Finding Short Definitions of Terms on Web Pages

Gerasimos Lampouras[*] and **Ion Androutsopoulos**[*+]
[*]Department of Informatics, Athens University of Economics and Business, Greece
[+]Digital Curation Unit, Research Centre "Athena", Athens, Greece

Abstract

We present a system that finds short definitions of terms on Web pages. It employs a Maximum Entropy classifier, but it is trained on automatically generated examples; hence, it is in effect unsupervised. We use ROUGE-W to generate training examples from encyclopedias and Web snippets, a method that outperforms an alternative centroid-based one. After training, our system can be used to find definitions of terms that are not covered by encyclopedias. The system outperforms a comparable publicly available system, as well as a previously published form of our system.

1 Introduction

Definitions of terms are among the most common types of information users search for on the Web. In the TREC 2001 QA track (Voorhees, 2001), where the distribution of question types reflected that of real user logs, 27% of the questions were requests for definitions (e.g., "What is gasohol?", "Who was Duke Ellington?"). Consequently, some Web search engines provide special facilities (e.g., Google's "define:" query prefix) that seek definitions of user-specified terms in online encyclopedias or glossaries; to save space, we call both "encyclopedias". There are, however, often terms that are too recent, too old, or less widely used to be included in encyclopedias. Their definitions may be present on other Web pages (e.g., newspaper articles), but they may be provided indirectly (e.g., "He said that *gasohol, a mixture of gasoline and ethanol*, has been great for his business.") and they may be difficult to locate with generic search engines that may return dozens of pages containing, but not defining the terms.

We present a system to find short definitions of user-specified terms on Web pages. It can be used as an add-on to generic search engines, when no definitions can be found in on-line encyclopedias. The system first invokes a search engine using the (possibly multi-word) term whose definition is sought, the *target term*, as the query. It then scans the top pages returned by the search engine to locate 250-character snippets with the target term at their centers; we call these snippets *windows*. The windows are candidate definitions of the target term, and they are then classified as acceptable (positive class) or unacceptable (negative class) using supervised machine learning. The system reports the windows for which it is most confident that they belong in the positive class. Table 1 shows examples of short definitions found by our system. In our experiments, we allow the system to return up to five windows per target term, and the system's response is counted as correct if any of the returned windows contains an acceptable short definition of the target. This is similar to the treatment of definition questions in TREC 2000 and 2001 (Voorhees, 2000; Voorhees, 2001), but the answer is sought on the Web, not in a given document collection of a particular genre.

More recent TREC QA tracks required definition questions to be answered by lists of complementary text snippets, jointly providing required or optional information nuggets (Voorhees, 2003). In contrast, we focus on locating single snippets that include self-contained short definitions. Despite its simpler nature, we believe the task we address is of practical use: a list of single-snippet definitions from Web pages accompanied by the source URLs is a good starting point for users seeking definitions of terms not covered by encyclopedias. We also note that evaluating multi-snippet definitions can be problematic, because it is often difficult to agree which information nuggets should be treated as required, or even optional (Hildebrandt et al., 2004). In contrast, earlier experimental results we have reported (Androutsopoulos and Galanis, 2005) show strong inter-assessor agreement ($K > 0.8$) for single-snippet definitions (Eugenio and Glass, 2004). The task we address also differs from DUC's query focused summarization (Dang, 2005; Dang, 2006). Our queries are single terms, whereas DUC queries are longer topic

Target term: Babesiosis
(...) Babesiosis is a rare, severe and sometimes fatal tick-borne disease caused by various types of Babesia, a microscopic parasite that infects red blood cells. In New York state, the causative parasite is babesia microti. Who gets Babesiosis? Babesiosis (...)
Target term: anorexia nervosa
(...) anorexia nervosa is an illness that usually occurs in teenage girls, but it can also occur in teenage boys, and adult women and men. People with anorexia are obsessed with being thin. They lose a lot of weight and are terrified of gaining weight. The (...)
Target term: Kinabalu
(...) one hundred and thirty eight kilometers from Kota Kinabalu, the capital of the Malaysian state of Sabah, rises the majestic mount Kinabalu. With its peak at 4,101 meters (and growing), mount Kinabalu is the highest mountain in south-east Asia. This (...)
Target term: Pythagoras
(...) Pythagoras of Samos about 569 BC - about 475 BC click the picture above to see eleven larger pictures Pythagoras was a Greek philosopher who made important developments in mathematics, astronomy, and the theory of music. The theorem now known as (...)
Target term: Sacajawea
(...) Sacajawea was a Shoshone Indian princess. The Shoshone lived from the rocky mountains to the plains. They lived primarily on buffalo meat. The shoshone traveled for many days searching for buffalo. They hunted on horseback using the buffalo for food (...)
Target term: tale of Genji
(...) the tale of Genji This site aims to promote a wider understanding and appreciation of the tale of Genji - the 11th century Japanese classic written by a Heian court lady known as Murasaki Shikibu. It also serves as a kind of travel guide to the world (...)
Target term: Jacques Lacan
(...) who is Jacques Lacan? John Haber in New York city a primer for pre-post-structuralists Jacques Lacan is a Parisian psychoanalyst who has influenced literary criticism and feminism. He began work in the 1950s, in the Freudian society there. It was a (...)

Table 1: Definitions found by our system.

descriptions, often entire paragraphs; furthermore, we do not attempt to compose coherent and cohesive summaries from several snippets.

The system we present is based on our earlier work (Miliaraki and Androutsopoulos, 2004), where an SVM classifier (Cristianini and Shawe-Taylor, 2000) was used to separate acceptable windows from unacceptable ones; the SVM also returned confidence scores, which were used to rank the acceptable windows. On datasets from the TREC 2000 and 2001 QA tracks, our earlier system clearly outperformed the methods of Joho and Sanderson (2000; 2001) and Prager et al. (2001; 2002), as reported in previous work (Miliaraki and Androutsopoulos, 2004). To train the SVM, however, thousands of training windows were required, each tagged as a positive or negative example. Obtaining large numbers of training windows is easy, but manually tagging them is very time-consuming. In the TREC 2000 and 2001 datasets, it was possible to tag the training windows automatically by using training target terms and accompanying regular expression patterns provided by the TREC organizers. The regular expressions covered all the known acceptable definitions of the corresponding terms that can be extracted from the datasets. When the training windows, however, are obtained from the Web, it is impossible to construct manually regular expressions for all the possible phrasings of the acceptable definitions in the training windows.

In subsequent work (Androutsopoulos and Galanis, 2005), we developed ATTW (automatic tagging of training windows), a technique that produces arbitrarily large collections of training windows from the Web with practically no manual effort, in effect making our overall system unsupervised. ATTW uses training terms for which several encyclopedia definitions are available, and compares each Web training window (each window extracted from the pages the search engine returned for a training term) to the corresponding encyclopedia definitions. Web training windows that are very similar (or dissimilar) to the corresponding encyclopedia definitions are tagged as positive (or negative) examples; if the similarity is neither too high nor too low, the window is not included in the classifier's training data. Previously reported experiments (Androutsopoulos and Galanis, 2005) showed that ATTW leads to significantly better results, compared to training the classifier on all the available TREC windows, for which regular expressions are available, and then using it to classify Web windows.

Note that in ATTW the encyclopedia definitions are used only during training. Once the classifier has been trained, it can be used to discover definitions on arbitrary Web pages. In fact, during testing we discard windows originating from on-line encyclopedias, simulating the case where we seek definitions of terms not covered by encyclopedias; we also ignore windows from on-line encyclopedias during training. Also, note that the classifier is trained on Web windows, not directly on encyclopedia definitions, which allows it to avoid relying excessively on phrasings that are common in encyclopedia definitions, but uncommon in more indirect definitions of arbitrary Web pages. Furthermore, training the classifier directly on encyclopedia definitions would not provide negative examples.

In our previous work with ATTW (Androutsopoulos and Galanis, 2005) we used a measure constructed by ourselves to assess the similarity between Web windows and encyclopedia definitions. Here, we use the more established ROUGE-W measure (Lin, 2004) instead. ROUGE-W and other versions of ROUGE have been used in summarization to measure how close a machine-authored summary is to multiple human summaries of the same input. We use ROUGE-W in a similar setting, to measure how close a training window is to multiple encyclopedia definitions of the same term. A further difference from our previous work is that we also use ROUGE-W when computing the features of the windows to be classified. Previously, the SVM relied, among others, on Boolean features indicating if the target term was preceded or followed in the window to be classified by a particular phrase indicating a definition (e.g., “target, a kind of”, “such as target”). The indicative phrases are selected automatically during training, but now the corresponding features are not Boolean; their values are the ROUGE-W similarity scores between an indicative phrase and the context of the target term in the window. This allows the system to soft-match the phrases to the windows (e.g., encountering “target, another kind of”, instead of “target, a kind of”).[1]

In our new system we also use a Maximum Entropy (MAXENT) classifier (Ratnaparkhi, 1997) instead of an SVM, because much faster implementations of the former are available.[2] We present experimental results showing that our new system significantly outperforms our previously published one. The use of the MAXENT classifier by itself improved slightly our results, but the improvements come mostly from using ROUGE-W.

Apart from presenting an improved version of our system, the main contribution of this paper is a detailed experimental comparison of our new system against Cui et al.’s (2004; 2005; 2006; 2007). The latter is particularly interesting, because it is well published, it includes both an alternative, centroid-based technique to automatically tag training examples and a soft-matching classifier, and it is publicly available.[3] We show that ATTW outperforms Cui et al.’s centroid-based technique, and that our overall system is also clearly better than Cui et al.’s in the task we address.

Section 2 discusses ATTW with ROUGE-W, Cui et al.’s centroid-based method to tag training examples, and experiments showing that ATTW is better. Section 3 describes our new overall system, the system of Cui et al., and the baselines. Section 4 reports experimental results showing that our system is better than Cui et al.’s, and better than our previously published system. Section 5 discusses related work; and section 6 concludes.

2 Tagging training windows

During both training and testing, for each target term we keep the r most highly ranked Web pages the search engine returns. We then extract the first f windows of the target term from each page, since early occurrences of the target terms on pages are more likely to be definitions. We, thus, obtain $r \cdot f$ windows per term.[4] When testing, we return the k windows of the target term that the classifier is most certain they belong in the positive class. In our experiments, $r = 10$, $f = 5$, $k = 5$. During training, we train the classifier on the $q \cdot r \cdot f$ windows we obtain for q training target terms; in our experiments, q ranged from 50 to 1500. Training requires tagging first the training windows as positive or negative, possibly discarding windows that cannot be tagged automatically.

2.1 ATTW with ROUGE-W similarity

To tag a training window w of a training term t with ATTW and ROUGE-W, we obtain a set C_t of definitions of t from encyclopedias.[5] Stop-words, punctuation, and non-alphanumeric characters are removed from C_t and w, and a stemmer is applied; the testing windows undergo the same preprocessing.[6] For each definition $d \in C_t$, we find the longest common word subsequence of w and d. If w is the word sequence $\langle A, B, F, C, D, E \rangle$

[1] We also experimented with other similarity measures (e.g., edit distance) and ROUGE variants, but we obtained the best results with ROUGE-W.

[2] We use Stanford’s classifier; see http://nlp.stanford.edu/.

[3] See http://www.cuihang.com/software.html. The software and a demo of our system, and the datasets we used are also freely available; see http://nlp.cs.aueb.gr/.

[4] We used Altavista in our experiments. We remove HTML tags and retain only the plain text of the pages.

[5] The training terms were randomly selected from the index of http://www.encyclopedia.com/. We used Google’s “define:” to obtain definitions from other encyclopedias.

[6] We use the 100 most frequent words of the BNC corpus (http://www.natcorp.ox.ac.uk/) as the stop-list, and Porter’s stemmer (http://tartarus.org/∼martin/PorterStemmer/).

and $d = \langle A, B, E, C, G, D \rangle$, the longest common subsequence is $\langle A, B, C, D \rangle$. The longest common subsequence is divided into consecutive matches, producing in our example $\langle A, B|C|D \rangle$. We then compute the following score (weighted longest common subsequence), where m is the number of consecutive matches, k_i is the length of the i-th consecutive match, and f is a weighting function. We use $f(k) = k^a$, where $a > 1$ is a parameter we tune experimentally.

$$WLCS(w, d) = \sum_{i=0}^{m} f(k_i)$$

We then compute the following quantities, where $|\cdot|$ is word length, and f^{-1} is the inverse of f.

$$P(w, d) = f^{-1}(\frac{WLCS(w,d)}{f(|w|)})$$
$$R(w, d) = f^{-1}(\frac{WLCS(w,d)}{f(|d|)})$$
$$F(w, d) = \frac{(1+\beta^2) \cdot R(w,d) \cdot P(w,d)}{R(w,d)+\beta^2 \cdot P(w,d)}$$

In effect, $P(w, d)$ examines how close the longest common substring is to w and $R(w, d)$ how close it is to d. Following Lin (2004), we use $\beta = 8$, assigning greater importance to $R(w, d)$. If $R(w, d)$ is high, the longest common substring is very similar to d; then w (which also includes the longest common substring) intuitively contains almost all the information of d, i.e., all the information of a known acceptable definition (high recall). If $P(w, d)$ is high, the longest common substring is very similar to w; then d (which also includes the longest common substring) contains almost all the information of w, i.e., w does not contain any (redundant) information not included in a known acceptable definition, something we care less for.

The ROUGE-W similarity $sim(w, C_t)$ between w and C_t is the maximum $F(w, d)$, for all $d \in C_t$. Training windows with $sim(w, C_t) > T_+$ are tagged as positive; if $sim(w, C_t) < T_-$, they are tagged as negative; and if $T_- \leq sim(w, C_t) \leq T_+$, they are discarded. We tune the thresholds T_+ and T_- experimentally, as discussed below.

2.2 The centroid-based tagging approach

This method is used in the system of Cui et al. (2004; 2005; 2006; 2007). For each training target term, we construct a "centroid" pseudo-text containing the words that co-occur most frequently with the target term. We then compute the similarity between each training window and the centroid of its target term. If it exceeds a threshold, the window is tagged as positive; Cui et al. produce only positive examples.

The centroid of a training target term t is constructed as follows. For each word u in t's training windows, we compute the centrality score defined below, where SF_t is the number of t's training windows, SF_u is the number of u's windows that can be extracted from the retained Web pages the search engine returned for t, $SF_{t \cap u}$ is the number of windows on the same pages that contain both t and u, and *idf*(u) is the inverse document frequency of w.[7] Centrality scores are pointwise mutual information with an extra $idf(u)$ factor.

$$centrality(u) = -log(\frac{SF_{t \cap u}}{SF_t + SF_u}) \cdot idf(u)$$

The words u whose centrality scores exceed the mean by at least a standard deviation are added to the centroid of t. Before computing the centrality scores, stop-words, punctuation, and non-alphanumeric characters are removed, and a stemmer is applied, as in ATTW. The similarities between training windows and centroids are then computed using cosine similarity, after turning the centroids and windows into binary vectors that show which words they contain.

2.3 Comparing the tagging approaches

To evaluate the two methods that tag training windows, we selected randomly $q = 200$ target terms, different from those used for training and testing. We collected the $q \cdot r \cdot f = 200 \cdot 10 \cdot 5$ windows from the corresponding Web pages, we selected randomly 400 from the collected 10,000 windows, and tagged them manually as positive or negative.

Figure 1 plots the positive precision of the two methods against their positive recall, and figure 2 shows negative precision against negative recall. For different values of T_+, we obtain a different point in figure 1; similarly for T_- and figure 2. Positive precision is $TP/(TP + FP)$, positive recall is $TP/(TP + FN)$, and likewise for negative precision and recall; TP (true positives) are the positive training windows the method has correctly tagged as positive, FP are the negative windows the method has tagged as positives etc.

For very high (strict) T_+ values, the methods tag very few (or none) training windows as positive; hence, both TP and $TP + FP$ approach (or become) zero; we take positive precision to be zero in that case. Positive recall also approaches (or becomes) zero, which is why both positive recall and

[7] We obtained $idf(u)$ from BNC. Cui et al. use sentences instead of windows, reducing the risk of truncating definitions. We used windows in all systems, to compare fairly.

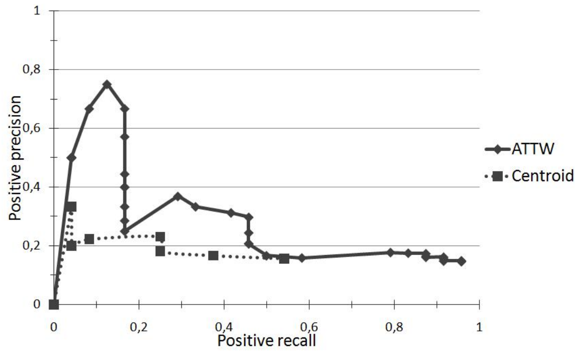

Figure 1: Results of generating positive examples.

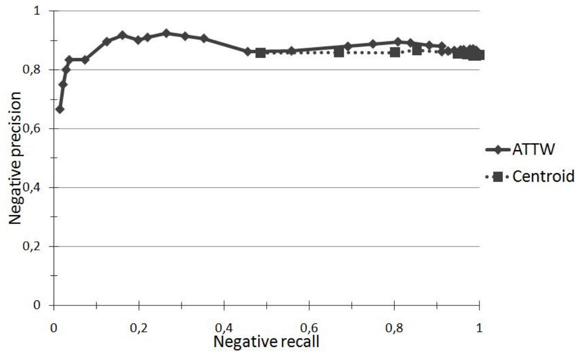

Figure 2: Results of generating negative examples.

precision reach zero in the left of figure 1. Similar comments apply to figure 2, though both methods always tagged correctly at least a few training windows as negative, for the T_{-} values we tried; hence, negative precision was never zero.

Positive precision shows how certain we can be that training windows tagged as positive are indeed positive; whereas positive recall is the percentage of true positive examples that we manage to tag as such. Figure 1 shows that when using ATTW, we need to settle for a low positive recall, i.e., miss out many positive examples, in order to obtain a reasonably high precision. It also shows that the centroid method is clearly worse when tagging positive examples; its positive precision is almost always less than 0.3. Figure 2 shows that both methods achieve high negative precision and recall; they manage to assign trustworthy negative labels without missing many negative examples. However, ATTW is significantly better when tagging positive examples, as shown in figure 1; hence, it is better than the centroid method.[8]

When using ATTW in practice, we need to select T_{+} and T_{-}. We assign more importance to selecting a T_{+} (a point of ATTW's curve in figure 1) that yields high positive precision; the choice of T_{-} (point in figure 2) is less important, because ATTW's negative precision is always reasonably high. Based on figure 1, we set T_{+} to 0.58, which corresponds to positive precision 0.66 and positive recall 0.16. By tuning the two thresholds we can control the number of positively or negatively tagged examples we produce (and their ratio), and the number of examples we discard. Having set T_{+}, we set T_{-} to 0.30, a value that maintains the ratio of truly positive to truly negative windows of the 400 manually tagged windows (0.2 to 1), since this is approximately the ratio the classifier will confront during testing; we also experimented with a 1 to 1 ratio, but the results were worse. This T_{-} value corresponds negative precision 0.70 and negative recall 0.02. Thus, both positive and negative precision is approximately 0.7, which means that approximately 30% of the tags we assign to the examples are incorrect. Our experiments, however, indicate that the classifier is able to generalize well over this noise.

3 Finding new definitions

We now present our overall system, the system of Cui et al., and the baselines.

3.1 Our system

Given a target term, our system extracts $r \cdot f = 10 \cdot 5$ windows from the pages returned by the search engine, and uses the MAXENT classifier to separate them into acceptable and unacceptable definitions.[9] It then returns the $k = 5$ windows the classifier is most confident they are acceptable. The classifier is trained on windows tagged as positive or negative using ATTW. It views each window as a vector of the following features:[10]

SN: The ordinal number of the window on the page it originates from (e.g., second window of the target term from the beginning of the page). Early mentions of a term are more likely to define it.

RK: The ranking of the Web page the window originates from, as returned by the search engine.

[8] We tried different values of ROUGE-W's a parameter in the interval $(1, 2]$. We use $a = 1.4$, which was the value with the best results on the 400 windows. We did not try $a > 2$, as the results were declining as a approached 2.

[9] We do not discuss MAXENT classifiers, since they are a well documented in the literature.

[10] *SN* and *WC* originate from Joho and Sanderson (2000).

WC: We create a simple centroid of the window's target term, much as in section 2.2. The centroid's words are chosen based on their frequency in the $r \cdot f$ windows of the target term; the 20 most frequent words are chosen. WC is the percentage of the 20 words that appear in the vector's window.

Manual patterns: 13 Boolean features, each signaling if the window matches a different manually constructed lexical pattern (e.g., "target, a/an/the", as in "Tony Blair, the British prime minister"). The patterns are those used by Joho and Sanderson (2000), and four more introduced in our previous work (Androutsopoulos and Galanis, 2005) and (Miliaraki and Androutsopoulos, 2004). They are intended to perform well across text genres.

Automatic patterns: m numeric features, each showing the degree to which the window matches a different automatically acquired lexical pattern. The patterns are word n-grams ($n \in \{1, 2, 3\}$) that must occur directly before or after the target term (e.g., "*target* which is"). The patterns are acquired as follows. First, all the n-grams directly before or after any target term in the training windows are collected. The n-grams that have been encountered at least 10 times are candidate patterns. From those, the m patterns with the highest precision scores are retained, where precision is the number of positive training windows the pattern matches over the total number of training windows it matches; we use $m = 300$ in our experiments, based on the results of our previous work. The automatically acquired patterns allow the system to detect definition contexts that are not captured by the manual patterns, including genre-specific contexts. The value of each feature is the ROUGE-W score between a pattern and the left or right context of the target term in the window.

3.2 Cui et al.'s system

Given a target term t, Cui et al. (2004; 2005; 2006; 2007) initially locate sentences containing t in relevant documents. We use the $r \cdot f = 10 \cdot 5$ windows from the pages returned by the search engine, instead of sentences. Cui et al. then construct the centroid of t, and compute the cosine similarity of each one of the $r \cdot f$ windows to the centroid, as in section 2.2. The 10 windows that are closer to the centroid are considered candidate answers. All candidate answers are then processed by a part-of-speech (POS) tagger and a chunker. The words of the centroid are replaced in all the candidate answers by their POS tags; the target term, noun phrases, forms of the verb "to be", and articles are replaced by special tags (e.g., TARGET, NP), while adjectives and adverbs are removed. The candidate answers are then cropped to L tokens to the left and right of the target term, producing two subsequences (left and right) per candidate answer; we set $L = 3$, which is Cui et al.'s default.

Cui et al. experimented with two approaches to rank the candidate answers, called Bigram Model and Profile Hidden Markov Model (PHMM). Both are learning components that produce soft patterns, though PHMM is much more complicated. In their earlier work, Cui et al. (2005) found the Bigram Model to perform better than PHMM; in more recent experiments with more data (Cui, 2006; Cui et al., 2007) they found PHMM to perform better, but the difference was not statistically significant. Given these results and the complexity of PHMM, we experimented only with the Bigram Model.

In the Bigram Model, the left and right subsequences of each candidate answer are considered separately. Below $S_1, \dots, S_L$ refer to the slots (word positions) of a (left or right) subsequence, and $t_1, \dots, t_L$ to the particular words in the slots. For each subsequence $\langle S_1 = t_1, \dots, S_L = t_L \rangle$ of a candidate answer, we first estimate:

$$\begin{aligned} P(t_i|S_i) &= \frac{|S_i(t_i)| + \delta}{\sum_{t'} |S_i(t_i)| + \delta \cdot N} \\ P(t_i|t_{i-1}) &= \frac{|S_i(t_i) \wedge S_{i-1}(t_{i-1})|}{|S_i(t_i)|} \end{aligned}$$

$P(t_i|S_i)$ is the probability that t_i will appear in slot S_i of a left or right subsequence (depending on the subsequence considered) of an acceptable candidate answer. $P(t_i|t_{i-1})$ is the probability that t_i will follow t_{i-1} in a (left or right) subsequence of an acceptable candidate answer. Cui et al. use only positive training examples, generated by the centroid-based approach of section 2.2. $|S_i(t_i)|$ is the number of times t_i appeared in S_i in the (left or right) subsequences of the training examples. t' ranges over all the words that occurred in S_i in the training examples. $|S_i(t_i) \wedge S_{i-1}(t_{i-1})|$ is the number of times t_i and t_{i-1} co-occurred in the corresponding slots in the training examples. N is the number of different words that occurred in the (left or right) training subsequences, and δ is a constant set to 2, as in Cui et al.'s experiments. Following Cui et al., if t_i is a POS or other special tag then the probabilities above are estimated by counting only the tags of the training examples. Similarly, if t_i is an actual word, only the actual words (not tags) of the training examples are considered.

The probability of each subsequence could then be estimated as:

$$P(t_1, \ldots, t_L) = P(t_1|S_1) \cdot \prod_{i=2}^{L} (\lambda \cdot P(t_i|t_{i-1}) + (1 - \lambda) \cdot P(t_i|S_i))$$

Instead, Cui et al. use the following scoring measure, which also accounts for the fact that some subsequences may have length $l < L$. They tune λ by Expectation Maximization.

$$P_{norm}(t_1, \ldots, t_L) = \frac{1}{l} \cdot [\log P(t_1|S_1) + \sum_{i=2}^{L} \log(\lambda \cdot P(t_i|t_{i-1}) + (1 - \lambda) \cdot P(t_i|S_i))]$$

The overall score of a candidate answer is then:

$$P = (1 - \alpha) \cdot P_{norm}(\mathit{left}) + \alpha \cdot P_{norm}(\mathit{right})$$

Again, Cui et al. tune a by Expectation Maximization. Instead, we tuned λ and α by a grid search in $[0, 1] \times [0, 1]$, with step 0.1 for both parameters. For the tuning, we trained Cui et al.'s system on 2,000 randomly selected target terms, excluding terms used for other purposes. We used 160 manually tagged windows to evaluate the system's performance with the different values of λ and α; the 160 windows were selected randomly from the 10,000 windows of section 2.3, after excluding the 400 manually tagged windows of that section. The resulting values for λ and α were 0.7 and 0.6, respectively. Apart from the modifications we mentioned, we use Cui et al.'s original implementation.

3.3 Baseline methods

The first baseline selects the first window of each one of the five highest ranked Web pages, as returned by the search engine, and returns the five windows. The second baseline returns five windows chosen randomly from the $r \cdot f = 10 \cdot 5$ available ones. The third baseline (centroid baseline) creates a centroid of the $r \cdot f$ windows, as in section 2.2, and returns the five windows with the highest cosine similarity to the centroid.[11]

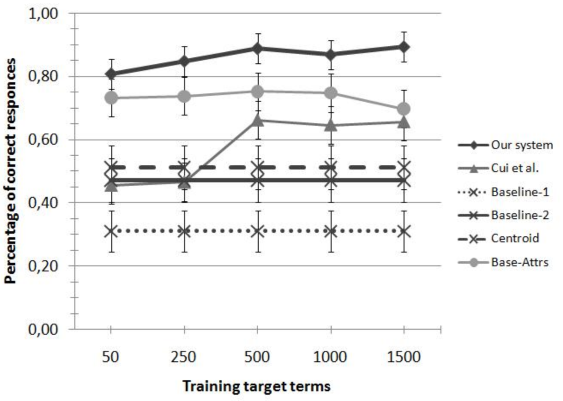

Figure 3: Correct responses, 5 answers/question.

4 Evaluation of systems

We used q training target terms in the experiments of this section, with q ranging from 50 to 1500, and 200 testing terms, with no overlap between training and testing terms, and excluding terms that had been used for other purpose.[12] We had to use testing terms for which encyclopedia definitions were also available, to judge the acceptability of the systems' responses, since many terms are highly technical. We discarded, however, windows extracted from encyclopedia pages when testing, simulating the case where the target terms are not covered by encyclopedias.

As already mentioned, for each target term we extract $r \cdot f = 10 \cdot 5$ windows (or fewer, if fewer are available) from the pages the search engine returns. We then provide these windows to each of the systems, allowing them to return up to $k = 5$ windows, ordered by decreasing confidence. If any of the k windows contains an acceptable short definition of the target term, as judged by a human evaluator, the system's response is counted as correct. We also calculate the Mean Reciprocal Rank (MRR) of each system's responses, as in the TREC QA track: if the first acceptable definition of a response is in the j-th position $(1 \leq j \leq k)$, the response's score is $1/j$; MRR is the mean of the responses' scores, i.e., it rewards systems that return acceptable definitions higher in their responses.

Figures 3 and 4 show the results of our experiments as percentage of correct responses and MRR, respectively; the error bars of figure 3 correspond to 95% confidence intervals. Our system clearly outperforms Cui et al.'s, despite the fact that the

[11] We also reimplemented the definitions component of Chu-Carroll et al. (2004; 2005), but its performance was worse than our centroid baseline.

[12] The reader is reminded that all terms were selected randomly from the index of an on-line encyclopedia.

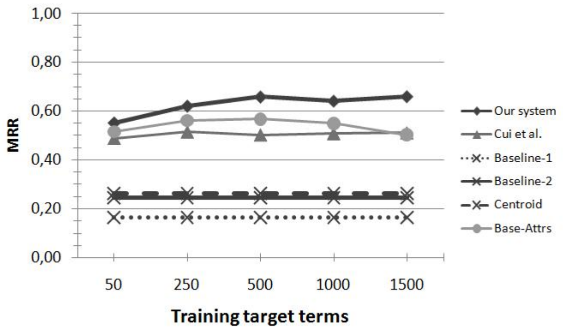

Figure 4: MRR scores, 5 answers per question.

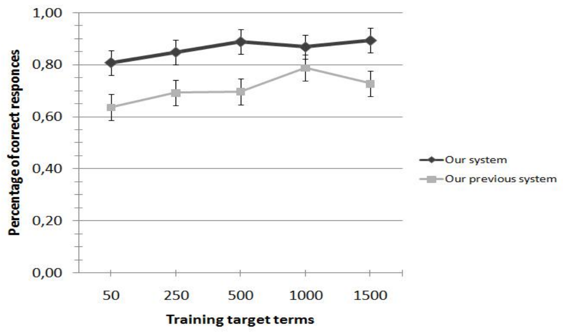

Figure 5: Correct responses of our new and previous system, allowing 5 answers per question.

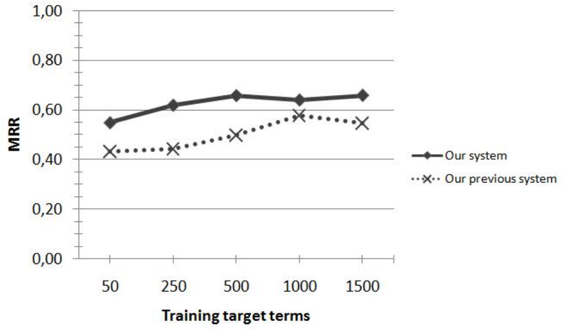

Figure 6: MRR of our new and previous system.

latter uses more linguistic resources (a POS tagger and a chunker). Both systems outperform the baselines, of which the centroid baseline is the best, and both systems perform better as the size of the training set increases. The baselines contain no learning components; hence, their curves are flat. We also show the results (Base-Attrs) of our system when the features that correspond to automatically acquired patterns are excluded. Clearly, these patterns help our system achieve significantly better results; however, our system outperforms Cui et al.'s even without them. Without the automatic patterns, our system also shows signs of saturation as the training data increase.

Figures 5 and 6 show the performance of our new system against our previously published one (Androutsopoulos and Galanis, 2005); the new system clearly outperforms the old one. Additional experiments we conducted with the old system replacing the SVM by the MAXENT classifier (without using ROUGE-W) indicate that the use of MAXENT by itself also improved slightly the results, but the differences are too minor to show; the improvement is mostly due to the use of ROUGE-W instead of our previous measure.

5 Related work

Xu et al. (2004) use an information extraction engine to extract linguistic features from documents relevant to the target term. The features are mostly phrases, such as appositives, and phrases expressing relations. The features are then ranked by their type and similarity to a centroid, and the most highly ranked ones are returned. Xu et al. seem to aim at generating multi-snippet definitions, unlike the single-snippet definitions we seek.

Blair-Goldensohn et al. (2003; 2004) extract sentences that may provide definitional information from documents retrieved for the target term; a decision tree learner and manually tagged training data are used. The sentences are then matched against manually constructed patterns, which operate on syntax trees, to detect sentences expressing the target term's genus, species, or both (genus+species). The system composes its answer by placing first the genus+species sentence that is closer to the centroid of the extracted sentences. The remaining sentences are ranked by their distance from the centroid, and the most highly ranked ones are clustered. The system then selects iteratively the cluster that is closer to the centroid of the extracted sentences and the most recently used cluster. The cluster's most representative sentence, i.e., the sentence closest to the centroid of the cluster's sentences, is added to the response. The iterations stop when a maximum response length is reached. Multi-snippet definitions are generated.

Han et al. (2004; 2006) parse a definition question to locate the head word of the target term. They also use a named entity recognizer to determine the target term's type (person, organization, etc.). They then extract from documents relevant to the target term sentences containing its head word, as well as sentences the extracted ones refer to (e.g., via pronouns). The resulting sentences are matched against manually constructed syntactic patterns to detect phrases conveying definitional information. The resulting phrases are ranked by criteria like the degree to which the phrase contains words common in definitions of the target term's type, and the highest ranked phrases are included in a multi-snippet summary. Other mechanisms discard phrases duplicating information.

Xu et al. (2005) aim to extract all the definitions in a document collection. They parse the documents to detect base noun phrases (without embedded noun phrases). Base noun phrases are possible target terms; the paragraphs containing them are matched against manually constructed patterns that look for definitions. An SVM then separates the remaining paragraphs into good, indifferent, and bad definitions. Redundant paragraphs, identified by edit distance similarity, are removed.

6 Conclusions and future work

We presented a freely available system that finds short definitions of user-specified terms on Web pages. It employs a MAXENT classifier, which is trained on automatically generated examples; hence, the system is in effect unsupervised. We use ROUGE-W to generate training examples from Web snippets and encyclopedias, a method that outperforms an alternative centroid-based one. Once our system has been trained, it can find short definitions of terms that are not covered by encyclopedias. Experiments show our system outperforms a comparable well-published system and a previously published form of our system.

Our system does not require linguistic processing tools, such as named entity recognizers, POS taggers, chunkers, parsers; hence, it can be easily used in languages where such tools are unavailable. It could be improved by exploiting the HTML markup of Web pages and the Web's hyperlinks. For example, the target term is sometimes written in italics in definitions, and some definitions are provided on pages (e.g., pop-up windows) that occurrences of the target term link to.

The work reported here was conducted in the context of project INDIGO, where an autonomous robotic guide for museum collections is being developed (Galanis et al., 2009). The guide engages the museum's visitors in spoken dialogues, and it describes the exhibits the visitors select by generating spoken natural language descriptions from an ontology. Among other requests, the visitors can ask follow up questions, and we have found that the most common kind of follow up questions are requests to define terms (e.g., names of persons, events, architectural terms, etc.) mentioned in the generated exhibit descriptions. Some of these definition requests can be handled by generating new texts from the ontology, but some times the ontology contains no information for the target terms. We are, thus, experimenting with the possibility of obtaining short definitions from the Web, using the system we presented.

Acknowledgements

This work was carried out in INDIGO, an FP6 IST project funded by the European Union, with additional funding provided by the Greek General Secretariat of Research and Technology.[13]

References

Androutsopoulos, I., and Galanis, D. 2005. *A Practically Unsupervised Learning Method to Identify Single-Snippet Answers to Definition Questions on the Web.* In HLT/EMNLP, Vancouver, Canada, 323–330.

Blair-Goldensohn, S., McKeown, K., Schlaikjer, A.H. 2003. *A Hybrid Approach for QA Track Definitional Questions.* In TREC 2003, Gaithersburg, MD, USA.

Blair-Goldensohn, S., McKeown, K.R., and Schlaikjer, A.H. 2004. *Answering Definitional Questions: A Hybrid Approach.* In Maybury, M. (Ed.), New Directions in Question answering, AAAI Press.

Chu-Carroll, J., Czuba, K., Prager, J., Ittycheriah, A., Blair-Goldensohn, S. 2004. *IBM's PIQUANT II in TREC 2004.* In TREC 2004, Gaithersburg, MD, USA.

Chu-Carroll, J., Czuba, K., Duboue, P., and Prager, J. 2005. *IBM's PIQUANT II in TREC 2005.* In TREC 2005, Gaithersburg, MD, USA.

Cristianini, N. and Shawe-Taylor, J. 2000. *An Introduction to SVMs.* Cambridge University Press.

Cui, H., Kan, M.-Y., Chua, T.-S., and Xiao, J. 2004. *A Comparative Study on Sentence Retrieval for Definitional Question Answering.* In SIGIR workshop on Information Retrieval for Question Answering, Salvador, Brazil.

[13] Consult http://www.ics.forth.gr/indigo/.

Cui, H., Kan, M.Y., Chua, T.S. 2004. *Unsupervised Learning of Soft Patterns for Generating Definitions from Online News*. In WWW, New York, NY, USA.

Cui, H., Kan, M.Y., Chua, T.S. 2005. *Generic Soft Pattern Models for Definitional Question Answering*. In ACM SIGIR, Salvador, Brazil.

Cui, H. 2006. *Soft Matching for Question Answering*. Ph.D. thesis, National University of Singapore.

Cui, H., Kan, M., and Chua, T. 2007. *Soft Pattern Matching Models for Definitional Question Answering*. ACM Transactions on Information Systems, 25(2):1–30.

Dang, H. T. 2005. *Overview of DUC 2005*. In DUC at HLT-EMNLP, Vancouver, Canada.

Dang, H. T. 2006. *Overview of DUC 2006*. In DUC at HLT-NAACL, New York, NY, USA.

Eugenio, B. D., Glass, M. 2004. *The Kappa Statistic: a Second Look*. Computational Linguistics, 301(1):95-101.

Galanis, D., Karakatsiotis, G., Lampouras, G., and Androutsopoulos, I. 2009. *An Open-Source Natural Language Generator for* OWL *Ontologies and its Use in Protege and Second Life*. EACL system demonstration, Athens, Greece.

Han, K.S., Chung, H., Kim, S.B., Song, Y.I., Lee, J.Y., Rim, H.C. 2004. *Korea University QA System at TREC 2004*. In TREC 2004, Gaithersburg, MD, USA.

Han, K.S., Song, Y.I., Kim, S.B., and Rim, H.C. 2006. *A Definitional Question Answering System Based on Phrase Extraction Using Syntactic Patterns*. IEICE Transactions on Information and Systems, vol. E89-D, No. 4, 1601–1605.

Hildebrandt, W., Katz, B., and Lin, J. 2004. *Answering Definition Questions Using Multiple Knowledge Sources*. In HLT-NAACL, Boston, MA, USA, 49–56.

Joho, H. and Sanderson, M. 2000. *Retrieving Descriptive Phrases from Large Amounts of Free Text*. International Conference on Information and Knowledge Management, McLean, VA, USA, 180–186.

Joho, H. and Sanderson, M. 2001. *Large Scale Testing of a Descriptive Phrase Finder*. In HLT-NAACL, San Diego, CA, USA, 219–221.

Lin, C.Y. 2004. *ROUGE: A Package for Automatic Evaluation of Summaries*. In ACL workshop “Text Summarization Branches Out”, Barcelona, Spain.

Miliaraki, S. and Androutsopoulos, I. 2004. *Learning to Identify Single-Snippet Answers to Definition Questions*. In COLING, Geneva, Switzerland, 1360–1366.

Prager, J., Radev, D., and Czuba, K. 2001. *Answering What-Is Questions by Virtual Annotation*. In HLT-NAACL, San Diego, CA, USA, 26–30.

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

Ratnaparkhi A. 1997. *A Simple Introduction to Maximum Entropy Models for Natural Language Processing*. Technical Report 97-08, Institute for Research in Cognitive Science, University of Pennsylvania, 1997.

Voorhees, E.M. 2000. *Overview of the TREC-9 Question Answering Track*. NIST, USA.

Voorhees, E.M. 2001. *Overview of the TREC 2001 Question Answering Track*. NIST, USA.

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

Voorhees, E.M. 2003. *Evaluating Answers to Definition Questions*. In HLT-NAACL, Edmonton, Canada.

Xu, J., Weischedel, R., Licuanan, A. 2004. *Evaluation of an Extraction-based Approach to Answering Definitional Questions*. In ACM SIGIR, Sheffield, UK.

Xu, J., Cao, Y., Li, H., Zhao, M. 2005. *Ranking Definitions with Supervised Learning Methods*. In WWW, Chiba, Japan, 811–819.