A Text Mining Approach as Baseline for QA4MRE'12

Mathias Verbeke and Jesse Davis

Department of Computer Science, Katholieke Universiteit Leuven, Belgium
{mathias.verbeke,jesse.davis}@cs.kuleuven.be

Abstract. This paper describes the participation of the KU Leuven DTAI team in the pilot task on machine reading of biomedical texts about the Alzheimer disease, which is part of the 2012 Question Answering for Machine Reading Evaluation campaign (QA4MRE'12). The main objective of our research was to develop a text mining system as a strong baseline for the task. Based on the outcome of this system, we want to investigate which types of questions can be answered based solely on the input text and the question string, and for which ones we need more advanced techniques that also consider the previously acquired background knowledge from the reference document collection. Furthermore this should give us some insights into the system behavior for specific question types and background information for the development of a tailored inference algorithm.

Keywords: Question Answering, Machine Reading, Text Mining, Baseline, Biomedical Natural Language Processing

1 Introduction

Machine Reading (MR) or Natural Language Understanding [1] is one of the core objectives since the emergence of Artificial Intelligence and Natural Language Processing. Its main goal is to automatically extract knowledge from unstructured text and to use this knowledge to make decisions or answer questions. The latter is also the focus of the Question Answering for Machine Reading Evaluation (QA4MRE [2]) campaign, where the task is to read single documents and identify the answers to a set of multiple-choice questions about information that appears explicitly of implicitly in the text.

The pilot task on machine reading of biomedical texts aims at exploring the ability of systems to answer questions about a specific scientific topic, namely the Alzheimer disease. Question answering in this domain poses additional challenges for natural language processing which mainly arise because domain knowledge is essential for achieving deep understanding in this setting. Consider the following example for the QA4MRE pilot task on MR for biomedical texts:

Text: [...] *Additionally, no estrogen could be detected in the APP23/Ar / mice (data not shown), suggesting that aromatase gene knock-out prevented the conversion of endogenous testosterone into estrogen.* [...]

Question: *What experimental approach is useful to create an in vivo system where conversion of testosterone into estrogen is blocked?*
Candidate Answers:

1. *ELISA analysis*
2. *hole-board memory task*
3. *NEP activity assay*
4. *Western blot*
5. *knock-out of the aromatase gene*

In this case, the machine reading system should identify 'knock-out of the aromatase gene" as the correct answer.

In order to evaluate the system behavior for specific question types and investigate the necessity of using and reasoning about the background knowledge contained in the reference document collection, our approach is based on basic text mining techniques and does not make use of any existing resources. It is developed to provide some insights towards a tailored inference algorithm and can be seen as a strong baseline for the task.

The paper is organized as follows. The methodology of the system built for QA4MRE'12 is presented in section 2. Section 3 discusses the results and analyses them by means of a detailed error analysis. Finally, section 4, concludes and presents ideas for future work.

2 Methodology

The main goal of our approach is to investigate the importance of background knowledge for MR of biomedical texts. Therefore our system is solely based on basic text mining techniques, and does not rely on the reference document collection or any other external resources. Since the outcome should be able to serve as a baseline for the task, preprocessing is kept to a minimum. Sentence splitting is performed, for which we relied on the splits from the preprocessed files provided by the task organizers. For stopword removal, the English stop word corpus provided by NLTK [3] was used.

We propose two different strategies, that mainly differ in the order in which the text mining techniques are applied. Both approaches only use the input text, the question string, and the multiple choice answers.

2.1 Approach 1: Question Similarity

The first approach computes the similarity between each question in the reading test and every sentence in the input document and selects the top k most similar sentences. Next, each sentence votes on an answer by checking to see if it contains (part of) an answer. If it does, the respective answer's vote is incremented by either 1 or the normalized similarity value. The answer with the highest number of votes, i.e., the highest weight, is selected. The pseudocode (Algorithm 1) describes the question similarity approach.

Algorithm 1 Pseudocode of the question similarity algorithm

```
1: for all question q in reading test do
2:     q_tok = wordTokenize(q)
3:     for all sentence s in input document do
4:         s_tok = wordTokenize(s)
5:         similarity(q_tok, s_tok)
6:     end for
7:     for all answer a in choice list do
8:         a_tok = wordTokenize(a)
9:         for all top k sentence ts do
10:            if element of a_tok in ts then
11:                incrementVote(a)
12:            end if
13:        end for
14:        return top voted a
15:    end for
16: end for
```

For the similarity calculation in line 5, we employed two different measures, namely the Jaccard similarity [4] (Equation 1) and the MASI distance [5] (Equation 2). Both measures operate on sets, so both the sentence from the input document as well as the question need to be tokenized to transform them into a set of words. Suppose S is the set of words from a sentence in the input document and Q is the set of words from the question. Then the two measures are defined as follows:

$$Jaccard(S, Q) = \frac{|S \cap Q|}{|S \cup Q|} \tag{1}$$

and,

$$MASI(S, Q) = 1 - \frac{|S \cup Q|}{max(|S|, |Q|)} \tag{2}$$

To convert the MASI distance into a similarity measure, we subtract it from 1. As can be seen from the formula, the MASI distance is a weighted version of the Jaccard similarity, that takes into account partial agreement between sets, by downweighting the Jaccard score when partial overlap between the sets exists.

For tokens that refer to numbers, we also check if the written form of numbers is contained in the sentence. In future work we also plan to integrate the reverse, i.e. if the numerical form of a written token appears in the sentence.

We also explored the following variations of algorithm 1:

Answer concatenation Instead of first calculating the similarity between the question and the sentence, followed by checking if the answer appears in the sentence, the question and the answer can also be concatenated before the similarity calculation. Subsequently the concatenation of the question and every

answer is compared with every sentence in the input document. Lastly, the average similarity of the k most similar sentences is calculated from which the best one is chosen.

Weighting by similarity Instead of giving the same weight to each answer when it appears in a top k sentence, the answers can also be weighted with the calculated similarity value. This sum of similarity values is then normalized by the number of times the answer was voted for, before it is used to select an answer based on majority voting.

Overall similarity If we drop the top k requirement when voting for the answer in the last step of the algorithm, all sentences that are not completely different from the question (i.e. have a MASI and Jaccard similarity different from 0) are taken into account in the voting process. The answer that is contained in the greatest number of the sentences is chosen.

2.2 Approach 2: Answer Containment

The second approach reverses the procedure of the question similarity approach, and first checks if the answer appears in the sentence. If it does, the similarity is calculated and the sentence with the highest similarity is selected, which can be seen from the pseudocode in Algorithm 2.

Algorithm 2 Pseudocode of the answer containment algorithm

```
1: for all question q in reading test do
2:     q_tok = wordTokenize(q)
3:     containingSentences = [ ]
4:     for all answer a in choice list do
5:         a_tok = wordTokenize(a)
6:         for all sentence s in input document do
7:             s_tok = wordTokenize(s)
8:             if element of a_tok in s_tok then
9:                 containingSentences.append(s_tok)
10:            end if
11:        end for
12:        for all sentence cs in containingSentences do
13:            cs_tok = wordTokenize(cs)
14:            similarity(q_tok, cs_tok)
15:        end for
16:        for all top k sentence ts do
17:            incrementVote(a)
18:        end for
19:        return top voted a
20:    end for
21: end for
```

For the similarity calculation, the weighting by similarity approach as described for Algorithm 1 is used, with MASI as the similarity measure.

3 Results and Discussion

These two approaches and the described variations were tested on the 40 questions from the four reading tests. The scoring of the output for each reading test is done using c@1 [6], an evaluation measure between 0 and 1 that rewards systems that, while maintaining the number of correct answers, are able to reduce the number of incorrect ones by leaving some questions unanswered. It can be formulated as follows:

$$c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n}) \tag{3}$$

where

n_R: the number of correctly answered questions
n_U: the number of unanswered questions
n: the number of questions

Furthermore, accuracy was measured.

In Table 1, the question answering evaluation for each run is listed. It contains the number of answered and unanswered questions, together with the accuracy and c@1 measure over all readings tests. Furthermore, it shows the used algorithm, its parameter setting for k (i.e. the number of most similar sentences taken into account in the voting process), the employed similarity measures and the variations to the basic algorithm.

	unanswered				answered			all		algorithm			
Run	**#**	**Right**	**Wrong**	**Empty**	**#**	**Right**	**Wrong**	**Acc.**	**C@1**	**algor.**	**k**	**similarity**	**variation**
01	7	0	0	7	33	7	26	0.18	0.21	alg. 1	5	MASI	n/a
02	8	0	0	8	32	6	26	0.15	0.18	alg. 1	5	Jaccard	n/a
03	0	0	0	0	40	9	31	0.23	0.23	alg. 1	10	MASI	n/a
04	0	0	0	0	40	10	30	0.25	0.25	alg. 1	10	Jaccard	n/a
05	0	0	0	0	40	10	30	0.25	0.25	alg. 1	5	MASI	Answer concat.
06	0	0	0	0	40	12	28	0.30	0.30	alg. 1	n/a	MASI	Weighting
07	0	0	0	0	40	8	32	0.20	0.20	alg. 2	5	n/a	n/a
08	0	0	0	0	40	8	32	0.20	0.20	alg. 2	10	n/a	n/a
09	0	0	0	0	40	10	30	0.25	0.25	alg. 1	n/a	n/a	Overall similarity
10	0	0	0	0	40	12	28	0.30	0.30	alg. 1	n/a	Jaccard	Weighting

Table 1. QA level evaluation for each run

Based on the results for the full dataset, the following observations can be made:

- For the standard version of the first algorithm, a higher k value has a positive influence on the results. This is caused by the fact that when k is set to 5, for some questions, none of the answers appear in the selected sentences, meaning they do not get answered (respectively 7 and 8 questions, in run 01 and 02), whereas with a higher k, the correct answer can be selected in some of the cases. During development, higher numbers of k were tried on the sample reading test, but this did not yield better results.
- In algorithm 2, it is checked if the answer tokens appear in sentences of the input document at the start of the algorithm. Since for every question, one or more of the answers could be found verbatim in the input document, this gave no option for leaving the question unanswered, which seems to have a negative effect on the results.
- The variation of algorithm 1, where the answers are weighted by similarity, gives on average the best results, with a score of 0.30 for both accuracy and c@1. Furthermore, this seems to be independent of the similarity measure used, with equal scores for MASI and Jaccard.
- Using the answer concatenation and overall similarity variants of algorithm 1 resulted in small improvements when compared to the standard version, where MASI was used as the similarity measure.

Table 2 lists the results for the four individual reading tests. For each reading test, the c@1 score is shown, whereas also the median, the average and the standard deviation over all the reading tests are given. From these results, the following observations can be made:

- On average over all runs, reading test 2 gave the best results (average c@1 of 0.317), followed by test 4 (0.259), test 3 (0.24) and test 1 (0.121). Although test 2 is also the one with the highest standard deviation, whereas task 4 has the lowest.
- The best scoring runs, namely 06 and 10, which both use the weighted by similarity variation of algorithm 1, also have the lowest standard deviation, which could be an indication that this algorithm is more robust to different types of questions.
- The highest individual reading test score of 0.50 is achieved by run 09 for test 2. The overall similarity approach used for this run highly resembles the weighting by similarity variant, except for the top k requirement. If the answer to some of the questions in reading test 2 was only mentioned in sentences that were more dissimilar than the k most similar ones, this could explain this result.

Also at the individual question level, we analyzed our results, with the following main observations:

- The following questions could be correctly answered in at least half of the runs (the number before the dot indicates the reading test, the number behind it the question, and the one between brackets the number of runs in which it was answered correctly): 1.2 (5), 2.6 (9), 2.8 (7), 3.3 (5), 4.2 (8) and

	individual reading tests				overall		
Run	**1**	**2**	**3**	**4**	**Median**	**Average**	**Std. dev.**
01	0.11	0.11	0.36	0.26	0.19	0.21	0.12
02	0.00	0.36	0.24	0.13	0.19	0.18	0.15
03	0.10	0.30	0.30	0.20	0.25	0.23	0.10
04	0.20	0.40	0.20	0.20	0.20	0.25	0.10
05	0.10	0.30	0.30	0.30	0.30	0.25	0.10
06	0.20	0.40	0.30	0.30	0.30	0.30	0.08
07	0.10	0.20	0.20	0.30	0.20	0.20	0.08
08	0.10	0.20	0.10	0.30	0.20	0.20	0.12
09	0.10	0.50	0.10	0.30	0.20	0.25	0.19
10	0.20	0.40	0.30	0.30	0.30	0.30	0.08

Table 2. Detailed QA level evaluation per reading test

4.6 (6). For all of these questions, the input document contained sentences that included one or multiple tokens from the question together with the answer.

- The proposed approach does not perform an analysis of the question beforehand. Doing so could help in correctly answering a substantial number of questions. For example, in question 4.4, the experimental technique that was used to purify the γ-secretase complex is asked for. Since only 2 of the 5 questions deal with experimental techniques, the other answers could be discarded beforehand.
- 13 out of the 40 questions were incorrectly answered in all runs. A deeper analysis should confirm this, but we suppose answering these questions requires combining information from multiple sentences, performing some form of inference.

When the test set preparation procedure is made available, we would like to analyze to which level of question difficulty our system is able to answer questions correctly. In order to do this, it should be known which facts are explicitly stated in the text, which ones are present but are not explicitly related and for which facts inference is needed to connect them and form the answer. This will also clarify to what extent the reference background document collection is a prerequisite to answer certain (types of) questions. In addition, this will also give some insights on which kinds of textual inference and features are important for the task, i.e., lexical information (e.g., acronym, synonymy, hyperonymy-hyponymy), syntactic structures (e.g., nominalization-verbalization, causative, paraphrase, active-passive) or the identification of discourse relations (e.g., co-reference, anaphora ellipsis).

From our current system design, it is clear that only questions for which the answer is mentioned in the same sentence can be answered, since it does not combine information at the document level. Nor does it consult the background collection or perform inference to reason about facts stated in the text. In a

more extensive error analysis, we will also analyze the influence of the similarity measure on the results.

Furthermore, the answer level needs to be analyzed, in order to determine which types of answers the system is able to distinguish. For example, can the system discriminate between different answers if they are all enzymes, or only answer questions that ask for an entity of which the type is unique in the answer list? Currently, the only preprocessing that is done at the answer level is the transformation of verbatim numbers to their numerical form, but other transformations may be necessary.

4 Conclusions and Future Work

We presented the methodology of our system for machine reading of biomedical texts about the Alzheimer disease. Question answering about a scientific topic poses additional challenges to a MR system, which is mainly caused by the increased importance of the background knowledge. The main purpose of our approach was to investigate how far-reaching this influence was. Therefore we developed a system using basic text mining techniques, without considering any external resources.

The proposed approach can be seen as a strong baseline for the task. Futhermore, the error analysis of the results provides valuable insights for further improvement. In future work, we want adopt this knowledge to develop a tailored inference algorithm for this task.

5 Acknowledgements

Mathias Verbeke is funded by the Research Foundation Flanders (FWO-project G.0478.10 - Statistical Relational Learning of Natural Language). Jesse Davis is partially supported by the Research Foundation Flanders (FWO-project G.0356.12 - A Synergestic Approach to Extraction, Learning and Reasoning for Machine Reading).

References

1. Allen, J.: Natural Language Understanding (2nd edition). Benjamin/Cummings (1995)
2. Question Answering for Machine Reading Evaluation, `http://celct.fbk.eu/QA4MRE`
3. Bird, S., Klein, E., and Loper, E.: Natural Language Processing with Python. O'Reilly Media (2009)
4. Jaccard, P.: Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la Société Vaudoise des Sciences Naturelles, 547–579 (1901)
5. Passonneau, R.: Measuring agreement on set-valued items (MASI) for semantic and pragmatic annotation. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC) (2006)

6. Peñas, A., Rodrigo, A.: A Simple Measure to Assess Non-response. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 1415-1424, ACL, Portland, Oregon, USA (2011)