

Feature	Description	Intuition
Sum IDF	The sum of the IDF scores for all terms in the original question and the paraphrase.	Paraphrases with more informative terms for the corpus at hand should be preferred.
Lengths	Number of query terms for each of the paraphrase and the original question.	We expect QA systems to prefer shorter paraphrases.
Cosine Distance	The distance between the vectors of both questions, IDF-weighted.	Certain paraphrases diverge too much from the original.
Answer Types	Whether answer types, as predicted by our question analyzer, are the same or overlap.	Choosing a paraphrase that does not share an answer type with the original question is risky.

Table 1: Our features, computed for each paraphrase by comparing it against the original question.

der not to lower the performance with respect to the original question. They are ordered by their relative contributions to the error rate reduction.

Paraphrase Selection Module. To select a paraphrase, we used JRip, the Java re-implementation of ripper (Cohen, 1996), a supervised rule learner in the Weka toolkit (Witten and Frank, 2000).

We initially formulated paraphrase selection as a three-way classification problem, with an attempt to label each paraphrase as being “worse,” the “same,” or “better” than the original question. Our objective was to **replace** the original question with a paraphrase labeled “better.” However, the priors for these classes are roughly 30% for “worse,” 65% for “same,” and 5% for “better”. Our empirical evidence shows that successfully pinpointing a “better” paraphrase improves, on average, the reciprocal rank for a question by 0.5, while erroneously picking a “worse” paraphrase results in a 0.75 decrease. That is to say, errors are 1.5 times more costly than successes (and five times more likely). This scenario strongly suggests that a high precision algorithm is critical for this component to be effective.

To increase precision, we took two steps. First, we trained a cascade of two binary classifiers. The first one classifies “worse” versus “same or better,” with a bias for “worse.” The second classifier has classes “worse or same” versus “better,” now with a bias towards “better.” The second step is to constrain the confidence of the classifier and only accept paraphrases where the second classifier has a 100% confidence. These steps are necessary to avoid decreasing performance with respect to the original question, as we will show in the next section.

4 Experimental Results

We trained the paraphrase selection module using our QA system, PIQUANT (Chu-Carroll et al., 2006). Our target corpus is the AQUAINT corpus, employed in the TREC QA track since 2002.

As for MT engines, we employed Babelfish and Google MT,⁸ rule-based systems developed by SYSTRAN and Google, respectively. We adopted different MT engines based on the hypothesis that differences in their translation rules will improve the effectiveness of the paraphrasing module.

To measure performance, we trained and tested by cross-validation over 712 questions from the TREC 9 and 10 datasets. We paraphrased the questions using the four possible combinations of MT engines with up to 11 intermediate languages, obtaining a total of 15,802 paraphrases. These questions were then fed to our system and evaluated per TREC answer key. We obtained a baseline MRR (top five answers) of 0.345 running over the original questions. An oracle run, in which the best paraphrase (or the original question) is always picked would yield a MRR of 0.48. This potential increase is substantial, taking into account that a 35% improvement separated the tenth participant from the second in TREC-9. Our three-fold cross validation using the features and algorithm described in Section 3 yielded a MRR of 0.347. Over 712 questions, it replaced 14, two of which improved performance, the rest stayed the same. On the other hand, random selection of paraphrases decreased performance to 0.156, clearly showing the importance of selecting a good paraphrase.

⁸<http://translate.google.com>

5 Related Work

Most of the work in QA and paraphrasing focused on folding paraphrasing knowledge into the question analyzer or the answer locator (Rinaldi et al., 2003; Tomuro, 2003). Our work, on the contrary, focuses on question paraphrasing as an external component, independent of the QA system architecture.

Some authors (Dumais et al., 2002; Echihabi et al., 2004) considered the query sent to a search engine as a “paraphrase” of the original natural language question. For instance, Echihabi et al. (2004) presented a large number of “reformulations” that transformed the query into assertions that could match the answers in text. Here we understand a question paraphrase as a reformulation that is itself a question, not a search engine query.

Other efforts in using paraphrasing for QA (Duclaye et al., 2003) focused on using the Web to obtain different verbalizations for a seed relation (e.g., Author/Book); however, they have yet to apply their learned paraphrases to QA.

Recently, there has been work on identifying paraphrases equivalence classes for log analysis (Hedstrom, 2005). Hedstrom used a vector model from Information Retrieval that inspired our cosine measure feature described in Section 3.

6 Conclusions

The work presented here makes contributions at three different levels. First, we have shown that potential impact of paraphrasing with respect to QA performance is significant. Replacing a question with a more felicitously worded question can potentially result in a 35% performance increase.

Second, we performed our experiments by tapping into a readily available paraphrase resource: MT engines. Our results speak of the usefulness of the approach in producing paraphrases. This technique of obtaining a large, although low quality, set of paraphrases can be easily employed by other NLP practitioners wishing to investigate the impact of paraphrasing on their own problems.

Third, we have shown that the task of selecting a better phrasing is amenable to learning, though more work is required to achieve its full potential. In that respect, the features and architecture discussed in Section 3 are a necessary first step in that direction.

In future work, we are interested in developing effective filtering techniques to reduce our candidate set to a small number of high precision paraphrases, in experimenting with state-of-the-art paraphrasers, and in using paraphrasing to improve the stability of the QA system.

Acknowledgments

The authors would like to thank Nelson Correa and Annie Ying for helpful discussions and comments. This work was supported in part by the Disruptive Technology Office (DTO)’s Advanced Question Answering for Intelligence (AQUAINT) Program under contract number H98230-04-C-1577.

References

- Regina Barzilay and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-EACL 2001)*, Toulouse, France, July.
- Jennifer Chu-Carroll, Pablo A. Duboue, John M. Prager, and Krzysztof Czuba. 2006. IBM’s piquant II in TREC 2005. In E. M. Voorhees and Lori P. Buckland, editors, *Proceedings of the Fourteenth Text REtrieval Conference Proceedings (TREC 2005)*, Gaithersburg, MD, USA.
- William Cohen. 1996. Learning trees and rules with set-valued features. In *Proceedings of the 14th joint American Association for Artificial Intelligence and IAAI Conference (AAAI/IAAI-96)*, pages 709–716. American Association for Artificial Intelligence.
- Mona Diab. 2000. An unsupervised method for word sense tagging using parallel corpora: A preliminary investigation. In *Special Interest Group in Lexical Semantics (SIGLEX) Workshop, Association for Computational Linguistics*, Hong Kong, China, October.
- Florence Duclaye, Francois Yvon, and Olivier Collin. 2003. Learning paraphrases to improve a question-answering system. In *EACL 2003, 11th Conference of the European Chapter of the Association for Computational Linguistics, Workshop in NLP for QA*, Budapest, Hungary, April.
- S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. 2002. Web question answering: is more always better? In *Proc. SIGIR ’02*, pages 291–298, New York, NY, USA. ACM Press.
- A. Echihabi, U.Hermjakob, E. Hovy, D. Marcu, E. Melz, and D. Ravichandran. 2004. Multiple-engine question answering in textmap. In *Proc. TREC 2003*.
- S. Harabagiu, D. Moldovan, C. Clark, M. Bowden, A. Hickl, and P. Wang. 2006. Employing two question answering systems. In *Proc. TREC 2005*.
- Anna Hedstrom. 2005. Question categorization for a question answering system using a vector space model. Master’s thesis, Department of Linguistics and Philology (Language Technology Programme) Uppsala University, Uppsala, Sweden.
- Fabio Rinaldi, James Dowdall, Kaarel Kaljurand, Michael Hess, and Diego Mollá. 2003. Exploiting paraphrases in a question answering system. In *Proceedings of the Second International Workshop on Paraphrasing*, pages 25–32, July.
- R. Sun, J. Jiang, Y.F. Tan, H. Cui, T.-S. Chua, and M.-Y. Kan. 2006. Using syntactic and semantic relation analysis in question answering. In *Proc. TREC 2005*.
- Noriko Tomuro. 2003. Interrogative reformulation patterns and acquisition of question paraphrases. In *Proceedings of the Second International Workshop on Paraphrasing*, pages 33–40, July.
- Ian H. Witten and Eibe Frank. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers.

State-of-the-art Question Answering (QA) systems are very sensitive to variations in the phrasing of an information need. Finding the preferred language for such a need is a valuable task. We investigate that claim by adopting a simple MT-based paraphrasing technique and evaluating QA system performance on paraphrased questions. We found a potential increase of 35% in MRR with respect to the original question. [View PDF](#). [Save to Library](#).