Towards Knowledge-enriched Cross-Lingual Answer Validation

Valentin Zhikov[1] and Georgi Georgiev[1]

Ontotext AD,
Polygraphia Office Center fl. 4, 47 A Tsarigradsko Shosse, 1504 Sofia, Bulgaria
{valentin.zhikov,laura.tolosi,georgiev}@ontotext.com
http://www.ontotext.com

Abstract. Our baseline approach from the 2012 year includes three language-independent methods for the task of answer validation. All methods are based on a scoring mechanism that reflects the degree of similarity between the question-answer pairs and the supporting text. We evaluate the proposed methods when using various string similarity metrics, such as exact matching, Levenshtein, Jaro and Jaro-Winkler. In addition to this baseline approach, we take advantage of the multilingual QA4MRE dataset, and devise an ensemble method, which chooses the answer indicated as correct by the largest number of analyses of the individual translations. Finally, we present a language-augmented method that enriches the questions and answers with paraphrases obtained by means of machine translation. We show that all of the described approaches achieve a significant improvement over the random baseline, and that both majority voting and language augmentation lead to superior accuracy as compared with the original method. However, the addition of some knowledge-based components in year 2013 plus the complexity of the datasets led to decrease in overall accuracy for Bulgarian language.

Key words: answer validation, approximate matching

1 Introduction

Question answering (QA) is a difficult problem situated at the intersection of several domains, including natural language processing and knowledge representation [1]. A subproblem of question answering is the answer validation task, which consists of deciding whether a given answer is correct or not, based on a text collection. The problem of answer validation remains challenging, the state-of-the-art performance being not larger than 60% accuracy [2], whereas the human performance is around 80% [2]. In the frames of the QA4MRE competition at CLEF, many approaches for answer validation have been proposed. The techniques employed include part-of-speech tagging, named entity recognition, syntactic transformations, semantic role labeling, logical representations, theorem provers and others. Many QA systems make use of external knowledge resources such as encyclopedia, ontologies, gazetteers, thesauri, etc. An optimal

combination between these approaches and resources is necessary in order to provide with a performant system.

Identifying paraphrases of the question and answer in the supporting text helps locating the sentences containing their correct answer. In order to obtain paraphrases, semantic and syntactic resources have been used [6], [7], [8]. In this article, we describe our current system, which builds on machine translation techniques for generating paraphrases, by translating text to another (dissimilar) language and then back to the source language. Our experience with statistical machine translation (by our involvement into the MOLTO European project[1]) shows that the resulting text is not identical with the initial text, but often contains synonymous paraphrases.

This year we also rely on additional preprocessing strategies as well as on some lexical resources, such as synonymy dictionary as well as paraphrases, extracted from Wikipedia. We focused on Bulgarian question answering task only, preforming 8 runs. As it is discussed in the next sections, the results drop in accuracy in comparison with the previous year's ones.

2 Method

We present here three methods for answer validation: an overlap-based algorithm (denoted by OV), a language augmented approach which builds on top of the overlap approach (called LAM-OV) and an ensemble model based on majority voting called voting overlap (V-OV). We will make use of the following simple notations: the questions are denoted by $Q(1), ..., Q(n)$, the answers pertaining to question i are denoted as $A(i, 1), ..., A(i, 5)$ and the supporting text is called T.

We note that both our algorithms always indicate the best scoring answer as the correct answer (according to our scoring scheme) and never leave a question unanswered. Also, our approaches are entirely based on the supporting, but this time we consider also some additional knowledge sources).

2.1 The OV method

The OV algorithm performs two steps: first, a filtering approach selects only the sentences from the supporting text that are similar to both the question and the answers. Then, the pairs (answer, supporting sentence) that yield highest similarity are returned.

More precisely, for each question $Q(i)$, the OV algorithm performs the following steps: first, it compares the lexical overlap between all concatenated question-answer pairs $\{\langle Q(i), A(i, 1)\rangle, ..., \langle Q(i), A(i, 5)\rangle\}$, and all sentences of the supporting text $s_1, ..., s_{|T|} \in T$. The overlap is computed using a function δ_ϕ of some similarity measure between text snippets ϕ. We will discuss the scoring function δ_ϕ and our choices for ϕ later in this section. We proceed by computing a

[1] http://www.molto-project.eu/

relevance score:

$$\rho(s_k) = max_{j=1,...,5}\delta_\phi(\langle Q(i), A(i,j)\rangle, s_k), k \in \{1, ..., |T|\}$$

and retain the top-scoring l sentences for futher analysis, concatenating them into a single long string. Hence, for each question $Q(i)$, a text extract $S(i)$ results. These extracts combine the sentences that are most relevant to any of the given question-answer combinations.

Then, the OV algorithm ranks the answers $A(i,1), ..., A(i,5)$ in decreasing order by their similarity to the text in $S(i)$. The pair with largest similarity $\delta_\phi(A(i,j), S(i))$ gives the winning answer $A(i,j)$ to the question $Q(i)$.

The number l and the similarity measure ϕ are parameters of the OV method. In our experiments, we tried several values of $l \in \{1,2,3,4,5\}$ and several similarity measures ϕ. Specifically, for two text snippets (e.g. sentences, represented as bag-of-words), a target t_0 and an arbitrary t, the similarity between t and the target t_0 is defined as follows:

$$\delta_\phi(t_0, t) = \frac{\sum_{i=1}^{|t_0|} \max_{j=1}^{|t|} \phi(t_0(i), t(j))}{|t_0|},$$

where ϕ corresponds to a distance measure between two words. In our experiments, ϕ is either exact matching, Levenshtein [3], Jaro [4] or Jaro-Winkler [5] similarity. For the final models, we selected the values of l and ϕ that gave best results on the corpus from CLEF2011. (Pseudo-code for the described algorithm is available in Appendix A.)

2.2 The V-OV method

The V-OV approach that we present is an ensemble method. For a specific question, each of the models based on the parallel corpora vote for the correct answer choice. The answer that gathers most votes is indicated as correct. The assumption that we make is that some answers are easier to validate in some languages and more difficult in others. However, this approach heavily relies on the parallelism of the corpora in different languages, in the sense that the sentences forming the supporting text, the questions and the answers must carry the same information, and the questions and answers must follow the same order. Also, the prediction of the correct answer is identical, irrespective of the target language.

2.3 The LAM-OV method

The LAM-OV method uses automated translation as a means of enriching the text with paraphrases and synonyms prior to executing the answer selection algorithm, in order to improve its performance. Specifically, for each target language, we transform the questions and answers by successive translations into intermediate languages. For example, in order to obtain several (synonymous)

paraphrases in Bulgarian, we translate the question and answers from the Bulgarian corpus into other languages (English, German, Swedish, Arabic) and then back to Bulgarian. Thus, the answers that contain paraphrases of the support text have a higher chance of being matched. We used the online Google Translate[2] service for obtaining translations.

2.4 Preprocessing

Before the algorithms are applied we perform the following preprocessing steps. All questions, answers and supporting text are converted to lower-case. Next, possible abbreviations are discovered via a regular expression that looks for recurring sequences comprising letters and periods without any white-space characters in between, and the period symbols are deleted from the matched sequences. Also, we added several rules that instruct the algorithm to ignore several common abbreviations of the type 'years' (г.), 'millions' (мил.), 'billions' (бил.), etc. by eliminating the period character in such cases. The supporting text is then segmented into sentences by splitting the transformed strings at each remaining period symbol. All text undergoes one more phase of preprocessing, through which symbols other than numbers and letters are replaced with white-space characters (we use a common mask for all languages apart from Arabic, for which our system is not directly applicable). Eventually, we tokenize each sentence using the resulting white-space subsequences as a delimiter.

This year we also added stemming as a preprocessing step, a synonymy lexicon and compiled lexicons from Wikipedia. We used stemming in processing English questions and answers for getting a better generalization on the abstract semantic level. Note that the used Wikipedia resource contains not only typical paraphrases in the sense of synonyms, but also extensions of abbreviations, hyperonymic relations, etc.

3 Results from year 2012

For the QA4MRE competition at CLEF2012 we submitted a total of 10 models. A summary can be found in Table 1. We submitted models based on the OV method for 6 of the languages included in the competition. Performance figures are presented in the last column of Table 1. The performance is around 0.30 (accuracy), with larger values for Italian, Romanian and English and worse results for Bulgarian, German and Spanish. A similar trend was observed when applying the algorithms to the reading tests included in the CLEF2011 dataset. More details on the performance of the OV algorithm are given in Table 2. We show how the results vary with the choice of parameters l and ϕ, on two corpora (from 2011 and 2012) and for two of the languages (Bulgarian and English). We used the 2011 corpus for selecting the optimal parameters, specifically $l = 3$ and $\phi = \phi_{ExactMatching}$. These values maximized the mean accuracy of the system

[2] http://translate.google.com/

ID	Method	Language	Perf
01	OV	Bulgarian	0.28
02	OV	English	0.31
03	OV	Italian	0.35
08	OV	Romanian	0.34
09	OV	German	0.28
10	OV	Spanish	0.28
04	V-OV	Bulgarian	0.29
05	V-OV	English	0.29
06	V-OV	Italian	0.29
07	LAM-OV	Bulgarian	0.30

Table 1. Experiments submitted. Description of the method is given in the second column. Last column indicates the accuracy of the model.

		EN				BG			
	l \ ϕ	E	L	J	J-W	E	L	J	J-W
2011	1	0.36	0.26	0.26	0.28	–	–	–	–
	2	0.36	0.3	0.3	0.26	–	–	–	–
	3	**0.38**	0.3	0.32	0.28	–	–	–	–
	4	0.36	0.30	0.31	0.31	–	–	–	–
	5	0.36	0.31	0.32	0.32	–	–	–	–
2012	1	0.31	0.34	0.31	0.32	0.3	0.3	0.3	0.27
	2	0.33	0.34	0.31	0.29	0.27	0.24	0.29	0.29
	3	0.31	0.33	0.33	0.31	0.28	0.28	0.27	0.29
	4	0.28	0.31	0.31	0.31	0.29	0.29	0.26	0.29
	5	0.27	0.31	0.33	0.31	0.28	0.31	0.25	0.29

Table 2. Performance of the OV model for English and Bulgarian. Results for the corpora from 2011 and 2012 are shown. Values corresponding to parameters l and ϕ are presented, optimal values being indicated by the marked cell from the 2011 corpus. The values of the similarity δ_ϕ are E (exact match), L (Levenshtein), J (Jaro) and J-W (Jaro-Winkler).

calculated against the reading tests in all supported languages. In Table 2, the accuracy corresponding to these parameters is marked (0.38).

The performance of the voting algorithm V-OV (0.29) is superior than that of the OV algorithm for several languages, including Bulgarian, Spanish and German, but worse for English, Italian and Romanian (Table 1). The poor score is the consequence of the lack of parallelism between the corpora, meaning that the reading tasks, questions and answers were arranged in different order in the 2012 corpus available at submission time. We repeated our experiments against the synchronized dataset released after the system submission and found out that a simple ensemble voting scheme that excludes the worst-performing systems (Spanish and German languages, according to the results for the 2011 corpus) would have achieved an accuracy of 0.38. We report this number in this manuscript as the best result that we have obtained against the CLEF2012 dataset.

We applied the LAM-OV approach only to the Bulgarian corpus. In order to enrich the questions and answers with paraphrases, we translated the original corpus to several other languages and then back to Bulgarian. We performed three such experiments, where the intermediate languages were: *i*) English, *ii*) German and *iii*) Swedish followed by Arabic. For cases *i*) and *ii*), we obtained 0.29 accuracy. In the case *iii*), the accuracy reached 0.31. In all cases, we improve the OV baseline. We carried out an additional experiment in which we concatenated all translations (from German, English and Swedish/Arabic) to the original. The performance of the model was 0.31.

4 Current results

This year we performed tests only on Bulgarian data, using LAM-OV approach only and adding of new pre-processing steps. The performed runs were 8. They are presented in Table 3.

ID	Method	Language	Perf
01	LAM-OV	Bulgarian	0.18
02	LAM-OV	Bulgarian	0.18
03	LAM-OV	Bulgarian	0.18
04	LAM-OV	Bulgarian	0.18
05	LAM-OV	Bulgarian	0.22
06	LAM-OV	Bulgarian	0.23
07	LAM-OV	Bulgarian	0.24
08	LAM-OV	Bulgarian	0.24

Table 3. Experiments submitted. Description of the method is given in the second column. Last column indicates the accuracy of the model.

All the models include stemming and synonymy enrichment. Additionally, the first four models use paraphrases from Wikipedia. The results show, however, that these models perform under the baseline and results from year 2012. The next four models do not rely on paraphrases. They additionally have the following specific features:

1. 05 model: looks up the answers in 1 sentence, which has the highest score; it does not give an answer, if it is not sure about it.
2. 06 model: looks up the answers in 1 sentence, which has the highest score; it always gives an answer.
3. 07 model: looks up the answers in top 3 sentences, which have the highest score; it does not give an answer, if it is not sure about it.
4. 08 model: looks up the answers in top 3 sentences, which have the highest score; it always gives an answer.

It seems that the best score is achieved by the systems which look up in top 3 sentences, which means - in a wider context; and irrespectively of their confidence behaviour. It should be noted that the addition of a knowledge-rich, but somewhat noisy resources, such as the Wikipedia-derived lexicon, performs worse than the models without it. In general, however, all the results are bellow the last year's ones. This might be interpreted as follows: the canonical lexicon-based synonymy is not enough for handling the question-answer paraphrases. The addition of stemming seems suitable for the English part of data, but not so helful for the generation of Bulgarian pairs.

5 Discussion

The OV algorithm is a very simple and generic approach, which can be applied to most of the languages included in the QA4MRE dataset without any supplementary resources. Its generality comes at the price of modest performance, although the accuracy is significantly larger than a random baseline of 0.20 (which picks the correct answer uniformly at random among the choices).

The OV approach essentially searches for common words between supporting text and question/answers using an approximate string matching paradigm. Interestingly, we found that metrics like Levenstein, Jaro, and Jaro-Winkler, which reflect the small differences between words, were not better than exact matching with respect to system performance. We expected that approximate matching would have a similar effect to applying a lemmatizer, with the advantage of language independence. However, the experiments did not support our expectations.

One of the reasons why our overlap-based method did not perform very well in 2012 lied in its inability to address more complex textual inferences, such as synonymy, paraphrases, nominalization/verbalization, etc. (Refer to [2] for more information regarding the use of specific means of expression.) Our error analysis revealed that a large fraction of the errors were indeed attributable to paraphrasing. In 2012 paper, we presented the language-augmented method as a cheap and fast, albeit not highly accurate, approach to obtaining paraphrases. The approach is based on bidirectional machine translation (to the target and then back to the source language) performed using Google Translate. We rely on the statistical variance of the automated translator, which, if applied several times with different intermediate languages, is likely to output a rich set of synonyms and paraphrases. We also believe that the more different the intermediate language is with the target language, the more likely it is to obtain paraphrases. This year we added some lexical knowledge to face the paraphrase variety in natural language. However, the results remained below the baseline and last year's accuracy values.

Below we list three classes of issues addressed by the language-augmented technique in 2012.

The first one is the generation of synonyms, in a form suitable for exact matching. For instance, we have been able to generate the term "states" from a sentence/answer pair containing the closely related term "countries" (originally: "страни" and "държави", in Bulgarian). Other examples include: "American"/"U.S." ("американското" / "на САЩ"), pairs of interchangeable Bulgarian terms for "industry" ("промишленост" / "индустрия"), "electricity" ("електричество" / "ток"), etc.

The second one is the generation of paraphrases, such as "част от Африка, южно от Сахара" and "част на Африка на юг от Сахара" (two expressions roughly translated as "a part of Africa to the south of Sahara"). Albeit the phrases generated in this way are not always gramatically correct, this class of transformations has the advantage of providing a more varied set of word forms

given a term from the source text, and thus can improve the recall of matching during the candidate scoring phase.

Last, we observed issues related to the alternative representations of numerical values. For instance, the correct answer to the question "For how long has Rebecca Lolosoli been working with MADRE?" (reading test 4, question 9, synchronized gold standard dataset) has been provided in both a numerical and lexical forms, accross the various translations of the dataset. As Google Translate can interchange numerical values with their string representation in some cases (that seem to depend on the particular choice of a language pair), the language-augmented method can be regarded as a simple ad-hoc approach for resolving this kind of issues.

Presently, we consider the best scoring sentences from the text as likely to contain the answer, based on the assumption that the answer is indeed contained in the provided supporting text. However, in a general setting, some or all of the best scoring sentences might not be 'good enough', in the sense that their overlap with the question/answers text is very low. Introducing a minimal threshold parameter that eliminates sentences with too small overlap can for example result in unanswered questions - a choice which is encouraged by the evaluation system at the QA4MRE challenge. Also, it would allow for efficient scanning of very large collections of text, in addition to the corpus provided. Choosing a minimal threshold for text similarity can be for example done by comparing the distributions of the similarities between question/true answer and sentences containing answer and the rest of the similarities (false answers, arbitrary text, etc).

6 Conclusions

In this paper, we have described the array of algorithms that constitute our consequent submission to the QA4MRE at CLEF2012 and CLEF2013 competition. The reported results reveal that our basic algorithm outperforms the random baseline irrespective of the language of the analyzed textual content, without resorting to any side resources nor language-specific tools.

We have shown that the results of the basic system can be improved significantly by incorporating a mechanism for majority voting based on the analysis of the individual translations included in the data collection. Also, we have shown that bidirectional statistical machine translation can introduce some amount of variation in the corpus that allows for improved overlap-based approaches. Last, we tried our system in 2013 with the addition of stemming and synonymy/paraphrases addition. The results, however, remained below the last year's experiments.

We could extend our unsupervised and language-independent approach by incorporating some importance-based weighting scheme (such as tf*idf) into the score computation mechanism, in order to boost the scores of answers containing terms of high relevance within the context of a concrete article. Similarly, an instantiation of the language-augmented approach that enriches the queries and

answers with semantically close terms, extracted by some clustering technique from the background collection, could also lead to a better performance.

Acknowledgements

This work was partially supported by the MOLTO European project (FP7-ICT-247914). Ontotext AD is a part of the MOLTO consortium.

References

1. Peñas, A., Rodrigo, Á., Sama, V., Verdejo, F.: Testing the Reasoning for Question Answering Validation. J. Log. and Comput. 18(3), 459–474 (2008)
2. Peñas, A., Hovy, E., Forner, P., Rodrigo, Á., Sutcliffe, R., Forascu, C., and Sporleder, C.: Overview of QA4MRE at CLEF 2011: Question Answering for Machine Reading Evaluation, CLEF 2011 Labs and Workshop Notebook Papers. (2011)
3. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady. 10(8), 70–710 (1966)
4. Jaro, M. A. : Advances in record linkage methodology as applied to the 1985 census of Tampa Florida. Journal of the American Statistical Society, 84(406): 414–420 (1989)
5. Winkler, W. E. : String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. Proceedings of the Section on Survey Research Methods (American Statistical Association), 354–359 (1990)
6. Lin, D. and Pantel, P. : DIRT – discovery of inference rules from text. Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. 323–328 (2001).
7. Barzilay, R., McKeown, K. R. and Elhadad, M. : Information fusion in the context of multi-document summarization. Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics. 550–557 (1999).
8. Richardson, S. D. : Determining similarity and inferring relations in a lexical knowledge base. PhD thesis (1997).

Appendix A

The OV Algorithm

```
program OV (l, phi)
   {Assume given the three components:
      - supporting text (T)
      - questions set Q(1), ..., Q(n)
      - corresponding multiple answers A(i, j), i=1..n, j=1..m}

   Preprocessing
      Trim spaces from T, Q and A;
      Apply lowercase conversion to T, Q and A;
      Remove "." from abbreviation-like strings; #matched using regex
      Segment into sentences T, Q and A, using the "." delimiter;
      Apply sentence tokenization based on white space characters;

   Identifying the correct answer
      for each question Q(i), i=1..n
          Remove first word of Q(i)
          for each sentence S(k), k=1..length_in_sentences(T)
              for each answer A(i, j), j=1..m
                  V(j) := Concatenate Q(i) and A(i,j);
                  score(S(k)) := max(score(S(k), delta_phi(V(j) and S(k))))
              endfor
          sort S by score(S(k)) in descending order
          R(i) = S(1) + ... + S(l)   # concatenate highest-ranking l sentences
          endfor

          highestSimilarity := -Inf;
          for each answer A(i, j), j=1..m
                  s := delta(A(i, j) and R(i));
                  if s > highestSimilarity
                      bestAnswer := A(i, j);
                      highestSimilarity := s;
                  endif
              endfor
          endfor
          Output bestAnswer for Q(i);
      endfor
end.
```

The OV Algorithm. The basic algorithm that underlies all of the described methods. Described in detail in section 2.1.