is highly unlikely that moving one or a few genes from a pathogen to…

…definite intraspecial physiological and morphological diversity. Ph. helianthi thrives at higher temperatures than other sunflower pathogens (Sclerotinia sclerotiorum and Botrytis cinerea) do. In various nutrient media, Ph. helianthi …

We select the five snippets to report using alternatively a baseline and four different configurations of our learning-based method.

3.1 Baseline

As a baseline, we use a reimplementation of Prager *et al.*'s method that operates with 250-character snippets.[3] Unlike Prager *et al.*, our reimplementation does not use a ranking function (Radev *et al.*, 2000). When more than five snippets contain both the input term and one of its best hypernyms, we rank the snippets according to the ranking (*RK*) of the documents they derive from, i.e., the ranking of the IR engine. Our evaluation results (section 4) indicate that the performance of our implementation is still very similar to that of Prager *et al.*

3.2 Learning-based method

In all the configurations of our learning-based method, we use a Support Vector Machine (SVM) with a simple inner product (polynomial of first degree) kernel (Schölkopf and Smola, 2002), which in effect learns an optimal linear classifier without moving to a higher-dimension space. (We have experimented with higher degree polynomial kernels, but there was no sign of improvement.) The SVM is trained as follows. Given a training set of terms to be defined and the corresponding documents that the IR engine returned, we collect from the documents all the 250-character snippets

[2] Joho *et al.* report better results when using a less stringent form of evaluation that admits partial answers, but their acceptance criteria in that case appear to be over-permissive.

[3] Following Prager *et al.* (2002), we consider synonyms of the input term as level-0 hypernyms, and include them in the search for best hypernyms; their LAC is the number of times they co-occur with the input term. We did not implement the tests for orthographic variations and count ratios of Prager *et al.* When an input term occurs in multiple synsets, which produces multiple paths towards hypernyms, we select the hypernyms with the best overall LAC scores, instead of the best scores per path, unlike Prager *et al.* (2001).

that have the term at their center. Each snippet is then converted to a training vector, the attributes of which differ across the configurations presented below. The training vectors are manually classified in two categories, depending on whether or not the corresponding snippets contain acceptable definitions. The SVM is trained on these vectors to predict the category of new, unseen vectors from their attributes.

The SVM implementation we used actually returns confidence scores, showing how probable it is that a particular vector belongs to each category.[4] Using a classifier that returns confidence scores instead of binary decisions is crucial, because the training vectors that correspond to definitions are much fewer than the vectors for non-definitions (3004 vs. 15469 in our dataset of section 4). As a result, the induced classifier is biased towards non-definitions, and, hence, most unseen vectors receive higher confidence scores for the category of non-definitions than for the category of definitions. We do not compare the two scores. We pick the five vectors whose confidence score for the category of definitions is highest, and report the corresponding snippets; in effect, we use the SVM as a ranker, rather than a classifier; see also Ravichandran *et al.* (2003). The imbalance between the two categories can be reduced by considering (during both training and classification) only the first three snippets of each document, which discards mostly non-definitions.

3.2.1 Configuration 1: attributes of Joho at al.

In the first configuration of our learning-based approach, the attributes of the vectors are roughly those of Joho *et al.*: two numeric attributes for *SN* and *WC* (section 2), and a binary attribute for each one of patterns (1)–(9) showing if the pattern is satisfied. We have also added binary attributes for the following manually crafted patterns, and a numeric attribute for *RK* (section 3.1).

(10) *dp* like *qn*
e.g., "antibiotics like amoxicillin"
(11) *qn* or *dp*
e.g., "autism or some other type of disorder"
(12) *qn* (can | refer | have) *dp*
e.g., "amphibians can live on land and…"
(13) *dp* (called | known as | defined) *qn*
e.g., "the giant wave known as tsunami"

This configuration can be seen as an approximation of Joho *et al.*'s method, although it is actually an improvement, because of the extra attributes and the fact that it uses an SVM learner instead of a weighted sum with hand-tuned weights to combine the attributes.

3.2.2 Configuration 2: adding WordNet

The second configuration is identical to the first one, except that it adds an extra binary attribute showing if the snippet contains one of its best hypernyms, as returned by the baseline. This is essentially a combination of the approaches of Prager *et al.* and Joho *et al.* Unlike simple voting (Chu-Carroll *et al.*, 2003), the two methods contribute attributes to the instance representations (vectors) of an overall learner (the SVM). This allows the overall learner to assess their reliability and adjust their weights accordingly.

3.2.3 Configuration 3: *n*-gram attributes

The third configuration adds *m* extra binary attributes, each corresponding to an automatically acquired pattern. (In the experiments of section 4, *m* ranges from 100 to 300.) Observing that most of the previously proposed patterns are anchored at the term to define, we collect from the documents that the IR engine returns for the training questions all the *n*-grams ($n \in \{1, 2, 3\}$) of tokens that occur immediately before or after the definition term. The *n*-grams that are encountered at least 10 times are considered candidate patterns. From these, we select the *m* patterns with the highest precision scores, where precision is defined as in section 2, but for snippets instead of sentences.

In our experiments, this pattern acquisition process re-discovered several of the patterns (1)–(13) or sub-expressions of them, namely [*dp* such as *qn*], [*qn* and other *dp*], [*qn* "(" *dp*], [*dp* ")" *qn*], [*qn* is (a | an | the) *dp*], [*qn* (are | were) *dp*], [*qn* , (a | an | the) *dp*), [*qn* , which is *dp*], [*qn* , *dp*], [*qn* or *dp*], [*qn* can *dp*], [*dp* called *qn*], [*dp* known as *qn*]. It also discovered some reasonable variations of patterns (1)–(13); e.g, [*qn* is one *dp*], [*dp* , (a | an) *qn*] (as in "A sudden health problem, a heart attack or …"), [*dp* , *qn*], [*qn* , or *dp*]. We include *dp*, the phrase that defines *qn*, in the acquired patterns to make them easier to compare to patterns (1)–(13). The acquired patterns, however, do not predict the position of *dp* in a snippet.

Many of the acquired patterns at first look odd, but under a closer examination turn out to be reasonable. For example, definition sentences often start with the term they define, as in "Amphibians are…", "An alligator is…", sometimes with the term quoted, which is how patterns like [. *qn*], [. An *qn*], [. ' ' *qn*] arise. Many of the questions in our data set were about diseases, and, hence, several of the acquired patterns were expressions (e.g,, "people with", "symptoms of") that co-occur frequently with definitions of diseases in snippets.

[4] We used Weka's SVM implementation. Weka is available from http://www.cs.waikato.ac.nz/ml/weka/ .

This suggests that automatic pattern acquisition would allow domain-specific question answering systems (e.g., systems for medical documents) to exploit domain-specific indicators of definitions.

The pattern acquisition process also produced many patterns (e.g., "the hallucinogenic drug", "President Rafael") that do not seem to provide any useful information in general, although they occurred frequently in definition snippets of our training data. Since we did not filter manually the automatically acquired patterns, patterns of the latter kind gave rise to irrelevant attributes that carried noise. This is a common problem in machine learning, which most learning algorithms are designed to cope with. Our experimental results indicate that the SVM learner benefits from the *n*-gram attributes, despite the noise they introduce.

We also experimented with keeping both high and low precision patterns, hoping to acquire both *n*-grams that are indicative of definitions and *n*-grams that are indicative of contexts that do not constitute definitions. The experimental results, however, were inferior, and manual inspection of the low precision *n*-grams indicated that they carried mostly noise, suggesting that it is difficult to identify frequent *n*-grams whose presence rules out definitions reliably.

The reader may wonder why we rely only on precision, rather than selecting also attributes with high recall. (Here, recall is the number of snippets the pattern correctly identifies as containing a definition, over the total number of snippets that contain a definition.) In multi-source document collections, like the Web or the TREC documents, one can expect to find several definitions of the same term, phrased in different ways. Unlike traditional document retrieval tasks, we are not interested in obtaining all the definitions; a single correct one suffices. Furthermore, as the number of snippets that can be reported is limited, we need to be confident that the snippets we return are indeed definitions. Hence, we need to rely on high-precision indicators. Preliminary experiments we performed using the F-measure (with $\beta=1$), a combination of recall and precision, instead of precision led to inferior results, confirming that attribute recall is not helpful in this task.

We have also experimented with information gain. In that case, one selects the *n*-grams with the *m* highest *IG(C,X)* scores, defined below, where *C* and *X* are random variables denoting the category of a snippet and the value of the *n*-gram's binary attribute, respectively, and *H(C)* and *H(C|X)* are the entropy and conditional entropy of *C*. By selecting the attributes with the highest information gain scores, one selects the attributes that carry most information about the value of *C*.

$$IG(C,X) = H(C) - \sum_{x \in \{0,1\}} P(X=x) \cdot H(C \mid X=x)$$

Although information gain is one of the best attribute selection measures in text categorization (Yang and Pedersen, 1997), in our case it led to very few attributes with non-zero *IG(C,X)* scores (around 90 attributes from the entire dataset of section 4). This is because most of the *n*-grams are very rare (i.e., *P(X=0)* is very large), and their absence (*X=0*) provides very little information about *C* (i.e., $H(C) \approx H(C \mid X = 0)$). For example, not encountering [*dp* such as *qn*] provides very little information on whether or not the snippet is a definition. The experiments we performed with the resulting attributes led to inferior results, compared to those that we got via precision.

3.2.4 Configuration 4: discarding WordNet

The fourth configuration is identical to the third one, except that it does not use the attribute that shows if the snippet contains one of the best hypernyms of Prager *et al.* (The attribute is present in configurations 2 and 3). The intention is to explore if the performance of configuration 3 can be sustained without the use of WordNet.

4 Experimental results

We evaluated the baseline and the machine learning configurations of section 3 on the definition questions of TREC-9 (2000) and TREC-2001, the same data used by Prager *et al.* For each question, the TREC organizers provide the 50 most highly ranked documents that an IR engine returned from the TREC documents. The task is to return for each question five 250-character snippets from the corresponding 50 documents, such that at least one of the snippets contains an acceptable definition. Following Prager *et al.*, we count a snippet as containing an acceptable definition, if it satisfies the Perl answer patterns that the TREC organizers provide for the corresponding question (Voorhees, 2001). The answer patterns incorporate the correct responses of all the participants of the corresponding TREC competition. In TREC-9, the correlation between system rankings produced by answer patterns and rankings produced by humans was at the same level as the average correlation between rankings of different human assessors (Voorhees and Tice 2000). In TREC-2001, the correlation between patterns and judges was lower, but still similar for 250-character responses (Voorhees, 2001).

All the experiments with machine learning configurations were performed with 10-fold cross-validation. That is, the question set was divided into 10 parts, and each experiment was repeated 10 times. At each iteration, the questions of a different

part and the corresponding documents were reserved for testing, while the questions and documents of the remaining nine parts were used for training. (In configurations 3 and 4, pattern acquisition was repeated at each iteration.) Table 1 reports the total number of questions that each method managed to handle successfully over the ten iterations; i.e., questions with at least one acceptable definition in the five returned snippets. When questions for which there was no answer in the corresponding 50 documents and/or there was no answer pattern are excluded, the results are those shown in italics. The second row contains the results reported by Prager *et al.* (2001, 2002), while the third one shows the results of our reimplementation. We include six questions that Prager *et al.* appear to have excluded, which is why the total number of questions is different; there are also some implementation differences, as discussed in section 3.

Method	% questions handled correctly
Prager *et al.*	51.95 (80/154), *60.15 (80/133)*
baseline	50.00 (80/160), *58.39 (80/137)*
config. 1	61.88 (99/160), *72.26 (99/137)*
config. 2	63.13 (101/160), *73.72 (101/137)*
config. 3	72.50 (116/160), *84.67 (116/137)*
config. 4	71.88 (115/160), *83.94 (115/137)*

Table 1: Results on TREC-9 & TREC-2001 data

The SVM learner with roughly Joho *et al.*'s attributes (config. 1) clearly outperforms Prager *et al.*'s WordNet-based method. Adding Prager *et al.*'s method as an extra attribute to the SVM (config. 2) leads to only a marginal improvement. Automatic pattern acquisition (config. 3) is much more beneficial. Removing the attribute of the WordNet-based method (config. 4) caused the system to fail in only one of the questions that configuration 3 handled successfully, which again suggests that the WordNet-based method does not contribute much to the performance of configuration 3. This is particularly interesting for languages with no WordNet-like resources.

The results of configurations 3 and 4 in table 1 were obtained using the 200 *n*-gram attributes with the highest precision scores. When using the 100 *n*-gram attributes with the highest and the 100 *n*-gram attributes with the lowest precision scores, the results of configuration 3 were 70.63% and *82.48*%. When using all the *n*-gram attributes with non-zero information gain scores, the results of configuration 3 were 66.25% and *77.42*%.
Configurations 2 and 3 achieved inferior results with 300 highest-precision *n*-gram attributes (table 2), which may be a sign that low reliability *n*-grams are beginning to dominate the attribute set.

n-grams	config. 3 (%)	config. 4 (%)
100	68.13, *79.56*	70.00, *81.75*
200	72.50, *84.67*	71.88, *83.94*
300	68.75, *80.29*	71.25, *83.21*

Table 2: Results for variable number of *n*-grams

5 Related work

Ng *et al.* (2000) use machine learning (C5 with boosting) to classify and rank candidate answers, but do not treat definition questions in any special way, and use only four generic attributes across all question categories. Some of their worst results are for "What …?" questions, that presumably include a large number of definition questions.

Ittycheriah and Roukos (2002) employ a maximum entropy model to rank candidate answers, which uses a very rich set of attributes that includes 8,500 patterns. The latter are *n*-grams of words that occur frequently in answers of the training data, each associated with a two-word question prefix (e.g., "What is") that also has to be matched for the pattern to be satisfied. Unlike our work, the *n*-grams have to be five or more words long, and, in the case of definition questions, they do not need to be anchored at the term to define. Ittycheriah and Roukos (2002) do not provide separate figures on the performance of their system on definition questions.

Blair-Goldensohn *et al.* (2003) focus on definition questions, but aim at producing coherent multi-sentence definitions, rather than identifying single defining snippets. At the heart of their approach is a component that uses machine learning (Riper) to identify sentences that are candidates for inclusion in the multi-sentence definition. This component plays a role similar to that of our SVM learner, but it is intended to admit a larger range of sentences, and appears to employ only attributes conveying the position of the sentence in its document and the frequency of the definition term in the context of the sentence.

Automatically acquired *n*-gram patterns can also be used for query expansion in information retrieval, as in Agichtein *et al.* (2001).

6 Conclusions and future work

We have presented a new method to identify single-snippet definitions in question answering systems. Our method combines previously proposed techniques as attributes of an SVM learner, to which an automatic pattern acquisition process contributes additional attributes. We have

evaluated several configurations of our method on TREC data, with results indicating it outperforms previous techniques.

The performance of our method may improve if *n*-grams that start or end within a margin of a few tokens from the term to define are added. This may allow definitions like "X, that Y defined as …" to be found. Further improvements may be possible by using a sentence splitter instead of windows of fixed length, anaphora resolution, clustering of similar snippets to avoid ranking them separately, and identifying additional *n*-gram attributes by bootstrapping (Ravichandran *et al.* 2003).

We believe that it is possible to address the post-2003 TREC task for definition questions with the same approach, but training the SVM learner to identify snippets that should be included in multi-snippet definitions. With sufficient training, we expect that *n*-grams indicative of information commonly included in multi-snippet definitions (e.g., dates of birth, important works for persons) will be discovered. Larger amounts of training data, however, will be required. We are currently working on a method to generate training examples in an unsupervised manner from parallel texts.

References

E. Agichtein, S. Lawrence, and L. Gravano. 2001. Learning Search Engine Specific Query Transformations for Question Answering. Proceedings of *WWW-10*, Hong Kong, pp. 169–178.

S. Blair-Goldensohn, K.R. McKeown, and A.H. Schlaikjer. 2003. A Hybrid Approach for Answering Definitional Questions. Technical Report CUCS-006-03, Columbia University.

S. Buchholtz and W. Daelemans. 2001. Complex Answers: A Case Study Using a WWW Question Answering System. *Natural Language Engineering*, 7(4):301–323.

J. Chu-Carroll, K. Czuba, J. Prager, and A. Ittycheriah. 2003. In Question Answering, Two Heads are Better than One. Proceedings of *HLT-NAACL 2003*, Edmonton, Canada, pp. 24–31.

M.A. Hearst. 1998. Automated Discovery of Wordnet Relations. In C. Fellbaum (Ed.), *WordNet: An Electronic Lexical Database*. MIT Press.

A. Ittycheriah and S. Roukos. 2002. IBM's Statistical Question Answering System – TREC-11. Proceedings of *TREC-2002*.

H. Joho and M. Sanderson. 2000. Retrieving Descriptive Phrases from Large Amounts of Free Text. Proceedings of the *9th ACM Conference on Information and Knowledge Management*, McLean, VA, pp. 180–186.

H. Joho and M. Sanderson. 2001. Large Scale Testing of a Descriptive Phrase Finder. Proceedings of the *1st Human Language Technology Conference*, San Diego, CA, pp. 219–221.

H.T. Ng, J.L.P. Kwan, and Y. Xia. 2001. Question Answering Using a Large Text Database: A Machine Learning Approach. Proceedings of *EMNLP 2001*, Pittsburgh, PA, pp. 67–73.

J. Prager, D. Radev, and K. Czuba. 2001. Answering What-Is Questions by Virtual Annotation. Proceedings of the *1st Human Language Technology Conference*, San Diego, CA, pp. 26–30.

J. Prager, J. Chu-Carroll, and K. Czuba. 2002. Use of WordNet Hypernyms for Answering What-Is Questions. Proceedings of *TREC-2001*.

D.R. Radev and K.R. McKeown. 1997. Buliding a Generation Knowledge Source using Internet-Accessible Newswire. Proceedings of the *5th ANLP*, Washington, D.C., pp. 221–228.

D.R. Radev, J. Prager, and V. Samn. 2000. Ranking Suspected Answers to Natural Language Questions Using Predictive Annotation. Proceedings of *NAACL/ANLP-2000*, Seattle, WA, pp. 150–157.

D. Ravichandran, E. Hovy, and F.J. Och. 2003. Statistical QA – Classifier vs. Ranker: What's the Difference? Proceedings of the ACL workshop on *Multilingual Summarization and Question Answering*, Sapporo, Japan.

D. Ravichandran, A. Ittycheriah, and S. Roukos. 2003. Automatic Derivation of Surface Text Patterns for a Maximum Entropy Based Question Answering System. Proceedings of *HLT-NAACL 2003*, Edmonton, Canada, pp. 85–87.

B. Schölkopf and A. Smola. 2002. *Learning with Kernels*. MIT Press.

E.M. Voorhees and D.M. Tice. 2000. Building a Question Answering Test Collection. Proc. of *SIGIR-2000*, Athens, Greece, pp. 200–207.

E.M. Voorhees. 2001. *The TREC QA Track*. *Natural Language Engineering*, 7(4):361–378.

E.M. Voorhees. 2003. *Evaluating Answers to Definition Questions*. Proceedings of HLT-NAACL 2003, Edmonton, Canada, pp. 109–111.

Y. Yang and J.O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. Proceedings of the *14th International Conference on Machine Learning*, Nashville, TN, pp. 412–420.