Enriched syntax-based meaning representation for answer extraction

Grzegorz Chrupała
Saarland University
Saarbrücken, Germany
gchrupala@lsv.uni-saarland.de

Georgiana Dinu
Saarland University
Saarbrücken, Germany
dinu@coli.uni-sb.de

Benjamin Roth
Saarland University
Saarbrücken, Germany
beroth@coli.uni-sb.de

ABSTRACT

In Question Answering a major challenge is the fact that similar meaning is very often expressed with different surface realizations in questions and in sentences containing the answer. In this paper we propose an enriched syntax-based representation which helps deal with this widespread variability and provides a degree of generalization. We encode uncertainty about the syntactic structure of the question by using multiple alternative dependency parse trees. We then augment the question meaning representation by including multiple paraphrases of each dependency path, derived from distributional analysis of a large corpus.

1. INTRODUCTION

In Question Answering one of the major challenges is the fact that similar meaning is expressed with different surface realizations in questions and in sentences containing the answer. For example the question might be: *Who discovered the Mississippi river*, whereas the answer might be expressed as: *The Mississippi was found by Hernando de Soto*. In order to find the correct answer we need to provide some mechanism to generalize over the different ways in which the same concept can be expressed. In this paper we focus on the use of enriched syntactic representations which provide such means of generalization.

The DIRT [Lin and Pantel, 2001a] algorithm uses distributional statistics to acquire paraphrases over paths in dependency trees from a large corpus. The paraphrases that are acquired this way vary from simple syntactic rewritings to more complex lexical-syntactic variations.

Despite the fact that DIRT was developed with applications such as QA in mind [Lin and Pantel, 2001b], and even though it is a resource which is easy to acquire and relatively accurate, there has been rather little research on using it solve the variability problem in NLP in general and in QA in particular.

We implemented a DIRT-based paraphrasing component in combination with a syntactic representation consisting of a set of paths in a lexicalized parse forest.

In order to evaluate this idea, we built a basic question answering pipeline. The meaning of the question is represented as a set of syntactic dependency paths in a set of n-best dependency trees. The paths connect the question word to other words in the question. Sentences containing the potential answer are represented in a similar fashion.

We rank answer candidates by matching question and answer dependency paths. Such methods have been shown to be effective in a number of studies, e.g. [Punyakanok et al., 2004, Cui et al., 2005, Shen and Klakow, 2006]. Unlike many of these approaches, we avoid learning mappings between question paths and sentences paths in a supervised fashion from question-answer pairs. Rather we augment the set of dependency paths in the meaning representation of a question with a number of synonymous paths, as determined by the paraphrasing component.

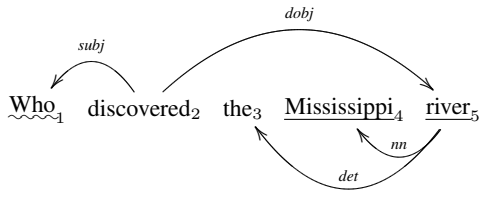
2. ANSWER EXTRACTION AND RANKING

In this section we describe the baseline answer extraction algorithm we use in all our experiments.

Candidate answer extraction.

We parse the question and its relevant sentences with a dependency parser. We then identify QUESTION WORDS and KEY WORDS in the question and the sentence parse trees. Question words are wh-words (*who, what, when, when, how, which*) and key words are common and proper nouns and numbers.

Given a dependency parse tree, a DEPENDENCY PATH is the sequence of nodes and labeled edges between two nodes. For questions, we extract a dependency path between any pair consisting of a question word and a key word. Consider the question *Who discovered the Mississippi river*. A possible parse tree is:



The following path will be extracted from this tree:

$$\underset{\sim}{\text{who}}_1 \xleftarrow{subj} \text{discover} \xrightarrow{dobj} \text{river} \xrightarrow{nn} \underline{\text{Mississippi}}_4$$

Similarly, for sentences relevant to the question, we extract the paths between pairs of key words. For example the sentence *The Mississippi river was discovered by Hernando de Soto in 1541* contains the following dependency path:

$$\underset{\sim}{\text{Soto}}_9 \xleftarrow{pobj} \text{by} \xleftarrow{prep} \text{discover} \xrightarrow{dobj} \text{river} \xrightarrow{nn} \underline{\text{Mississippi}}_2$$

Note that our paths are lexicalized, i.e. they include words and not

only syntactic relations between words.

A sentence key word may match a key word in the question (ANCHOR, underlined with a single line). We treat the word at the other end of the path (which corresponds to the question word) as an ANSWER CANDIDATE. In the above example the answer candidate is "Soto". We assume that an answer candidate is likely to be the correct answer if it occurs in a pair where the two paths are maximally similar.

Candidate answer ranking.

We collect all the paths in a question connecting the question word and a particular anchor and train an bigram language model smoothed with absolute discounting [Zhai and Lafferty, 2004].

We score an answer candidate found in a sentence based on the perplexity of the language model trained on the path set corresponding to the matching anchor word in the question. We use an adjusted perplexity function so that it gives some preference to shorter paths:

$$PP_Q(\pi) = 2^{-\frac{1}{N+d}\sum_{i=1}^{N}\log_2 P_Q(\pi_i|\pi_{i-1})} \quad (1)$$

where π is the sentence path of length N, d is the "upcount" parameter and $P_Q(\pi_i|\pi_{i-1})$ is the probability of the i^{th} element of the path given the previous one, according to the language model trained on the question paths.

Next we generate candidate word strings corresponding to answer candidates (which are just word indices). We implement a simple POS tag-based heuristic: if we find the fragment $\mathit{the}^1_D\ \mathit{Spanish}^2_{JJ}\ \mathit{explorer}^3_{NN}\ \mathit{Hernando}^4_{NNP}\ \mathit{de}^5_{NNP}\ \mathit{Soto}^6_{NNP}$ then the following strings will correspond to the answer candidate with index 6: "Soto" "Hernando de Soto" and "explorer Hernando de Soto". Since all of them will be have the same perplexity score, we give preference to certain POS sequences based on the question word; e.g. for "when" we prefer POS sequences containing the number tag "CD".

3. N-BEST PARSING AND PARAPHRASING

The previous section describes our basic answer extraction system. In this section we discuss two extensions in order to deal with variation in expressing the same meaning.

N-best parsing.

Due to limitations of currently available parsing technology, an automatically obtained best parse tree for a naturally occurring sentence has a high probability of differing in at least one way from the analysis that a human expert would assign to it. The problem is exacerbated in question answering compared to other NLP applications by the fact that the most widely used training resource for English parsing, the Penn Treebank, contains a very small number of questions. In order to overcome this problem we represent a question by the set of dependency paths which are extracted not just from 1 best parse, but from n-best parses.

Paraphrasing.

The second major extension to the basic answer extraction approach described in section 2 is dependency-path-level paraphrasing. The question and sentence meaning representation as a set of syntactic dependency paths, as described so far, is not general enough: it encodes particular choices of lexical items and syntactic constructions.

We abstract away from these particulars by using unsupervised dependency-path-level paraphrasing. This simple approach stands in contrast with representations which explicitly posit and use abstract dependencies such as those of Frame Semantics [Fillmore, 1982, Baker et al., 1998].

$\mathrm{X} \xleftarrow{pobj}$ by $\xleftarrow{prep}$ discover $\xrightarrow{subjpass} \mathrm{Y}$
$\mathrm{X} \xleftarrow{subj}$ find $\xrightarrow{dobj} \mathrm{Y}$
$\mathrm{X} \xleftarrow{pobj}$ by $\xleftarrow{prep}$ find $\xrightarrow{subjpass} \mathrm{Y}$
$\mathrm{X} \xleftarrow{subj}$ unearth $\xrightarrow{dobj} \mathrm{Y}$
$\mathrm{X} \xleftarrow{subj}$ uncover $\xrightarrow{dobj} \mathrm{Y}$

Table 1: Top 5 paraphrases for X $\xleftarrow{subj}$ discover $\xrightarrow{dobj}$ Y

For each question we have associated an initial path set, obtained as described above. Following this, we paraphrase each path by matching it against the paraphrase collection. We allow paraphrase substitution by matching the whole path as well as any subpath (i.e. a valid subsequence of it).

The intuition behind our approach is that by including many realizations of the same meaning in the question representation, the chance that it will closely match a sentence containing the answer increases.

In order to account for the fact that paraphrases may introduce some noise, we rank paraphrased paths by their similarity to the original path, as returned by the DIRT algorithm). When training the language model on a path set, we weight the contribution of a path to the model by $\frac{1}{\log_2(\mathrm{R})}$ where R is the path's rank.

Acquiring paraphrases.

We acquire DIRT-style paraphrases using a 100-million-word portion of Gigaword [Graff et al., 2003], which we parse using the Stanford dependency parser [De Marneffe et al., 2006]. The maximum length of a path is two words, and we only consider noun-ending paths. In order to acquire paraphrases for a particular path, we compute the similarity between the target path and all other paths using the similarity measure due to [Lin and Pantel, 2001a].

$$\mathrm{sim}_{\mathrm{Lin}}(\mathbf{w},\mathbf{v}) = \frac{\sum_{i\in I(\mathbf{w})\cap I(\mathbf{v})}(w_i + v_i)}{\sum_{i\in I(\mathbf{w})} w_i + \sum_{i\in I(\mathbf{v})} v_i}$$

where $\mathbf{w}$ and $\mathbf{v}$ are vector representations of path distributions. The i^{th} value of a vector is the pointwise mutual information score between the path and i^{th} possible filler word. $I(\cdot)$ gives the indices of positive values in a vector. Two such scores are computed separately, one for the left and one for the right filler distribution, and their product is the path similarity for the whole path.

Table 1 shows the top 5 paraphrases for the path X $\xleftarrow{subj}$ discover $\xrightarrow{dobj}$ Y.

4. EVALUATION

Data.

For evaluation we used the QASP datasets [Kaisser and Lowe, 2008]. This resource was built using a subset of questions from TREC QA track datasets from years 2002-2006. For each TREC factoid question for which an answer could be found in the AQUAINT corpus, QASP provides the set of answer sentences, together with the corresponding answer string. [1]

[1]In QASP questions which in TREC where grouped into series about a certain topic are reformulated such that they can be answered in isolation – that is anaphoric references to the topic were replaced with the topic phrase itself.

# parses	1	5	10	15	20	25	30
MRR	43.8	45.3	45.3	45.3	45.4	45.5	45.4

Table 2: MRR on QASP 2002. Effect of varying the number of parses (using no paraphrasing)

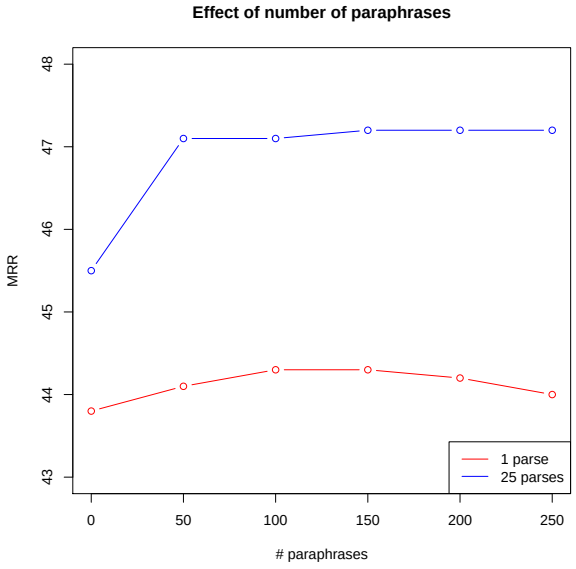


Figure 1: MRR on QASP 2002. Effect of varying the number of paraphrases, using 1-best parse and 25-best parses.

Configuration	MRR
Exp(1,0)	43.7
Exp(1,100)	44.3
Exp(25,0)	45.5
Exp(25,100)	47.2

Table 3: Results for four configurations on QASP 2002

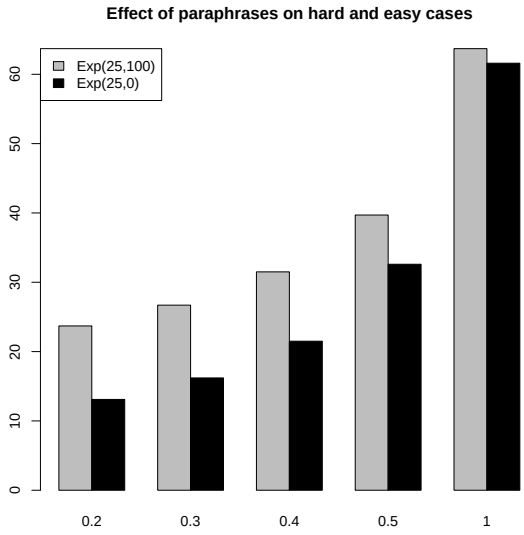


Figure 2: Effect of paraphrasing (Exp(25,0) vs. Exp(25,100)) on sentences of different difficulty. Each bin corresponds to the subset of sentences where the Exp(25,0) configuration had RR $\leq x$

It is worth emphasizing that we evaluate *at sentence-level*, that is we attempt to find the correct answer *for each question-answer pair in QASP*. This is in contrast to the typical TREC evaluations where systems were required to simply provide a single answer for each factoid question.

For dependency parsing we used the Stanford parser, running it in 1-best, lexicalized, mode for sentences and n-best mode for questions. As evaluation metric we used the mean reciprocal rank (MRR), i.e. the reciprocal of the rank of the correct answer averaged over all questions.

Results.

We used data from QASP 2002 for system development and we also provide below a detailed analysis on this portion of QASP.

We also report results on the whole QASP corpus in a cross-validation regime, where we use one year to find optimal parameters (i.e. number of parse trees and number of paraphrases), and evaluate on the remaining portion of the data (see Table 5).

We investigate the effects of expanding the question dependency paths by increasing: (i) the number of parses used and (ii) the number of paraphrases retrieved for each dependency path. As shown in Table 2, when varying the number of parses from 1 to 40 (using no paraphrasing) we found $\approx 2\%$ gain in MRR by adding 25-best parses. Most of the gain is obtain by adding the top 5 parses. Adding further ones brings only slight improvements. The scores start decreasing again at 30 parses.

Figure 1 summarizes the effect of varying the number of paraphrases used for expansion of paths. We show the results when the paraphrases are added on top of a 1-best parse, and of on top of a 25-best parses setting. We observe that adding n-best parses and increasing the number of paraphrases has a cumulative effect. While adding 100 paraphrases to 1 parse brings only 0.6% MRR gain, adding the same number to 25 parses brings 1.7% improvement. Similarly to expanding with multiple parses, most of the gain is obtained by adding a rather small number of paraphrases.

Table 3 shows the results on all the question-sentence pairs on QASP 2002. Exp(n,m) stands for a configuration using n parse trees and m paraphrases.

Paraphrasing component.

In order to get more insight into the effect of the paraphrasing component, we analyzed the development set in more detail.

When using the 25-best parses, adding 100 paraphrases increases the MRR from 45.5. to 47.2 overall. In Table 4 we consider only the subset of sentences where the correct answer appears in the candidate list (73.9% of all the sentences), since the paraphrasing component cannot correct this error. We further divide this subset into the portion in which paraphrases change the score and those in which they do not (Par+ vs. Par-).

We observe that the paraphrases affect the ranking of the candidate answers in only less than 20% of the total number of sentences. On this fragment of the data, they increased the score by 9%. This subset turns out to be much more difficult than the sentences where paraphrases did not change the score: 0.35 as opposed to 0.70 MRR. This indicates that paraphrasing is only helpful in a minority of cases, but those are the most difficult answers to ex-

Subset:	Par+	Par-
No. sents.	370	1109
Exp(25,0)	35.7	70.2
Exp(25,100)	44.4	70.2
No. sents.	116	1218
Exp(1,0)	23.1	65.6
Exp(1,100)	55.4	65.6

Table 4: MRR on QASP 2002. Effect of paraphrasing on 25-best parses 1-parse on the subset where the correct answer is in the candidate list. Par+/-: paraphrases do/do not change the baseline score

Model	MRR
Baseline	39.03
Enriched	40.64

Table 5: Cross-validation on QASP 2002-2006, tuning number of parse trees and number of paraphrases. The baseline uses 1 parse and no paraphrases

tract. To confirm this observation we computed the scores of the Exp(25,100) and Exp(25,0) configurations for those question sentence pairs where the no-paraphrase RR score is $\leq$ than a threshold x. Figure 2 shows the results for several values of the threshold. It is clear that paraphrases bring a lot of improvement on the more difficult sentences and that this effect weakens as the sentences become easier.

In the case where only one parse is used (Table 4), paraphrasing changes the scores in even fewer cases ($\approx$5% of the data), however the improvement they provide when they do is even higher than for 25 parses (over 30%).

The evaluation on the whole QASP dataset is summarized in Table 5. The improvement from the enhanced query representation is consistent but rather moderate. This is to be expected given our analysis above – the richer query representation is only able to affect the results in a minority of (especially difficult) cases. A challenge for future research is to determine how this limitation can be addressed.

5. CONCLUSION

Dependency path matching as well as paraphrasing components for passage retrieval or answer extraction have been used before in the literature. Unlike previous work, we focused on enhancing a baseline syntactic system solely with knowledge acquired in an unsupervised fashion. Our focus was to build a robust model, in which the paraphrasing is not used only when it provides an exact match between a question and candidate sentence (such as [Poon and Domingos, 2009]) but rather to allow question and answer-containing sentences to be brought closer.

Our proposed methods of enriching question meaning representation are simple to understand, easy to implement, and leverage well-understood techniques such as n-nest parsing, and unsupervised, distributional semantics.

The paraphrase component together with alternative parses for the question allow us to expand the representation of the question's meaning; this brings improvements in answer extraction MRR scores, particularly in sentences which are difficult for the basic syntactic method. Similarly to previous work such as [Dinu and Wang, 2009] we have noticed that paraphrases help only in a minority of cases, and we plan to investigate methods to improve that.

In this paper we have only evaluated sentence level scores. In the future we plan to investigate the ways in which the improvements at sentence level carry over to question level as well as to automatically retrieved sentences.

Acknowledgements

Grzegorz Chrupała was funded by the BMBF project NL-Search under contract number 01IS08020B. Georgiana Dinu was funded by the IRTG PhD program.

6. REFERENCES

[Baker et al., 1998] Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics Morristown, NJ, USA.

[Cui et al., 2005] Cui, H., Sun, R., Li, K., Kan, M., and Chua, T. (2005). Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 400–407. ACM New York, NY, USA.

[De Marneffe et al., 2006] De Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *LREC 2006*.

[Dinu and Wang, 2009] Dinu, G. and Wang, R. (2009). Inference rules and their application to recognizing textual entailment. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 211–219, Athens, Greece. Association for Computational Linguistics.

[Fillmore, 1982] Fillmore, C. J. (1982). Frame semantics. In *Cognitive linguistics: basic readings*, pages 373–400. Walter de Gruyter.

[Graff et al., 2003] Graff, D., Kong, J., Chen, K., and Maeda, K. (2003). English gigaword. Linguistic Data Consortium, Philadelphia.

[Kaisser and Lowe, 2008] Kaisser, M. and Lowe, J. B. (2008). Creating a Research Collection of Question Answer Sentence Pairs with Amazon's Mechanical Turk. In *Proc. of the Fifth International Conference on Language Resources and Evaluation (LREC-2008)*.

[Lin and Pantel, 2001a] Lin, D. and Pantel, P. (2001a). DIRT – Discovery of Inference Rules from Text. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD-01)*, San Francisco, CA.

[Lin and Pantel, 2001b] Lin, D. and Pantel, P. (2001b). Discovery of inference rules for question-answering. *Nat. Lang. Eng.*, 7(4):343–360.

[Poon and Domingos, 2009] Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore. Association for Computational Linguistics.

[Punyakanok et al., 2004] Punyakanok, V., Roth, D., and Yih, W. (2004). Mapping dependencies trees: An application to question answering. In *Proceedings of AI&Math 2004*. Citeseer.

[Shen and Klakow, 2006] Shen, D. and Klakow, D. (2006). Exploring correlation of dependency relation paths for answer extraction. In *Proceedings of the ACL*, volume 44, page 889.

[Zhai and Lafferty, 2004] Zhai, C. and Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):214.