USING DEPENDENCY PATHS FOR ANSWERING DEFINITION QUESTIONS ON THE WEB

Alejandro Figueroa
Deutsches Forschungszentrum für Künstliche Intelligenz - DFKI, Stuhlsatzenhausweg 3, D - 66123, Saarbrücken, Germany

John Atkinson
Department of Computer Sciences, Universidad de Concepción, Concepción, Chile

Keywords: Web question answering, Definition questions, Lexical dependency paths, n-Gram language models.

Abstract: This work presents a new approach to automatically answer definition questions from the Web. This approach learns n-gram language models from lexicalised dependency paths taken from abstracts provided by *Wikipedia* and uses context information to identify candidate descriptive sentences containing target answers. Results using a prototype of the model showed the effectiveness of lexicalised dependency paths as salient indicators for the presence of definitions in natural language texts.

1 INTRODUCTION

In the context of web question-answering systems, definition questions differ markedly from standard factoid questions. Factoid questions require a single fact to be returned to the user, whereas, definition questions require a substantially more complex response which succinctly defines the topic of the question (a.k.a. *definiendum* or target) which the user wishes to know about.

Definition questions have become especially interesting in recent years as about 25% of the questions in real user logs and queries submitted to search engines are requests for definitions (Rose and Levinson, 2004). Question Answering Systems that focus on discovering answers to definition questions usually aim at finding succinct, diverse and accurate factual information about the *definiendum*. These pieces of information are usually called *nuggets* or "*Semantic Context Units*". Specifically, answers to questions about politicians would then comprise important dates in their lives (birth, marriage and death), their major achievements, and any other interesting items, such as party membership or leadership. For instance, an answer to the question"*Who is Gordon Brown?*" would contain the following descriptive sentence:

```
Gordon Brown is a British politician and
leader of the Labour Party.
```

Accordingly, this paper investigates the extent to which descriptive sentences, taken from the web, can be characterised by some regularities in their lexicalised dependency paths. These regularities are assumed to identify definitions in web documents.

2 RELATED WORK

Question-Answering Systems (QAS) are usually assessed as a part of the QA track of the *Text REtrieval Conference* (TREC). QAS attempt to extract answers from a target collection of news documents: the AQUAINT corpus. In order to discover correct answers to definition questions, QAS in TREC extract nuggets from several external specific resources of descriptive information (e.g. online encyclopedia and dictionaries), and must then project them into the corpus. Generally speaking, this projection strategy relies on two main tasks:

1. Extract external resources containing entries corresponding to the *definiendum*.
2. Find overlaps between terms in definitions (within the target collection) and terms in the specific resources.

In order to extract sentences related to the *definiendum*, some approaches take advantage of ex-

Figueroa A. and Atkinson J.
USING DEPENDENCY PATHS FOR ANSWERING DEFINITION QUESTIONS ON THE WEB.
DOI: 10.5220/0001833806380645
In *Proceedings of the Fifth International Conference on Web Information Systems and Technologies (WEBIST 2009)*, page
ISBN: 978-989-8111-81-4

ternal resources (e.g., *WordNet*), online specific resources (e.g., Wikipedia) and Web snippets (Cui et al., 2004). These are then used to learn frequencies of words that correlate to the *definiendum*. Experiments showed that definitional websites greatly improved the performance by leaving few unanswered questions: Wikipedia covered 34 out of the 50 TREC–2003 definition queries, whereas *biography.com* covered 23 out of 30 questions regarding people, all together providing answers to 42 queries. These correlated words were then used to form a centroid vector so that sentences can be ranked according to the cosine distance to this vector.

One advantage of this kind of model is that this ranks candidate answers according to the degree in which their respective words characterise the *definiendum*, which is the principle known as the Distributional Hypothesis (Harris, 1954). However, the approach fails to capture sentences containing the correct answers with words having low correlation with the *definiendum*. This in turn causes a less diverse output, thus decreasing the coverage. In addition, taking into account only semantic relationships is insufficient for ranking answer candidates: the co-occurrence of the *definiendum* with learnt words across candidate sentences does not necessarily guarantee that they are syntactically dependent. An example of this can be seen in the following sentence regarding "*British Prime Minister Gordon Brown*":

`According to the Iraqi Prime Minister's office, Gordon Brown was reluctant to signal the withdrawal of British troops.`

In order to deal with this issue, (Chen et al., 2006) introduced a method that extended centroid vectors to include word dependencies which are learnt from the 350 most frequent stemmed co-occurring terms taken from the best 500 snippets retrieved by *Google*. These snippets were fetched by expanding the original query by a set of highly co-occurring terms. These terms co-occur with the *definiendum* in sentences obtained by submitting the original query plus some task specific clues, (e.g.,"*biography*"). Nevertheless, having a threshold of 350 frequent words is more suitable for technical or accurate *definiendums* (i.e., "*SchadenFreude*"), than for ambiguous or biographical *definiendums* (i.e., "*Alexander Hamilton*") which need more words to describe many writings of their several facets. These 350 words are then used for building an ordered centroid vector by retaining their original order within the sentences. To illustrate this, consider the following example:

`Today's Highlight in History: On November 14, 1900, Aaron Copland, one of America's leading 20th century composers, was born in New York City.` $\Longrightarrow$

The corresponding ordered centroid vectors become the words "`November 14 1900 Aaron Copland America composer born New York City.`" which are then used for training statistical language models and ranking candidate answers. Bi-gram language models were observed to significantly improve the quality of the extracted answers. Furthermore, Bi-term language models yield better results, showing that flexibility and relative position of lexical terms capture shallow information about their syntactic relation (Belkin and Goldsmith, 2002).

While *Google* provides facilities to search for definitions on the web, other approaches (Cui et al., 2004; Chen et al., 2006) are aimed at discovering answers from the AQUAINT corpus. Every time a user enters "define:*definiendum*", the search engine returns a set of glossaries containing definitions of the term. Although it is unknown how *Google* gathers these glossaries: which strategies are involved? What is manual or automatic? (Xu et al., 2005) observed that these glossaries seem to have some common properties: pages are titled with task-specific clues including "*glossary*" and "*dictionary*", the terms in the page are alphabetically sorted and presented with the same style, for instance, italics and bold print. Bearing this in mind, this method yields wider coverage. Nevertheless, succinct definitions taken from different glossaries are very likely to convey redundant information, while at the same time, new concepts are rarely found in glossaries, but in web-sites such as blogs or forums. All things considered, QAS are forced to search for additional information across several documents in order to satisfactorily provide an answer for the user.

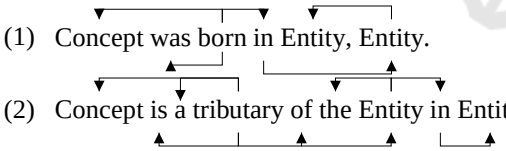
3 ANSWERING QUESTIONS BY LEARNING DEFINITION DEPENDENCY PATHS

We propose a model which is capable of answering definition questions by making use of contextual language models when ranking candidate sentences. For this, dependency paths are hypothesised to provide the balance between lexical semantic and syntactic information required to characterise definitions. In particular, this work claims that many descriptive sentences can be identified by means of contextual lexicalised dependency paths. To illustrate this, consider the following phrase:

`CONCEPT is a * politician and leader of the *`

Human readers would quickly notice that the sentence is a definition of a politician, despite the missing concept and words. This is made possible due to the existence of two dependency paths `ROOT`→`is`→`politician`, and `politician`→`leader`→`of`. The former acts as a *context indicator* indicating the type of *definiendum* being described, whereas the latter yields content that is very likely to be found across descriptions of this particular context indicator (*politician*). A key difference from the vast majority of TREC systems is that the inference is drawn by using contextual information conveyed by several descriptions of *politicians*, instead of using additional sources that provide information about a particular *definiendum* (e. g., "*Gordon Brown*").

In our approach, *context indicators* and their corresponding dependency paths are learnt from abstracts provided by *Wikipedia*. Specifically, contextual n-gram language models are constructed on top of these contextual dependency paths in order to recognise sentences conveying definitions. Unlike other QA systems (Hildebrandt et al., 2004), definition patterns are applied at the surface level (Soubbotin, 2001) and key named entities are identified using named-entity recognizers (NER)[1]. Preprocessed sentences are then parsed by using a lexicalised dependency parser[2], in which obtained lexical trees are used for building a treebank of lexicalised definition sentences. As an example, the following trees extracted from the treebank represent two highly-frequent definition sentences:



The treebank contains trees for $1,900,642$ different sentences in which each entity is replaced with a placeholder. This placeholder allows us to reduce the sparseness of the data and to obtain more reliable frequency counts. For the same reason, we did not consider different categories of entities and capitalised adjectives were mapped to the same placeholder.

From the sentences in the treebank, our method identifies potential *Context Indicators*. These involve words that signal what is being defined or what type of descriptive information is being expressed. Context indicators are recognised by walking through the dependency tree starting from the root node. Since only sentences matching definition patterns are taken into account, there are some regularities that are helpful to find the respective context indicator. Occasionally, the root node itself is a context indicator. However, whenever the root node is a word contained in the surface patterns (e.g. *is*, *was* and *are*), the method walks down the hierarchy. In the case that the root has several children, the first child (different from the concept) is interpreted as the context indicator. Note that the method must sometimes go down one more level in the tree depending of the expression holding the relationship between nodes (i.e., "*part/kind/sort/type/class/first of*"). Furthermore, the used lexical parser outputs trees that meet the projection constrain, hence the order of the sentence is preserved. Overall, $45,698$ different context indicators were obtained during parsing. Table 1 shows the most frequent indicators acquired with our method, where $P(c_s)$ is the probability of finding a sentence triggered by the context indicator c_s within the treebank.

Table 1: Some Interesting *Context Indicators*.

Indicator	$P(c_s)*10^4$	Indicator	$P(c_s)*10^4$
born	1,5034	company	1,32814
album	1,46046	game	1,31932
member	1,45059	organization	1,31836
player	1,38362	band	1,31794
film	1,37389	song	1,3162
town	1,37243	author	1,31601
school	1,35213	term	1,31402
village	1,35021	series	1,31388
station	1,34465	politician	1,30075
son	1,33464	group	1,29767

Next, candidate sentences are grouped according to the obtained context indicators. Consequently, highly-frequent directed dependency paths within a particular context are hypothesised to significantly characterise the meaning when describing an instance of the corresponding context indicator. This is strongly based on the extended distributional hypothesis (Lin and Pantel, 2001) which states that if two paths tend to occur in similar contexts, their meanings tend to be similar. In addition, the relationship between two entities in a sentence is almost exclusively concentrated in the shortest path between the two entities of the undirected version of the dependency graph (Bunescu and Mooney, 2005). Hence, one entity can be interpreted as the *definiendum*, and the other can be any entity within the sentence. Therefore, paths linking a particular type of *definiendum* with a class of entity relevant to its type will be highly frequent in the context (e. g., $politician \rightarrow leader \rightarrow of \rightarrow ENTITY$).

For each context, all directed paths containing two to five nodes are extracted. Longer paths are not taken into consideration as they are likely to indicate weaker

[1] http://nlp.stanford.edu/software/CRF-NER.shtml
[2] http://nlp.stanford.edu/software/lex-parser.shtml

syntactic/semantic relations. Directions are mainly considered, because relevant syntactical information regarding word order is missed when going up the dependency tree. Otherwise, undirected graphs would lead to a significant increase in the number of paths as it might go from any node to any other node. Some illustrative directed paths obtained from the treebank for the context indicator *politician* are shown below:

politician→activist→leader→of
politician→affiliated→with→Entity
politician→considered→ally→of→Entity
politician→head→of→state→of
politician→leader→of→opposition
politician→member→of→chamber
president→house→of→Entity
proclaimed→on→Entity

From the obtained dependency paths, an n-gram statistical language model ($n = 5$) for each context was built in order to estimate the most relevant dependency path. The probability of a dependency path $\vec{dp}$ in a context c_s is defined by the likely dependency links that compose the path in the context c_s, with each link probability conditional on the last $n-1$ linked words:

$$p(\vec{dp} \mid c_s) \approx \prod_{i=1}^{l} (w_i \mid c_s, w_{i-n+1}^{i-1}) \qquad (1)$$

Where $p(w_i \mid c_s, w_{i-n+1}^{i-1})$ is the probability of word w_i is linked with the previous word w_{i-1} after seeing the dependency path $w_{i-n+1} \ldots w_{i-1}$. In simple words, the likelihood that w_i is a dependent node of w_{i-1}, and w_{i-2} is the head of w_{i-1}, and so forth (see example in figure 1).

The probabilities $p(w_i \mid c_s, w_{i-n+1}^{i-1})$ are usually computed by computing the *Maximum Likelihood Estimate*: $\frac{count(c_s, w_{i-n+1}^{i})}{count(c_s, w_{i-n+1}^{i-1})}$. However, in our case, the word count $c(c_s, w_{i-n+1}^{i})$ can frequently be greater than $c(c_s, w_{i-n+1}^{i-1})$. For example, in the following definition sentence:

(4) Concept is a band formed in Entity in Entity.

The word "*formed*" is the head of two "*in*", hence the denominator of $p(w_i \mid c_s, w_{i-n+1}^{i-1})$ is the number of times w_{i-1} is the head of a word (after seeing w_{i-n+1}^{i-1}). The obtained 5-gram language model is smoothed by interpolating with shorter dependency paths (Zhai and Lafferty, 2004; Chen and Goodman, 1996) as follows:

$$P_{interp}(w_i \mid c_s, w_{i-n+1}^{i-1}) = \lambda_{c_s, w_{i-n+1}^{i-1}} P(w_i \mid c_s, w_{i-n+1}^{i-1}) + (1 - \lambda_{c_s, w_{i-n+1}^{i-1}}) P_{interp}(w_i \mid c_s, w_{i-n+2}^{i-1})$$

The probability of a path is accordingly computed as shown in equation 1 by accounting for the recursive interpolated probabilities instead of raw *P*s. Note also that $\lambda_{c_s, w_{i-n+1}^{i-1}}$ is computed for each context c_s (Chen and Goodman, 1996). A sentence S is ranked according to its likelihood of being a definition as follows:

$$rank(S) = p(c_s) \sum_{\forall \vec{dp} \in S} p(\vec{dp} \mid c_s) \qquad (2)$$

In order to avoid counting redundant dependency paths, only paths ending with a **leave node** are taken into account, whereas **duplicate** paths are discarded.

3.1 Extracting Candidate Answers

Our model extracts answers to definition questions from Web snippets. Thus, sentences matching definition patterns at the surface level are pre-processed[3] and parsed in order to get the corresponding lexicalised dependency trees. Given a set of test sentences/dependency trees extracted from the snippets, our approach discovers answers to definition question by iteratively selecting sentences.

Algorithm 1: ANSWER EXTRACTOR.

```
1:  ϕ = ∅
2:  indHis = getContextIndicatorsHistogram(T)
3:  for highest to lowest frequent ι ∈ indHis do
4:      while true do
5:          next = null
6:          for all t_t ∈ T do
7:              if ind(t_i)==ι then
8:                  rank = rank(t_i,ϕ);
9:                  if next == null or rank > rank(next) then
10:                     next = t_i
11:                 end if
12:             end if
13:         end for
14:         if next == null or rank(next) ≤ 0.005 then
15:             break;
16:         end if
17:         print next
18:         addPaths(next,ϕ)
19:     end while
20: end for
```

The general strategy for this iterative selection task can be seen in algorithm 1 whose input is the set of dependency path (T). This first initialises a set ϕ, which keeps the dependency paths belonging to previously selected sentences (line 1). Next, context indicators for each candidate sentence are extracted so as to build an histogram *indHist* (line 2). Since highly-frequent context indicators indicate more reliable potential senses, the method favours candidate sentences

[3] http://www.comp.nus.edu.sg/ qiul/NLPTools/JavaRAP.html

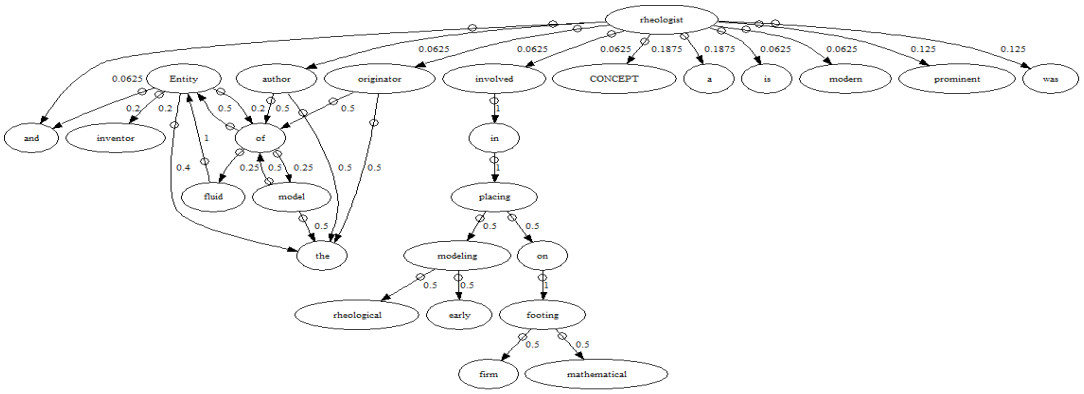


Figure 1: Bigram raw probabilities for c_s = "*rheologist*"

according to their context indicator frequencies (line 3). Sentences matching the current context indicator are ranked according to equation 2 (lines 7 and 8). However, only paths $\vec{dp}$ in $t_i - \phi$ are taken into consideration while computing equation 2. Sentences are thus ranked according to their novel paths respecting to previously selected sentences, while at the same time, sentences carrying redundant information, decrease their ranking value systematically. The highest ranked sentences are selected after each iteration (lines 9-11), and their corresponding dependency paths are added to ϕ (line 18). If the highest ranked sentence meets the halting conditions, the extraction task finishes. Halting conditions must ensure that no more sentences are left and that there are no more candidate sentences containing strong evidence of carrying novel descriptive content.

In this answer extraction approach, candidate sentences become less relevant as long as their overlap with all previously selected sentences becomes larger. Unlike other approaches (Hildebrandt et al., 2004; Chen et al., 2006) which control the overlap at the word level, our basic unit is a dependency path, that is, a group of related words. Thus, our method favours novel content, while at the same time, making a global check of the redundant content. Also, the use of paths instead of words as units ensures that different instances of a word, that contribute with different descriptive content, will be accounted accordingly.

4 EXPERIMENTS AND RESULTS

In order to assess our initial hypothesis, a prototype of our model was built and assessed by using 189 definition questions taken from TREC 2003-2004-2005 tracks. Since our model extracts answers from the web, these TREC datasets were only used as reference question sets. For each question, the best 300 web snippets were retrieved by using MSN Search and manually inspected in order to create a gold standard. Accordingly, the search strategy described in (Figueroa and Neumann, 2007) was utilised for fetching these web snippets. It is important to note that there was no descriptive information for 11 questions corresponding to the TREC 2005 data set. For experiment purposes, two baselines were implemented, and the three systems were provided with the same set of snippets. As different F-scores get involved, the evaluation stuck to the most recent standard by using uniform weights for the nuggets (Lin and Demner-Fushman, 2006).

Table 2: Some associations with w_*="politician".

$\vec{w} =< w_1, w_2 >$	$I_2(\vec{w})$	$\vec{w} =< w_1, w_2, w_3 >$	$I_3(\vec{w})$
$< w_*$, diplomat$>$	7.06	$<$ a, w_*, currently$>$	7.41
$< w_*$, currently$>$	4.33	$< w_*$, who, currently$>$	7.14
$< w_*$, opposition$>$	4.15	$<$ a, w_*, conservative$>$	2.93
$< w_*$, conservative$>$	3.44	$<$ a, w_*, opposition$>$	2.71

While our model was almost exclusively built upon dependency paths, the first baseline (BASELINE I) was constructed on top of word association norms (Church and Hanks, 1990). These norms were computed from the same set of 1,900,642 preprocessed sentences taken from abstracts of *Wikipedia*. These norms comprise pairs I_2 and triplets I_3 of ordered words as sketched in table 2. Next, the baseline chooses sentences according to algorithm 1, but making allowances for these norms instead of dependency paths. Sentences are then ranked according to the sum of the matching norms which are normalised by dividing them by the highest value. This baseline does not account for context indicators, so that every sentence is assumed to have the same context indicator.

These word association norms compare the probability of observing w_2 followed by w_1 within a fixed window of ten words with the probabilities of observ-

Table 3: Sample output sentences regarding "*Andrea Bocelli*".

Born in Lajatico, Italy, tenor singer Andrea Bocelli became blind at the age of 12 after a sports injury, and later studied law, but decided on a singing career.
Born on September 22, 1958, Andrea Bocelli is an Italian operatic pop tenor and a classical crossover singer who has also performed in operas.
Andrea Bocelli is a world class Italian tenor and classical crossover artist.
Andrea Bocelli was born 22 September 1958 in Lajatico in Tuscany, Italy.
Andrea Bocelli is an Italian singer and songwriter from Italy.
Andrea Bocelli has been the world's most successful classical artist for the past five years, selling 45 million albums.
Andrea Bocelli is an Italian singer who is famous throughout the world.
Andrea Bocelli has been a bestselling Italian artist, with over 12 million albums sold in Europe since the debut of his self-titled CD in 1993.
Andrea bocelli was born in italy in 1958, and began to sing as a child.
Andrea Bocelli, the world's most popular tenor (and Best selling) and pop sensation as well, has recorded The Best of Andrea.

ing w_1 and w_2 independently. Since the major difference between both systems is the use of these norms instead of dependency paths, the baseline provides a good starting point for measuring the contribution of our dependency-based models.

A second baseline (BASELINE II) makes allowances for the centroid vector (Cui et al., 2004). Sentences are thus selected by using algorithm 1, but ranked according to their similarity with this vector. Since our strategies are aimed specifically at being independent of looking specific entries in external resources, this centroid vector was learnt from all retrieved sentences containing the *definiendum*. These sentences include those which did not match definition patterns. In the same way, all these sentences are seen as candidates later, and hence, contrary to the two other systems, this baseline can identify descriptions from sentences that do not match definition patterns.

Table 4: Results for TREC question sets.

	TREC 2003	TREC 2004	TREC 2005
Size	50	64	(64)/75
	BASELINE I		
Recall	0.52±0.18	0.47±0.13	**0.49**±0.20
Precision	0.27±0.14	0.26±0.11	**0.29**±0.24
F(3) Score	0.46±0.14	0.42±0.11	**0.43**±0.17
	BASELINE II		
Recall	0.27±0.23	0.27±0.16	0.24±0.17
Precision	0.20±0.19	0.20±0.19	0.18±0.23
F(3) Score	0.24±0.18	0.25±0.15	0.22±0.16
	OUR SYSTEM		
Recall	**0.57**±0.17	**0.50**±0.18	0.42±0.22
Precision	**0.39**±0.21	**0.40**±0.19	**0.29**±0.21
F(3) Score	**0.53**±0.15	**0.47**±0.17	0.38±0.19

The main results obtained can be seen at table 4. Overall, our model outperformed BASELINE I in 5.22% and 11.90% for the TREC 2003 and 2004 datasets, respectively. These increases are mainly due to *definiendums* such as "*Allen Iverson*" and "*Fred Durst*", while the performance worsened for "*Rhodes Scholars*" and "*Albert Ghiorso*". In terms of the standard deviation, the increase in dispersion may be due to the fact that our language models are independently built for each context indicator, whereas the association norms are computed as if every sentence belonged to the same context.

Consequently, due to the limited coverage provided by *Wikipedia*, some contexts were obtained with few samples, causing some low $p(c_s)$ values. Hence, our method may miss many nuggets whenever a low-frequent context indicator is the predominant potential sense. This can be addressed by taking abstracts into consideration in newer and older versions of *Wikipedia*. In addition, collecting short definitions from glossaries across documents on the Web can also be beneficial. These glossaries can be automatically extracted by identifiying regularities in their lay-outs: tables, entries alphabetically sorted, and bold print.

In general, our approach identified more nuggets than both baselines, and as we hypothesised, these pieces of information were characterised by regularities in their contextual dependency paths. In the case of TREC 2003, the average recall increased from 0.52 to 0.57 (9.6%), whereas it improved 6.4% for the TREC 2004 dataset. An illustrative output produced by our system can be seen in table 3. On the other hand, *definiendums* such as "*Jennifer Capriatti*" and "*Heaven's Gate*" resulted in significant recall improvements, whereas "*Abercrombie and Fitch*" and "*Chester Nimitz*" went into steep declines.

Furthermore, our approach achieved higher precision for two datasets. In the case of the TREC 2003, the increase was 44.44%, whereas it was 53.84% for the TREC 2004 question set. Our model was capable of filtering out a larger amount of sentences that did not yield descriptions. As a result, linguistic infor-

Table 5: Sample containing issues regarding performance.

NOTES: Presents an examination of the Teapot Dome scandal that took place during the presidency of Warren G. Harding in the 1920s.
Teapot Dome Scandal was a scandal that occurred during the Harding Administration.
This article focuses on the Teapot Dome scandal, which took place during the administration of U. S. President Warren G. Harding.
The Teapot Dome Scandal was a scandal under the administration of President Warren Harding which involved critical government oil fields.
Teapot Dome Scandal cartoon The Teapot Dome Scandal was an oil reserve scandal during the 1920s.
The Teapot Dome scandal became a parlor issue in the presidential election of 1924 but, as the investigation had only just started earlier that year, neither party could claim full.
The Teapot Dome scandal was a victory for neither political party in the 1920's, it did become a major issue in the presidential election of 1924, but neither party could claim full.

mation provided by lexicalised dependency paths was observed to be particularly important to increase the accuracy of the answers.

As for TREC 2005, our system finished with a lower recall and F(3)-Score. A closer look at the achieved results shows that our system increased the performance in 37 out of the 64 questions, while in 24 cases the performance was reduced. A key point here was that in six of these 24 cases, our system obtained a recall of zero. These zero recall values cause F(3)-Scores equal to zero, and eventually, bringing about a considerable decline in the average F(3)-Score. Three of these six questions correspond to the *definiendums*: "*Rose Crumb*" and "*1980 Mount St. Helens eruption*" as well as "*Crash of EgyptAir Flight 990*".

Two common issues for these six scenarios are: (a) few nuggets were found within the fetched snippets, and (b) these nuggets had a low frequency. Hence, whenever our system missed any or all of them, the performance was detrimental. This situation becomes graver whenever the nuggets are in contexts that are very unlikely to be in our models. To measure the impact of these six cases, the average F(3)-Score was compared by accounting solely for the other 58 questions: 0.43 for our system, and 0.41 for the first baseline. In order to investigate the overall precision of the approaches, the *Mean Average Precision* (MAP) of the top one and five ranked sentences (accounting for "*Precision at one and five*", respectively) was computed as seen in table 6.

Obtained MAP scores show that using our contextual models effectively contributes to improving the ranking of the sentences. Essentially, they help to bias the ranking in favour of descriptive sentences that: (a) have some lexico-syntactic similaritieswith sentences in Wikipedia abstracts, and more importantly (b) correspond to predominant and hence, more reliable, potential senses. One important finding is that our system did not only outperform the other two strategies, but it also finished with a high precision in ranking, containing a valid definition at the top in about 80% of the cases.

Table 6: *Mean Average Precision* (MAP).

	BASELINE I	BASELINE II	OUR SYSTEM
		TREC 2003	
MAP-1	0.64	0.16	**0.82**
MAP-5	0.64	0.21	**0.82**
		TREC 2004	
MAP-1	0.66	0.27	**0.88**
MAP-5	0.62	0.25	**0.82**
		TREC 2005	
MAP-1	0.77	0.18	**0.79**
MAP-5	0.70	0.24	**0.77**

Unlike TREC systems, our system was evaluated by using sentences extracted from the web. While we took advantage of sophisticated search engines, these are not optimised for QA tasks. In addition, many TREC systems make use of off-line processing on the AQUAINT corpus in order to boost the performance (Hildebrandt et al., 2004) so that when ranking, they use extra features such as entities, which are also useful in recognising definitions. Instead, our approach achieves a competitive performance, when ranking by accounting almost exclusively for the lexical syntactic and semantic similarities to previously known definitions that describe another instances of the same kind of *definiendum*. Note also that sense taggers might be applied to accurately recognise entities. It is somehow traded off by ranking definitions based on dependency paths which require less time to compute.

The additional knowledge used when ranking is the frequency of the context indicators, which assists the model in ranking frequent potential senses, and more reliable sentences. Our experiments thus showed that dependency paths provide key lexico-semantic and syntactic information that characterises definitions at the sentence level.

The use of relations between a group of words instead of isolated terms for ranking sentences also ensures a certain level of grammaticality in the candidate answers. Since web snippets are often truncated

by search engines, relations allow us to select truncated sentences that are more likely to convey a complete idea than others. On the other hand, two different dependency paths can yield the same descriptive information, causing an increment in redundancy (see "*Teapot Dome Scandal*" in table 5).

5 CONCLUSIONS

Experiments using our model showed that lexicalised dependency paths serve as salient indicators for the presence of definitions in natural language texts. The model also outperformed some baseline built from previous TREC dataset showing the promise of the approach by using context information. This suggests that learning contextual entities may improve the performance.

Further strategies to detect redundancy can be developed by recognising similar dependency paths (Chiu et al., 2007). This provides a key advantage of using dependency paths for answering definition questions. Context indicators defined for our approach can also be used to cluster definition sentences according to their senses.

ACKNOWLEDGEMENTS

This work was partially supported by a research grant from the German Federal Ministry of Education, Science, Research and Technology (BMBF) to the DFKI project HyLaP (FKZ: 01 IW F02) and the EC- funded project QALL-ME - FP6 IST-033860 (http://qallme.fbk.eu). Additionally, this research was partially sponsored by the National Council for Scientific and Technological Research (FONDECYT, Chile) under grant number 1070714.

REFERENCES

Belkin, M. and Goldsmith, J. (2002). Using eigenvectors of the bigram graph to infer gram- matical features and categories. In *Proceedings of the Morphology/Phonology Learning Workshop of ACL-02*.

Bunescu, R. and Mooney, R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of HLT/EMNLP*.

Chen, S. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the ACL*, pages 310–318.

Chen, Y., Zhon, M., and Wang, S. (2006). Reranking answers for definitional qa using language modeling. In *Coling/ACL-2006*, pages 1081–1088.

Chiu, A., Poupart, P., and DiMarco, C. (2007). Generating lexical analogies using dependency relations. In *Proceedings of the 2007 Joint Conference on EMNLP and Computational Natural Language Learning*, pages 561–570.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. In *Computational Linguistics, Vol. 16, No. 1*, pages 22–29.

Cui, T., Kan, M., and Xiao, J. (2004). A comparative study on sentence retrieval for definitional question answering. In *SIGIR Workshop on Information Retrieval for Question Answering (IR4QA)*, pages 383–390.

Figueroa, A. and Neumann, G. (2007). A Multilingual Framework for Searching Definitions on Web Snippets. In *KI*, pages 144–159.

Harris, Z. (1954). Distributional structure. In *Distributional structure. Word, 10(23)*, pages 146–162.

Hildebrandt, W., Katz, B., and Lin, J. (2004). Answering definition questions using multiple knowledge sources. In *Proceedings of HLT-NAACL*, pages 49–56.

Joho, H. and Sanderson, M. (2000). Retrieving descriptive phrases from large amounts of free text. In *9th ACM conference on Information and Knowledge Management*, pages 180–186.

Joho, H. and Sanderson, M. (2001). Large scale testing of a descriptive phrase finder. In *1st Human Language Technology Conference*, pages 219–221.

Lin, D. and Pantel, P. (2001). Discovery of inference rules for question answering. In *Journal of Natural Language Engineering, Volume 7*, pages 343–360.

Lin, J. and Demner-Fushman, D. (2006). Will pyramids built of nuggets topple over? In *Proceedings of the main conference on HTL/NAACL*, pages 383–390.

Rose, D. E. and Levinson, D. (2004). Understanding user goals in web search. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 13–19.

Soubbotin, M. M. (2001). Patterns of potential answer expressions as clues to the right answers. In *Proceedings of the TREC-10 Conference*.

Xu, J., Cao, Y., Li, H., and Zhao, M. (2005). Ranking definitions with supervised learning methods. In *WWW2005*, pages 811–819.

Zhai, C. and Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. In *ACM Transactions on Information Systems, Vol. 22, No. 2*, pages 179–214.