Automatic Feature Engineering for Italian Question Answering Systems

Antonio Uva[1] and Alessandro Moschitti[1,2]

[1] DISI, University of Trento, 38123 Povo (TN), Italy,
antonio.uva@unitn.it
[2] Qatar Computing Research Institute, 5825 Doha, Qatar
amoschitti@qf.org.qa

Abstract. In this paper, we propose automatic feature engineering for Italian QA systems. Our approach only requires a shallow syntactic representation of the questions and the answer passages. We apply Support Vector Machines using tree kernels to such trees for automatically generating relational syntactic patters, which significantly improve on BM25 retrieval models.

Keywords: Question Answering, Support Vector Machines, Kernel Methods

1 Introduction

Question Answering (QA) systems can be a valuable solution to the problem of information overload as they are effective tools for searching relevant information from unstructured text. QA systems differ from traditional search engines since they accept questions expressed in natural language. The major challenge in QA research is the design of effective answer search and extraction modules that can exploit the relations between the input question and the passages containing the answers.

Such relations or patterns can be used to decide if a retrieved passage does contain the correct answer. In past QA work, these patterns were mainly designed manually with consequently high engineering cost. However, machine learning has made this process much easier by enabling automatic pattern engineering.

In [1], we presented an automatic feature engineering approach based on support vectors machines using tree kernels for ranking answer passages. This approach consists of the following steps: (i) the set of possible candidate answers for all the input questions are retrieved by means of a search engine; (ii) each question is paired with all its candidate answer passages: positive pairs contain the correct answers and all the others are considered negative pairs; (iii) the pairs are represented with two syntactic trees: one for the question and the other for the candidate answer; and (iv) an SVM classifier is trained for ranking the answer passages represented as trees.

In this paper, we present a similar system that can rerank answer passages for factoid questions in Italian. This system is built on top of the Unstructured

Information Management Architecture (UIMA[3]) framework developed by IBM[4]. UIMA eases the task of assembling many text annotators together for performing different types of analysis over many text documents. These analytics are then used to encode questions and answers as linguistic structures and train the reranking module for our QA pipeline.

2 Learning to rank relevant documents

2.1 QA system

The QA system has a simple architecture: it takes in input a question and retrieves a list of candidates passages from the indexed dump of the Italian Wikipedia. Such list is then reranked by its relevancy respect the input question. The analysis of the question together with its candidate answers (e.g. PoS tags, Chunking, Named Entity, and many others) is performed by using the TextPro suite of NLP components for the Italian language. TextPro has been integrated as a stand-alone annotator in our UIMA pipeline. The annotations produced are used to build the tree representations of both questions and answers. The resulting question/answer tree pairs are used to train a classifier able to rank candidate passages according to their relevancy with the input question. The learned model is then used to improve the ordering of the answer passages provided by the search engine.

2.2 Answer reranking

Our goal is to rank text passages containing the correct answer higher in the list than irrelevant passages. For this purpose, we use the model we presented in [1], which is based on preference ranking [2]. This treats the reranking problem as a binary classification task, where each problem instance is a pair, (p_1, p_2), of question/answer pairs, i.e., $p_1 = (q, a_1)$ and $p_2 = (q, a_2)$. Positive training instances are pairs such that a_1 is a relevant passage and a_2 is an irrelevant passage otherwise (p_1, p_2) is considered a negative example.

These pairs can be used to train a binary classifier and build a reranking model. This model is later used at classification time for reranking the q/a pairs representing the test instances by simply using the classifier as a voter: a positive classification is a vote for a_1 whereas negative outcome is a vote for a_2. The more an answer receives votes the higher its rank will be.

2.3 Q/A pair representation

In our model, questions and answer passages are encoded as shallow syntactic trees we introduced in [3]. In each tree, the word lemmas constitute the terminal nodes and the Part-of-Speech(PoS) tags associated with each word constitute

[3] https://uima.apache.org/
[4] http://www.ibm.com/us/en/

the pre-terminal nodes. Also, the words are organized in constituents by adding an additional layer of chunk nodes. As the chunk of text spans several words, the chunk node is connected to the PoS nodes of its words. The sentence node is located at the top level and it is linked to the chunk nodes. A ROOT node is used to connect several sentence nodes. In addition, we encoded the relationships between the question and answer trees by means of a special tag REL.

The strategy used to establish the REL nodes is very simple: if two trees share the same terminal node (word lemma) then we mark both the node parent and grandparent with the REL tag. The REL approach leads to more accurate results [3].

3 Experiments

For our experiments we used factoid questions from the open-domain corpus TREC. More specifically, we used a subset of the questions from TREC 8, TREC 9, TREC 2000, TREC 2001 and TREC 2002 for a total of 1228 questions. An expert annotator translated the questions and answer gold keywords from English to Italian. The answers were supposed to be searched in the Italian Wikipedia. Thus, we train our reranker on such data.

Specifically, we split the Wikipedia corpus in paragraphs and considered each of them as a separate document to be indexed by an off-the-shelf search engine. After performing some text cleaning up, we were able to collect a total of 10 million documents. We used Lucene with the BM25 model for indexing and retrieval.

We trained our rerankers with the first 10 candidate answers retrieved by BM25 for each question of the train set. At test time, we retrieved a list of top 40 candidates for each test question and reranked them.

3.1 Metrics

In order to evaluate our systems we used the metrics most frequently used in QA: Precision at 1 documents (P@1) corresponds to the percentage of relevant documents ranked at position 1, Mean Reciprocal rank (MRR) and Mean Average Precision (MAP). The reported metrics are computed by conducting a 3-folds cross validation.

3.2 Results

The following table reports the performance of the reranking models trained using different strategies:

1. the baseline model using the score of the search engine (BM25)
2. the reranker model trained using only feature vectors[5] (V)
3. the reranker model trained using feature vectors and syntactic trees. (CH + V)

Models	MAP	MRR	P@1
BM25	0.18	23.11	15.22
V	0.21	26.85	18.23
CH + V	0.25	30.74	22.29

Table 1. The accuracy of the different ranking models

As can be seen from the results, the reranking model using structural representations yields an improvement of about 3 absolute points in MAP, MRR and P@1 when compared with the vector model and about 7 absolute points when compared with the baseline model. It is interesting to note that we did not operate any adjustment of the tree kernel model, we simply build an Italian pipeline and trained our models.

4 Conclusions

In this paper we showed an approach to QA requiring no manual feature engineering. Its main characteristic is the use of tree kernels for exploiting syntactic representations of question and answer passage pairs.

In the future, we would like to assess the performance of the reranking model using structural representations that can take into account additional information such as the category and the lexical answer type of the question.

References

1. Severyn, A., Nicosia, M., & Moschitti, A. (2013, August). Learning adaptable patterns for passage reranking. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning (pp. 75-83).
2. Joachims, T. (2002, July). Optimizing search engines using clickthrough data. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 133-142). ACM.
3. Severyn, A., & Moschitti, A. (2012, August). Structural relationships for large-scale learning of answer re-ranking. In Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval (pp. 741-750). ACM.
4. Pianta, E., Girardi, C., & Zanoli, R. (2008, May). The TextPro Tool Suite. In LREC.

[5] We used the same set of syntactic features who performed best in the Semantic Textual Similarity (STS) task at SemEval-2012