Simple similarity-based question answering strategies for biomedical text

David Martinez[1], Andrew MacKinlay[1], Diego Molla-Aliod[2], Lawrence Cavedon[1], and Karin Verspoor[1]

[1] National ICT Australia, Victoria Research Lab, Parkville VIC 3010, Australia
{david.martinez,andrew.mackinlay,lawrence.cavedon,karin.verspoor}@nicta.com.au
[2] Macquarie University
North Ryde, NSW Australia
diego.molla-aliod@mq.edu.au

Abstract. We introduce an approach to question answering in the biomedical domain that utilises similarity matching of question/answer pairs in a document, or a set of background documents, to select the best answer to a multiple-choice question. We explored a range of possible similarity matching methods, ranging from simple word overlap, to dependency graph matching, to feature-based vector similarity models that incorporate lexical, syntactic and/or semantic features. We found that while these methods performed reasonably well on a small training set, they did not generalise well to the final test data.

Keywords: question answering, biomedical natural language processing

1 Introduction

A combined NICTA/Macquarie team participated in the CLEF2012 QA4MRE pilot task on "Machine Reading of Biomedical Texts about Alzheimer". This task addresses the goal of obtaining a detailed understanding of the content of a text, in this case a text on the topic of Alzheimer's disease. A system's ability to interpret a text is measured practically through the performance of that system on a series of multiple choice questions about the text. Each question had five possible answers; the goal of the system was to select the correct answer from among those five.

We experimented with several simple approaches to this task, each based on the similarity of a candidate query constructed from the question plus a candidate answer to the information available in text. The approaches varied in the details of how similarity was measured, and what text sources were used to assess the relevance of the candidate answer. We will describe these details, and provide an assessment of the performance of each system variant, in the remainder of the paper.

2 Related Work

Question answering is a natural language processing task that has quite a long history of research. It was revived in the late 1990s with the Question Answering track of the Text REtrieval Conference (TREC) [23], with other evaluation-based competitions following suit. Systems typically focus on factoid question-answering where the answer is a specific fact such as a location, person, etc. They find the answer by first determining the answer type during a question classification step, retrieving a set of candidate documents or passages using standard information retrieval techniques, and then extracting the answer from those candidates. Answer extraction generally has involved techniques such as (a combination of) pattern matching [20], similarity matching with the question using simple word-based features [13] or Bayesian techniques [4], measurement of answer redundancy e.g. on the Web [2], and even methods based on logic [17].

Research on question answering methods specifically for biomedical text is a relatively new topic and systems attempt to find answers that are more complex than simple facts. Thus, MedQA [24] and AskHERMES [3] incorporate summarisation and clustering techniques. Other approaches such as Demner-Fushman et al.'s[6] system and EPoCare [16] use information extraction techniques to identify specific types of information relevant to biomedical research queries.

Research in multiple-choice question answering is less active though it is related to the task of answer validation, where question answering systems use techniques such as answer redundancy with the support of large corpora or the Web [12], or methods based on logical proving [17] and textual entailment [8]. Answer validation is the central task of the series of Answer Validation Exercises (AVE) at CLEF [18].

A similar approach to our methods is taken by [7], where three types of sentence similarity methods are explored: tree-distance, sequence similarity, and order invariant methods. Their empirical evaluation shows that the method to choose depends heavily on the testbed.

3 Approach

Each method that we experimented with is based on selecting the most likely answer from a set of multiple-choice candidate answers to a question, through an evaluation of the similarity of a candidate answer to the text in a given document or set of documents. The high-level process for each system variant was:

1. Construct a candidate query based on the combination of a question and a candidate answer.
2. Search the relevant text for sentences matching each candidate query.
3. Select the candidate query with the best match/most similar sentence in the text as the correct answer to the question.

The system variants differed on the matching algorithm employed, the matching criteria applied, and the text that was searched.

3.1 Preprocessing

For the primary article as well as the background articles, we worked from the pre-parsed sentences provided by the task organisers, since these were the cleanest source of data available, stripping the supplied dependency labels while preserving sentence boundaries and tokenisation. We normalised greek characters and numbers to address inconsistencies between the source article and the text in the questions and answers. Greek characters such as α were converted to the equivalent version spelled out using Latin characters (*alpha*), while spelled out numbers such as *three* were rewritten using digits (*3*). We parsed all text with ClearParser [5], a dependency-parser which has been demonstrated to have state-of-the-art performance on biomedical text [22], and which has a pre-trained biomedical parsing model available for download.

3.2 Construction of candidate queries using question/answer pairs

The starting point for each method is to construct a query based on a combination of the question and a candidate answer. The aim of this step is to produce a succinct statement of the information represented by a candidate answer, with the question providing appropriate context. For instance, a question/answer pair "Q: Which protein is known to remove Aβ from the brain? A: IDE" would be merged to form the query "IDE is known to remove Aβ from the brain". This process was applied for each candidate answer within the multiple-choice question. Thus we constructed 5 queries for each question.

Bag-of-words queries The simplest system we developed utilised simple word overlap as its matching algorithm. For this system, a correspondingly simple method was used to construct the query ("bag of words"), since the candidate sentences do not need to be grammatically sound. Each query was constructed from all question words excluding the initial question word, plus all of the words in the candidate answer.

Merging Question and Answer Graphs For some experiments, we used the outputs of a dependency parser to evaluate the similarity of question and answer sentences, to provide a semantically-richer method of comparison. This requires a distinct graph corresponding to each answer which also integrates the dependencies of the question to compare against the graphs from the evidence sentences. We achieve this by using a custom algorithm to insert the answer subgraph into the question graph. This merged graph also formed the starting point for the vector space model methods described in Section 3.4. The aim of this step is to produce a merged graph that looks very similar to the graph of a declarative sentence in which the particular answer to that question is stated. For example, consider the example question in (1) taken from the sample data and the corresponding answer in (2). Ideally, from the dependency graph of this question shown in Figure 1, and that of the answer fragment, we would create a

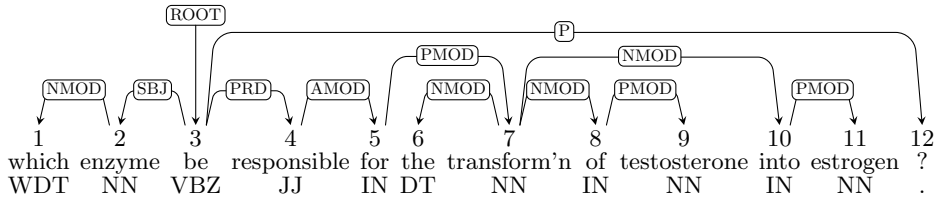

Fig. 1. The dependency graph of (1) produced by ClearParser

similar dependency graph to what would be obtained by parsing (3). We would like to replace the dependency nodes corresponding to *Which* and *enzyme* with a node for *Aromatase* derived from the answer subgraph.

(1) Which enzyme is responsible for the transformation of testosterone into estrogen?
(2) Aromatase
(3) ***Aromatase*** *is responsible for the transformation of testosterone into estrogen.*

The procedure for this which we use for most questions is as follows:

1. Find the "question node" – a single node within the question graph which has as its lemma any item from hand-created list of eight *Wh*-question lemmas, such as *what*, *how* and *where*. In Figure 1, this is node 1, for the token *which*.
2. Find the "root target node" in the question graph. This is the node linked by a dependency with label *ROOT*, generally corresponding to the main verb in a complete sentence. For Figure 1, this is node 3.
3. Find the subgraph corresponding to the path from the root target node to the question node, noting the label of link on the path closest to the root target node, which we denote the "question link label". Then delete this entire subgraph. This removes the nodes corresponding to the question phrase; in Figure 1, we would remove nodes 1 and 2 corresponding to the subject of the question sentence, *which enzyme*. In addition, if the question node is *how*, delete any connected nodes with lemma *many* (thus treating *how many* as a multi-token question word).
4. Find the root target node of the answer graph (which generally corresponds to the head of a noun phrase). Insert this node, along with all linked nodes (the complete graph apart from the root node) into the modified question graph, adding a link from the root target node of the question graph, with the label set to the question link label from above. The intuition here is that the answer should occupy the grammatical slot (most often *SBJ*) which was formerly filled by the question.

After applying this procedure to the graph in Figure 1 and the answer graph for the noun phrase *aromatase* from (2), we obtain the graph shown in Figure 2.

Upon examination of the test set, we found that further optimisation of these rules was required. In particular, these rules produced suboptimal results for questions phrased as full or reduced relative clauses such as (4) and (5).

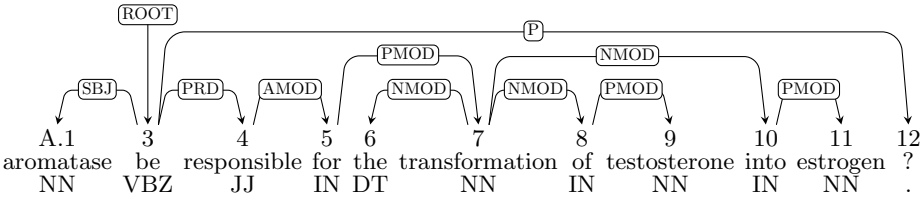

Fig. 2. The dependency graph from Figure 1 with the graph of the candidate answer *aromatase* inserted according to the method described, corresponding roughly with the parse we would obtain from (3).

(4) What is the major protease produced by microglia responsible for degrading A?
(5) What are the sst receptors that are expressed on rat astrocytes?

To handle these cases, we introduced a new set of rules to replace step 4 above, which we do not explain in detail here, but which attempt to directly attach the candidate answer graph to the main verb in the body of the relative clause or the omitted copular verb in the case of a restrictive relative such as (4). For the above examples, we would be attempting to produce combined dependency graphs which could be obtained by parsing sentences such as (6) and (7).

(6) ***IDE*** *is produced by microglia responsible for degrading A*
(7) ***Microglia*** *are expressed on rat astrocytes*

3.3 Word Overlap

The simplest algorithm that we experimented with considered the lexical overlap between the query "bag of words" described above, and the sentences in the reference corpus. We measured the number of overlapping words (where a word is defined as a single token), and assigned the candidate answer with highest word overlap with some sentence in the reference as the system's response. As reference corpus we experimented with different variants (reference document, background collection, and in-house background collection) as explained in Section 3.6. In case of ties, during the development we returned all tied answers, and assigned partial credit in the evaluation. This system had poor performance on the training data; we chose not to submit runs with this system for the test.

3.4 Vector Space Model (VSM)

For this method, we measured the similarity of each query sentence to each sentence in the reference text, using cosine similarity of feature vectors representing the sentences, consisting of lexical, syntactic, and semantic information. This method, known as the vector space model (VSM) has been widely used in Information Retrieval to compare documents and queries [9]. We utilised the merged question/answer queries from Section 3.2 as the starting point. The query-reference sentence pair that has the highest similarity score is selected as the answer. In this case ties are rare, and we choose the answer randomly when that happens. Again, our experiments relied on different background collections, described in Section 3.6.

We made use of the following feature types:

- Lexical (LEX): We lemmatised the text using the Genia tagger [21] in order to use lemmas as well as word-forms in the feature vectors. We also tested the effects of removing function words over the development data. Our final version of this feature type used lemmas, function word removal, and special features for NUMBER and DATE classes.
- Syntactic (SYN): We extracted all possible triples from the dependency parser output (cf. Section 3.5), and each of these relationships was used as a feature of the vector (e.g. "introduce-OBJ-morphometry").
- Semantic (SEM): We used MetaMap [1] to obtain phrases and concepts that occur in the UMLS metathesaurus [10]. The mapped concepts can be marked as negative by the built-in Negex tool, and in these cases we added a "NEG_" prefix to the feature. Each concept is also associated to one of the 73 Semantic Types that form the high level ontology of UMLS (e.g. "Enzyme"). We extracted three types of features from the MetaMap output: Concept identifiers (CUI), Semantic Types (ST), and hypernyms of the original concepts (HYP). We tested different combinations of these features.

In order to build the feature vectors, we tested using both raw frequencies, and tf-idf scores. We also incorporated a thresholded pre-filter to compare background sentences to answer candidate strings only, in order to remove sentences that have large overlap with the query but little with the possible answers. Finally, we explored combining the outputs of different VSM configurations, by choosing the candidate with the highest cosine similarity value from any of a set of underlying systems.

3.5 Graph Overlap

For the dependency-graph approach, we built on prior work from the AnswerFinder QA system [14]. The idea behind this was to create a rough dependency-based analog of the word-overlap method described above. For the dependency matching, we first apply the graph-merging algorithm described in Section 3.2. We then compare the merged question-answer graph for a candidate answer with the dependency graph for every supporting sentence. The intuition was that, for

the correct answer, there should be some corresponding declarative sentence in the text denoting the answer, and the graph of this should show a high degree of overlap with the merged question-answer graph. The supporting graph with the highest similarity score is likely to have the most similar declarative content to the candidate question-answer graph, providing evidence that the potential answer may be correct. We repeat this process for each candidate answer, and the answer with most closely matching (highest-scoring) supporting sentence is marked as correct.

Dependency Parsing The pre-processing described in Section 3.1 was applied to all background text, as well as questions and answers. For the reference corpora, the POS-tags from preprocessed files supplied by the organisers was preserved; for questions and answers, no preprocessed version was provided, so we POS-tagged the questions using the biomedical POS-tagging model of ClearParser. A qualitative analysis showed that this performed poorly over questions due to differences in the tagging model, so we added a subsequent post-correction phase. If any token among the first three corresponded to a *wh*-question word such as *which*, *what* or *how*, the tag was explicitly set to be the correct tag for the token to operate as a question word, ensuring, for example, that *which* and *what* are tagged correctly (according to [19]) as *WDT*,[3] rather than *IN*, the tag they were (surprisingly) assigned more frequently.

For parsing, we used the biomedical model over the sample data, where we found it gave acceptable accuracy after the POS-tags were corrected. However, over the questions in the test data, a manual inspection of the parser outputs revealed a large number of parsing errors probably due, as in POS-tagging, to the parsing model having very few question instances in its training data from which to learn parsing features. We switched instead to the pre-trained 'Medical' model which includes clinical questions in its training data, and observed a qualitative improvement in parsing accuracy.

Scoring Graphs for Similarity After merging candidate answers with question subgraphs, we have a set of distinct dependency graphs which can be compared against the graphs obtained by parsing the reference corpus as described above. The comparison method we used was based on the graph comparison techniques from AnswerFinder [15, 14]. Specifically, we converted the ClearParser outputs to the Logical Graphs of [15], requiring some minor extensions to the AnswerFinder codebase, and then compared the logical graphs from the question-answer parses to those of the reference sentences using the implementation of the MCS algorithm for graph comparison included with AnswerFinder. Unlike [14], this work here did not have a stage of learning QA-rules from training data, due to a lack of readily available in-domain data.

[3] In some cases, *what* as a question word should be tagged as *WP*, when it acts as the the head of the noun phrase [19], but we did not allow for this possibility here.

Each candidate merged query graph was compared against every reference sentence. The raw overlap score for a given pair is the size of the largest overlapping subgraph. This score was either kept as a raw count or normalised by the lengths of the respective sentences, to avoid a bias towards longer sentences. The score for a given answer was set to the maximum similarity found from any pairing involving the answer graph. The question/answer graph with the highest value for this maximum similarity was assumed to be most likely to contain the correct answer, and the corresponding answer would then be marked as correct.

We varied the reference corpus in our experiments. In some cases, we limited the comparison to the canonical main article supplied with the question data, while in others we used text from the supplied background document collections. Due to the computationally intensive nature of the graph comparison process, as well as the limited time available for experimentation over the test set, exhaustive comparison of each candidate answer sentence with all background sentences would have been infeasible. In cases where we used the background collection, this was filtered by thresholding against the word-vector similarity score for the sentence-pair from Section 3.4. This reduction in collection size would be unlikely to erroneously omit documents, since documents with a low word overlap are unlikely to have a high degree of graph overlap.

3.6 Resources

The only external knowledge resource that we used was the Unified Medical Language System (UMLS) [10]. We applied the MetaMap system [1] to recognise UMLS concepts in the texts. In several of the runs we submitted, we took advantage of the hierarchical structure of the UMLS in order to generalise observed concepts to their hypernyms. These hypernyms were used as features in some vector space model runs.

Runs varied in terms of which texts were used as the reference corpus for the query similarity matching. In some runs, only the source document associated with the questions was utilised. Other runs considered the full background set of documents that we were provided at the outset of the experiment by the organisers. Finally, we also constructed an extra background collection of 63,000 abstracts built by querying PUBMED with the terms "Alzheimer's disease". We experimented with various combinations of these collections.

4 Results

We submitted 10 runs for the test set in the evaluation. The results we obtained for each system variant on both the training set and the evaluation test set are shown in Table 1. 'VS' refers to the vector space model, 'WO' indicates word overlap, and 'GM' refers to the graph-matching approach. The 'Wtd Acc' columns refer to the 'weighted accuracy', where if multiple values tie for the highest rank, the system receives only partial credit — the reciprocal of the

number of tied answers, instead of one. For runs using 'GM', ties were broken by arbitrarily choosing the numerically-lower answer.

The 'Backgrd' column refers to supporting documents added from the background material which were used as evidence. This could be no documents, the complete collection, or a filtered subset thresholded on the basis of vector-space bag-of-words similarity ('BW'), in which case the threshold is shown.

Run	Meth	Backgrd	Features	Sample	Test Result		
				Wtd Acc	Wtd Acc	Correct (of 40)	c@1
nicta12012	VS	all	LEX (filter 0.1)	0.38	0.16	6	0.15
nicta12074	VS	all	LEX+CUI+ST+HYP(tf-idf)	0.70	0.16	6	0.15
nicta12091	GM	–	raw counts	0.51	0.22	8	0.20
nicta12024	VS	all	Comb: nicta12012, nicta12063	0.70	0.24	9	0.23
nicta12031	VS	–	LEX+SYN (filter 0.2)	0.55	0.25	9	0.23
nicta12041	VS	–	LEX (filter 0.2)	0.90	0.23	9	0.23
nicta12053	VS	–	LEX+CUI+ST+HYP	1.00	0.22	9	0.23
nicta12063	VS	–	LEX+CUI+ST	0.90	0.25	9	0.23
nicta12082	VS	–	LEX (tf-idf)	0.70	0.24	9	0.23
–	WO	–	word overlap	0.55	*0.25*	*9*	*0.23*
–	GM	–	normalised	0.41	*0.21*	*10*	*0.25*
nicta12102	GM	BW(0.45)	raw counts	0.32	0.23	11	0.28
–	GM	BW(0.45)	normalised	0.29	*0.27*	*12*	*0.30*

Table 1. Results of the NICTA/Macquarie team runs on the 2012 QA4MRE Alzheimer task. Italicised results (with no run ID) were not submitted as official runs to the task.

5 Discussion

The performance of our methods on the test set was significantly worse than the performance of the small sample data set we had been provided with. While several methods showed high accuracy over the sample data, the results over the test set were in general not significantly better than random, if at all.

5.1 Word overlap and VSM

The results of both the word overlap system and VSM were well above the random baseline over the development data. Moreover, with rich semantic features, such as hypernyms, the VSM model reached 100% accuracy (without parameter tuning). These results suggested that VSM could perform well in the challenge, however the results over the test set were at the level of the random baseline.

An analysis of the outputs of the VSM on the test data showed that there was large variability on the answers given by the different configurations, despite the similar low performance. However, even if we had an oracle system over the

outputs of these 8 systems, it would only achieve 62.5% accuracy. The systems performed particularly badly over the first document (22506010), with an oracle accuracy of only 30%. We manually analysed those errors. We found that one of the main sources of error seemed to be the selection of distractors, which were often terms with high frequency in the reference document; this misled our naive classifiers, which have minimal awareness of structure, and led them to retrieve large numbers of sentences with high similarity scores.

Manual analysis also showed that relevant sentences would usually appear towards the top of the ranking, but below other candidates with higher weight. This did not happen over development data, where one of the candidate sentences usually stood out. This suggests that this approach may be useful where the questions are more straightforward, like the ones given in the development set, or even as a initial filter in a harder challenge such as represented by the test data; but it is not effective as a complete solution. These results are consistent with [7], where VSM is the best performing system for one of the two target tasks (TREC 11 QAD, built semi-automatically), but failed to perform well for the other (GNU Library Manual, built manually).

Examining the different variants of VSM, adding the background collection clearly increased the confusion; the best results were obtained using the reference document only. Comparing the performance of different feature sets, it is difficult to make strong conclusions. Syntactic and semantic features seem to be generally useful, though inclusion of hypernyms reduced performance on the test data. As mentioned before, ties are rare, and choosing not to answer in these cases has minimal effect on the final scores.

5.2 Graph Matching

Over the sample data, the graph-matching approach performed appreciably better than random in at least some configurations, although the figures were much less promising than the VSM methods. As with the word-based methods, the accuracy dropped noticeably over the larger test set, although the magnitude of this drop was smaller, largely because the performance over the sample set was not as high to begin with. While the best accuracy was obtained over a graph matching run, much of this may be attributable to chance. An error analysis examining the eight questions which were answered correctly in the 'nicta12102' run but not in the VSM 'nicta12053' run showed that only two had genuinely selected a sentence which provided good evidence for a single postulated answer, while the remainder had the question correct by chance, either because a spurious sentence match lead to the correct answer anyway, or because there was a tie between two or more sentences and the best answer was arbitrarily selected.

There were other interesting differences between the sample data and the test set. In post-submission experiments, we found that the normalisation for sentence length had a positive effect over the test data, even though it was detrimental over the small set of sample data. Similarly adding in the filtered background collection caused changes in different directions over the test and development sets.

There were some easily repairable deficiencies in the graph-matching approach which were made obvious during more detailed analysis. The handling of numbers in our system was suboptimal, which is particularly a problem when the answers to a question are mostly numeric. The logical graphs which we check for overlaps use the lemma as the node identifier, which is a sensible approach for most words. However, the lemmas produced by ClearParser convert all numbers and contiguous sequences of digits to a single digit '0'. So '10' and '283.0' would both be mapped to '0', and the string 'P436Q' would be converted to 'p0q', in all cases losing potentially valuable information from a QA perspective. In future work, we will experiment with a different tool, e.g. the BioLemmatizer [11].

Another deficiency of the graph-matching approach was that there was often insufficient distinction between the different answers, if it happened that the answers were not part of a matching subgraph for any of the evidence sentences. In these cases, a fallback strategy (such as using a VSM approach) could have helped somewhat. Another option would have been to not postulate analyses for cases where there were multi-way ties. In post-hoc analysis, we investigated what the outcome would have been if we had refused to pick an answer when there were three or more answers tied for the highest score. For the graph matching variant with the background collection, this would have meant only answering 17 questions, but the c@1 score would have decreased from 0.30 to 0.23 (with a similar drop over the sample data), indicating that we are probably benefiting from chance co-occurrences in the displayed test score.

While the graph matching algorithm itself was fairly well-developed, the overall pipeline was fairly simplistic. It is possible that it would have been beneficial to to use a more sophisticated approach such as the rule-learning approach of [14], but this was difficult due to a lack of in-domain training data.

6 Conclusion

In this paper, we explored the application of similarity-based methods to a multiple-choice question answering task in the biomedical domain. The somewhat unusual multiple-choice nature of this question answering task meant that we were able to attempt interesting transformations of questions by inserting answer graphs into the question graphs, and compare those merged graphs to sentences in a reference document or collection. Our initial experiments on the development data showed promise particularly for the similarity measurements utilising the vector space model with a combination of lexical and semantic features; however these results did not generalise to the test data, where we saw consistently lower performance and no evidence that one feature type or similarity matching strategy is consistently better than another. The inclusion of the background document collection did not in general seem to help, except possibly when filtered to identify the most likely relevant subset of that background. Together, our results suggest that the test set of questions and answers required a more sophisticated solution for answer selection than we developed on the basis of the sample data.

References

1. Aronson, A.R.: Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. In: AMIA Annual Symposium Proceedings. pp. 17–21. Washington DC (2001)
2. Brill, E., Lin, J.J., Banko, M., Dumais, S., Ng, A.: Data-intensive question answering. In: Voorhees, E.M., Harman, D.K. (eds.) Proceedings TREC 2001 (2002)
3. Cao, Y.g., Liu, F., Simpson, P., Antieau, L., Bennett, A., Cimino, J.J., Ely, J.W., Yu, H.: AskHERMES: An online question answering system for complex clinical questions. Journal of biomedical informatics 44(2), 277–88 (Apr 2011), http://www.ncbi.nlm.nih.gov/pubmed/21256977
4. Celikyilmaz, A., Hakkani-Tur, D., Tur, G.: LDA based similarity modeling for question answering. In: Proceedings of the NAACL HLT 2010 Workshop on Semantic Search. pp. 1–9. Association for Computational Linguistics (2010), http://dl.acm.org/citation.cfm?id=1867768
5. Choi, J.D., Palmer, M.: Getting the most out of transition-based dependency parsing. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 687–692. Association for Computational Linguistics, Portland, Oregon, USA (June 2011)
6. Demner-Fushman, D., Lin, J.J.: Answering clinical questions with knowledge-based and statistical techniques. Computational Linguistics 33, 63–103 (2007)
7. Emms, M.: Variants of tree similarity in a question answering task. In: Proceedings of the Workshop on Linguistic Distances. pp. 100–108. LD '06, Association for Computational Linguistics, Stroudsburg, PA, USA (2006), http://dl.acm.org/citation.cfm?id=1641976.1641989
8. Ferrández, O., Muñoz, R., Palomar, M.: TE4AV: Textual entailment for answer validation. In: Proceedings NLP-KE. pp. 1–8 (2008)
9. Gerard Salton, A.W., Yang, C.S.: A vector space model for automatic indexing. In: Communications of the ACM, 18. pp. 613–620 (1975)
10. Lindberg, D., Humphreys, B., Mccray, A.: The unified medical language system. Methods Inf Med 32(4), 281–291 (1993), retreived file d2007.bin from http://www.nlm.nih.gov/cgi/request.meshdata on June 20, 2007.
11. Liu, H., Christiansen, T., Baumgartner, W.A., Verspoor, K.: BioLemmatizer: a lemmatization tool for morphological processing of biomedical text. Journal of Biomedical Semantics 3, 3 (2012)
12. Magnini, B., Negri, M., Prevete, R., Tanev, H.: Is it the right answer? exploiting web redundancy for answer validation. In: Proceedings ACL. pp. 425–432 (2002)
13. Moldovan, D.I., Harabagiu, S.M., sca, M.P., Mihalcea, R., Goodrum, R., Girju, R., Rus, V.: Lasso: A tool for surfing the answer net. In: Voorhees, E., Harman, D.K. (eds.) Proceedings TREC 8 (2000)
14. Mollá, D.: Learning of graph-based question answering rules. In: Proceedings of HLT/NAACL 2006 Workshop on Graph Algorithms for Natural Language Processing. pp. 37–44. (2006)
15. Mollá, D., van Zaanen, M.: Learning of graph rules for question answering. In: Proceedings of the 2005 Australasian Language Technology Workshop (2005)
16. Niu, Y., Hirst, G., McArthur, G., Rodriguez-Gianolli, P.: Answering clinical questions with role identification. In: Proc. ACL, Workshop on Natural Language Processing in Biomedicine (2003), http://citeseer.ist.psu.edu/581532.html
17. Paşca, M.A., Harabagiu, S.M.: High performance question answering. In: Proc. SIGIR'01. ACM, New Orleans, Luisiana, USA (2001), http://citeseer.ist.psu.edu/pasca01high.html

18. Peñas, A., Rodrigo, A.: Testing the reasoning for question answering validation (draft). Journal of Logic and Computation 18(3), 459474 (2006), http://logcom.oxfordjournals.org/content/18/3/459.short
19. Santorini, B.: Part-of-speech tagging guidelines for the penn treebank project (3rd revision). Tech. Rep. 570, University of Pennsylvania (1990)
20. Soubbotin, M.M.: Patterns of potential answer expression as clues to the right answers. In: Voorhees, E.M., Harman, D.K. (eds.) Proceedings TREC 2001. NIST (2002)
21. Tsuruoka, Y., Tateishi, Y., Kim, J.D., Ohta, T., McNaught, J., Ananiadou, S., Tsujii, J.: Developing a robust part-of-speech tagger for biomedical text. In: Advances in Informatics - 10th Panhellenic Conference on Informatics. pp. 382–392. Volas, Greece (2005)
22. Verspoor, K., Cohen, K.B., Lanfranchi, A., Warner, C., Johnson, H.L., Roeder, C., Choi, J.D., Funk, C., Malenkiy, Y., Eckert, M., Xue, N., Jr., W.A.B., Bada, M., Palmer, M., , Hunter, L.E.: A corpus of full-text journal articles is a robust evaluation tool for revealing differences in performance of biomedical natural language processing tools (in press). BMC Bioinformatics (2012)
23. Voorhees, E.M.: The TREC question answering track. Natural Language Engineering 7, 361–378 (2001)
24. Yu, H., Cao, Y.g.: Automatically extracting information needs from ad hoc clinical questions. In: AMIA Annual Symposium Proceedings. vol. 2008, p. 96. American Medical Informatics Association (2008), http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2655957/