Selecting answers with structured lexical expansion and discourse relations

LIMSI's participation at QA4MRE 2013

Martin Gleize[1], Brigitte Grau[1,3], Anne-Laure Ligozat[1,3], Van-Minh Pho[1,2], Gabriel Illouz[1,2], Frédéric Giannetti[1], and Loïc Lahondes[1]

LIMSI-CNRS, rue John von Neumann, 91403 Orsay cedex, France,
Université Paris-Sud, 91400 Orsay, France
ENSIIE, 1 Square de la Résistance, 91000 Evry, France
firstname.lastname@limsi.fr

Abstract. In this paper, we present the LIMSI's participation to QA4MRE 2013.We decided to test two kinds of methods. The first one focuses on complex questions, such as causal questions, and exploits discourse relations. Relation recognition shows promising results, however it has to be improved to have an impact on answer selection. The second method is based on semantic variations. We explored the English Wiktionary to find reformulations of words in the definitions, and used these reformulations to index the documents and select passages in the Entrance exams task.

Keywords: question answering, index expansion, discourse relation, question classification

1 Introduction

In this paper we present the LIMSI's participation to QA4MRE 2013. We decided to experiment two kinds of methods. The first one focuses on complex questions, such as causal questions, and exploits discourse relations. We created a question typology based on the one proposed by QA4MRE organizers, and linked it to the type of relation expected between the answer and the question information. In order to detect these relations in the texts, we wrote rules based on parse trees and connectors.

The second method is based on semantic variations. We explored the English Wiktionary to find reformulations of words in their definition, and used these reformulations to index the documents and select passages in the Entrance exams task.

The paper is organized as follows: in section 2, in order to give an overview of the methods we developed, we present the general architecture of our system. Section 3 details question analysis. In relation to question classification, section 4 presents discourse relation recognition. We then present the two methods for

passage selection and answer ranking in section 5. Selection of answer according to question category and discourse relation is described in section 6 before presenting our experimentations and results in section 7.

2 System overview

Figure 1 presents the different modules developed for QA4MRE tasks, which form the QALC4MRE system.

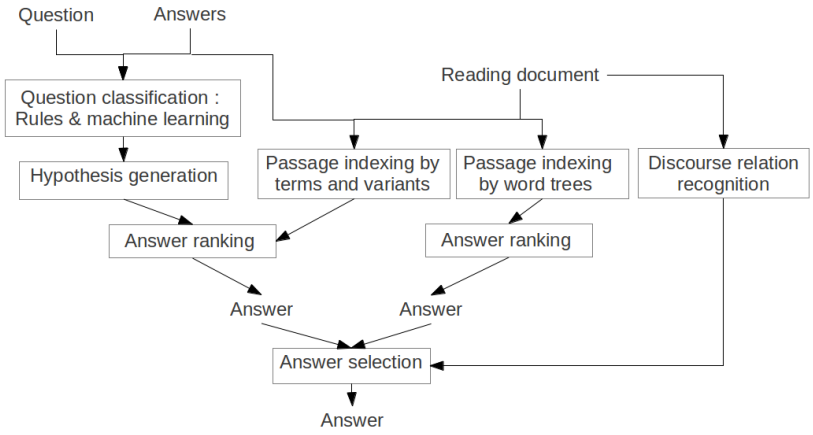

Fig. 1. Architecture of QALC4MRE

Reading documents used in main task and Alzheimer task are generally scientific papers and variations between words in questions and answers and words in the relevant passages of text are often based on paraphrases. Thus, these kinds of variations are handled by rules that take into account morphological, syntactic and semantic variants [1]. In entrance exams, there are more distant semantic variations between each set of words, such as hypernymy or causal relation for example. Thus, we tackle these problems by creating paths based on following dictionary definitions of question words towards document words. We developed two modules for passage retrieval: terms and variant indexing and word tree indexing. Question analysis is the same for all tasks. From the question parse trees, we generate hypotheses by applying rules written manually. For determining question types, we reuse existing question classification modules.

Complex types of questions are associated to discourse relations in documents which have to hold with the answer. In order to recognize these relations in documents, we wrote rules based on parse trees of document sentences.

Answers are ranked according to different measures. For answers to complex questions, if a corresponding relation if found on a candidate answer in the top passages, this candidate is returned.

3 Question analysis

The aim of the question analysis module is to determine the question category. As we decided to focus on discourse relations, we adapted our existing systems to detect the kind of discourse relation between the answer and the question words.

We kept the Factoid questions subclasses based on the expected answer type in terms of named entity type: person, organization, location, date...

We added the following classes, according to the task guidelines and to [2] taxonomy:

- causal/reason: there is a cause-consequence relation between the answer and question information.
  Why cannot bexarotene be used as a cure for Alzheimer's disease?
- method/manner: the question asks for the way something happens.
  How do vitamin D and bexarotene correlate?
- opinion: the question asks about the opinion about something.
  What was Cramer's attitude towards the music of Bach?
- definition: the expected answer is the definition, an instance or an equivalent of the question focus.
  What is a common characteristic for the neurodegenerative diseases?, Give two symptoms of dementia.
- thematic: the question asks for an event at a given time.
  What happened during the meal after the family had all taken their new seats?

We used two existing question analysis modules: one based on syntactic rules [1] and one based on machine learning classification [3].

The first module parses the question with the Stanford Parser [4] which provides a constituency parse tree. Then, syntactic rules determine the question class by recognizing a syntactic pattern with Tregex and Tsurgeon [5]. For example, for the question *Which singer made a hit record whose accompaniment was entirely synthesised?*, the rules detect the interrogative pronoun *which* and that it possesses a son *son* in the parse tree; this noun is compared to a list of triggers and is recognized as a trigger of the *person* question class.

After the evaluation, we evaluated the results of this module on the test sets of QA4MRE 2013. 73% of questions were correctly classified. Most errors were due to question formulations which had not been taken into account, such as

boolean questions, and some of them to misclassifications (for example *What is the cause...* was incorrectly classified as a factoid question).

The second module is based on an SVM based classifier using the LibSVM [6] tool with default parameters. The classifier was trained on [2] fine-grained question taxonomy, with each question category considered as a class. The features used are n-grams (n ranging 1..2) of words, lemmas and parts-of-speech (determined by the TreeTagger [7]), as well as the trigger lists of the first module and a regular expression based recognition of abbreviations. This module obtained 0.84 precision on [2] test corpus.

We also evaluated this module on QA4MRE 2013 test sets, and it obtained 0.85 correct classification. The main kinds of error are the misclassification of factoid question into definition questions and the absence of the *opinion* class in the hierarchy.

4 Discourse relation recognition

Our present work was a first attempt to take into account discourse relations in order to study if it was possible to relate them to question categories and thus to provide a supplementary criterion for selecting an answer. Thus we decided to model the recognition of some of them by rules in a first time, as we did not have an annotated corpus.

4.1 List of relations

We took into account the following four binary relations:

- Causality, to be related to causal/reason questions
- Opinion, to be related to opinion questions
- Definition, to be related to definition questions
- Example, to be related to questions asking for a concept in factual questions, such as which animal ... ?

Being binary relations, each of these relations presents two components which we detail below:

- Causality is composed of a cause and a consequence.
  [He would not provide his last name]$_{Csqce}$ *[because]*$_{Mark}$ *[he did not want people to know he had the E. coli strain.]*$_{Cause}$
- Opinion is composed of a Source and a Target.
  [Some users of the Apple computer]$_{Src}$ *[say]*$_{Mark}$ *[it smells sickening.]*$_{Trgt}$
- Definition is composed of a Concept and an Explanation.
  [a Rube Goldberg machine]$_{Cpt}$ *[is]*$_{Mark}$ *[a complicated contraption, an incredibly over-engineered piece of machinery that accomplishes a relatively simple task]*$_{Exp}$
- Example is composed of a Concept and a List.
  [other endangered North American animals]$_{Cpt}$ *[such as]*$_{Mark}$ *[the red wolf and the American crocodile.]*$_{List}$

Causes and consequences of causality relations can be found between two clauses or between phrases in a sentence; they can also be found in consecutive sentences. Thus we defined rules that recognize each of the two members separately.

Opinion relations were restricted to reported discourse.

Definition relations gather all types of clause that helps defining or specifying a precise concept. These can be embodied as appositive, as in *the tiger, the largest of all the big cats*, reformulation, as in *polar regions known as the cryosphere* or a canonical model of definition, as in *Rickettsia mooseri is a parasite of rats.*

Example relations encompass any instance of a larger concept. The expected result is a list of n instances, as to be found in *luxuries such as home air conditioning and swimming pools* or *great Black players like Michael Jordan or Elgin Baylor.*

4.2 Relation extraction

Regular expressions were defined on the syntactic trees of sentences. They were obtained by parsing a significant portion of the background collection of QA4MRE 2012 using Stanford Parser. We first defined a set of discriminating clue words ($_{Mark}$) for each of the aforementioned relations based on the selected corpus. We then developped a series of syntactic rules implemented according to the Tregex formalism [5] which allows to create *tgrep*-type patterns for matching tree node configurations. Constraints in rules are defined on left, right, child and parent nodes of the $_{Mark}$. They are about expected types of syntagms and POS categories.

In total, we defined a set of 42 rules to extract the different types of relations.

To evaluate the extraction of rules, we manually annotated the four texts of each thematic of the evaluation for the Main Task 2013 and the nine texts for English Exams Task. Twenty-five annotated documents were thus annotated, containing 162 causality relations, 53 opinions, 114 definitions and 57 examples, for a total of 416 relations.

We then compared the manual annotation to the one made by our system on these documents. To achieve this, we categorized found relationships in two types. If the relationship annotated by hand is strictly the same as the relationship found automatically, i.e. same type and same related members, this relationship is classified as "exact". If the relationship is incomplete, i.e. if there are missing or extra words in the related members, the relationship is classified as "loose". We will consider these kinds of relations as correct in a lenient evaluation. If the type of the relationship automatically annotated is false, it is "incorrect". Finally, we compute a fourth counter: the number of "missed" relationships, calculated as the difference between the number of manual annotations and the sum of the number of "correct" and "loose" relationships.

Results are given in tables 1 and 2. We can see that we obtain a very good precision in the lenient evaluation, which shows that relation types are well identified. As expected, recall is lower, but remains reasonable.

-	Correct	Loose	Incorrect	Missed
Causality	39	48	4	86
Opinion	23	19	12	50
Definition	50	13	14	51
Examples	22	10	2	14
Total	134	90	32	201

Table 1. Recognition of relations

-	Strict Precision	Strict Recall	Lenient Precision	Lenient Recall
Causality	0.429	0.225	0.956	0.503
Opinion	0.426	0.250	0.778	0.457
Definition	0.649	0.439	0.818	0.553
Examples	0.647	0.478	0.941	0.696
Total	0.523	0.315	0.875	0.527

Table 2. Recognition of relations in terms of precision and recall

5 Passage and answer weighting

5.1 QALC4MRE strategy

We apply the weighting scheme of [1] for sentences according to the question words and answers, named `P_REP`, the overlapping of weighted common words between a sentence and an answer, `TERp` and `treeEdit` distances between a sentence and an hypothesis.

For selecting answers, we give priority to passage weight, and secondary to answer weight, and define several combinations of these weights:

- the most frequent answer in the n top sentences. In case of equality of different answers, the answer in the best sentence is selected, and if several candidate answers remain in the same sentence, the answer with the best weight is selected. This selection scheme is named `freqTop`.
- the most frequent answer in the n top sentences which contain a candidate answer, with the same options in case of answer equality, named `maxS`.
- the best answer in the n top sentences, named `maxSTop`.

5.2 Dictionary-based passage retrieval

In a question answering system, passage retrieval aims at extracting the short text excerpt most likely to contain the answer from a relevant document. For the most realistic questions, direct matching of the surface form of the query and text sentences is not sufficient. As one of the most challenging and important processes in a QA system, passage retrieval would thus benefit from a more semantic approach.

We propose a passage retrieval method focusing on finding deep semantic links between words. We view a dictionary entry as a kind of word tree structure:

taken as a bag of words, the definition of a word makes up its children. Then the words in the definition of a child are this child's own children, and so on. From this point forward we will designate as *words* only lemmas from verbs, nouns, adjectives, adverbs and pronouns that are not stop-words. We assume a single purely textual document. *Document words* are words in the document.

Indexing the document This document pre-processing phase builds an index off of all the words in the document and their descendants in a given dictionary. This is similar to the index expansion of Attardi et al. [8], except we use dictionaries and not background documents. An entry in this index is composed of:

- a word w (the key in the index)
- a list $Inv(w)$ of pairs (index of a sentence containing w in the document, index of w in this sentence): this is standard inverted indexing.
- the tree $T(w)$ of w's word descendants (implemented as pointers to the entries of w's children)
- a list $Anc(w)$ of *document word ancestors*, pairs (w_2 document word, d depth) such that: $w_2 \in Anc(w)$ with depth d iff we can find w in w_2's tree at depth d (For example: at depth 2, we look at children of children of w_2 and w is among them).

To index a given word w, we check if w isn't already in the index (otherwise we build and add the entry), and we update the entry recursively, using an auxiliary children update procedure UPDATE in the main procedure INDEX(w, d, doc_ancestor):

1. w as key
2. if w is a document word:
   (a) add to $Inv(w)$ the pair (index of S, index of w in S).
   (b) add $(w, 0)$ to $Anc(w)$. Indeed, w is a document word, and he's the root of its tree (the only node of depth 0).
3. build $T(w)$ with an update procedure UPDATE(w, $d_{\max}$, doc_ancestor), which we define in the following.

In dictionaries, traversing all the words in a definition tree might not terminate. There are cycles: it can happen than the word itself appears in the definition of words of its own definition. So we choose to explore at most $d_{\max}$ levels of depth when building $T(w)$ for any w.
Let's now define UPDATE(w, d, doc_ancestor), which updates $T(w)$:

1. look up the definition of w in the dictionary. If not found we don't touch $T(w)$.
2. run INDEX(w_c, $d-1$) if needed ($d > 1$ and w_c not indexed), for each child w_c in the definition.
3. store the pointers to words of the definition in $T(w)$.

4. add $(doc\_ancestor, d_{\max} - d)$ to each $Anc(w_c)$.

To build the complete index of the document, we simply run INDEX(w, $d_{\max}$, w) for each w of each sentence (we use StanfordCoreNLP for tokenization and tagging [9]). This is the basis of our indexing, bar minor details of implementation (re-indexing in case we need to explore an indexed word at a greater depth, handling of multiple senses and POS-tags, ...)

Passage retrieval We first consider words of the query, then use the index to score their relevance, and finally compute a density-based sliding window ranking function to retrieve passages.
For each word w_q in the query, we run a version of INDEX(w_q, $d_{\max}$, NIL) which does not update document ancestors $Anc(w)$ (as the word of the query isn't truly a document word). In $T(w_q)$, we find descendants w of w_q which have been previously built during the indexing phase and thus have an non-empty $Anc(w)$, their document word ancestors, which are essentially the document words that initiated the access to w in the dictionary. We can compute a similarity between w_q and those document words, therefore rating the relevance of document words relatively to the query word:

$$Sim(w_q, (w_{doc\_anc}, d)) = idf(w_{doc\_anc}) \times \text{base}^{-(d_{min}+d)} \tag{1}$$

$$d_{min} = \min_{w_c \in T(w_q) \text{ at depth } d_c | w_{doc\_anc} \in Anc(w_c)} (d_c) \tag{2}$$

We choose *base* depending on how strongly we want to penalize words as we go deeper in the tree. We found $base = 2$ to be a good start, but the final system uses the number of children at the depth of the closest child containing w_{doc_anc} in Anc. The intuition is that the more words used in the definition of w, the less confident we are that each definition word is semantically related to w. We compute the similarity for each w_q in the query and each w_{doc} in document word ancestors and sum over the w_q to obtain a relevance score for the document word:

$$Relevance(w_{doc}) = \sum_{w_q \in \text{ query}} (\max_d Sim(w_q, (w_{doc}, d))) \tag{3}$$

Finally, we select candidate passages with a sliding window of 3 consecutive sentences, and rank them using a similar method to SiteQ's density-based scoring function described in [10], using *Relevance* as the weight of keywords.

6 Answer selection related to discourse relation

To select an answer which takes into account question category and discourse relations, we combine weights and discourse relations of the passages. First, we filter relations according to the category of the question and presence of the answer associated with the passage in the relations. Only relations whose type is the same as the category of the question and containing an answer are kept.

Then, passages are sorted according to their weights. Among the top n passages, if any of them has a relation, the answer associated with the best weighted passage is selected. Otherwise, we consider only passages containing relations and select the answer associated with the best of them.

7 Results

7.1 Main task and Alzheimer task

Table 3 presents results obtained on the QA4MRE2012 corpus, for the different selection scheme presented section 5.1, with the number n of top sentences set to 5 after experiments. The strategy described in 6 did not lead to different results, as either question category was not correctly identified or the associate relation was not recognized in the correct passage.

	Alzheimer Task			Main Task		
	`freq`	`freqTop`	`maxSTop`	`freq`	`freqTop`	`maxSTop`
`P_REP`	10 / 0.25	14 / 0.367	-	60 / 0.382	56 / 0.370	**62 / 0.395**
`TERp`	11 / 0.288	**15 / 0.393**	9 / 0.225	50 / 0.318	50 / 0.330	51 / 0.325
`treeEdit`	12 / 0.315	**15 / 0.393**	**14 / 0.35**	52 / 0.331	52 / 0.343	53 / 0.337
baseline	8 / 0.2			32 / 0.2		

Table 3. Results on the 2012 corpora in term of number of right selected answers.

We can see that, while textual entailment distances between an hypothesis and a sentence are useful to select an answer in Alzheimer task, they are overcome by lexical overlap weighting in the main task. This can be due to differences in answer length in the two tasks: shorter answers in Alzheimer task favour measures based on sentence structure.

We obtained analogous results on the 2013 evaluation for Alzheimer task, best c@1 is 0.42 for `treeEdit` combined with `freqTop`, while results on the main task are lower with a best c@1 at 0.28 with the combination `P_REP` with `maxS`. It may be due to new kinds of questions introduced this year, and the new kind "do not know" of answer.

7.2 Entrance exam task

The form of the task is essentially the same as the main task. Multiple-choice questions are taken from reading tests of Japanese university entrance exams. A crucial difference from the other QA4MRE tasks is that background text collections are not provided.

Given the difficulty of the questions and the lack of background knowledge, passage retrieval quickly appeared as a strong bottleneck for any question answering system attempting to solve the task. That is why we decided to design

the dictionary-based lexical expansion described in 5.2 and use Simple English Wiktionary [11] as the dictionary. Simple English Wiktionary is a collaborative dictionary written in a simplified form of English, primarily for children and English learners. Its definitions are clear, concise and get to the essence of the word without superfluous details, and seem fitted to acquire the "common sense knowledge" we need to solve this task [12].
We submitted a run at QA4MRE 2013 which used only this passage retrieval system and very simple heuristics to choose an answer. The results were worse than the random baseline, due to bugs in the early implementation and the discriminating roles a passage retrieval system alone cannot fill, as we will see in the following. We instead present the evaluation of our system for the sole task of passage retrieval, on the 9 reading tests (46 questions) of the test set, following Tellex's quantitative evaluation methodology [13]. We first annotated passages of the test set (which 2-to-4-sentence passage must be read to answer the question) to create a gold standard. We found quite straight forward to limit those annotations to contiguous passages, with only 2 questions needing disjoint passages. We then implemented several runs:

- MITRE as a weak baseline: simple word overlap algorithm [14]
- SiteQ as a strong baseline: sentences are weighted based on query term density [10], and include keyword forms such as lemmas, stems, and synonyms/hyponyms from WordNet synsets.
- SI($d_{\max}$), our Simple English Wiktionary-based indexing system, parameterized by $d_{\max}$

We used the following measures:

- MRR: mean reciprocal rank
- p@n: number of correct passages found in the top n
- nf: number of correct passages which weren't found at all

Results are shown in table 4. Our system outperforms both baselines significantly on all types of tasks and measures. The difference is most noticeable when the systems do not have access to choices of answers, which is really what we seek for the broader view of question answering. What is also interesting is the increase in performance for SI as we increase the maximum depth of search in the dictionary. This seems to confirm that Simple English Wiktionary fits this task well and that our score functions scale correctly with the amount of knowledge that it provides. Furthermore, although the question paired with the correct answer seems to yield a more reliable passage selection compared to with an incorrect answer, it is not by much, so it is unlikely that we could differentiate right and wrong answers by only looking at the passages they yield. It can be explained by the relatively high difficulty of the test: no answer choice seems completely absurd and is always related in some way to the relevant passage in the text. This confirms the well-known necessity of deeper answer processing to make the final call, which our earliest run attempt lacked.

Algorithm	MRR	p@1	p@3	p@5	p@10	nf
Question alone						
MITRE	0.215	0.13	0.20	0.26	0.37	**0.13**
SiteQ	0.337	0.22	0.39	0.52	0.61	0.37
SI(1)	0.355	0.22	0.43	0.59	0.69	0.28
SI(2)	0.392	0.24	**0.46**	**0.63**	**0.76**	0.20
SI(3)	**0.420**	**0.28**	**0.46**	**0.63**	0.74	**0.20**
Question + Correct answer						
MITRE	0.320	0.20	0.33	0.49	0.50	**0**
SiteQ	0.506	**0.37**	0.57	0.65	0.89	0.07
SI(3)	**0.523**	0.35	**0.65**	**0.74**	**0.93**	**0.07**
Question + Incorrect answer						
MITRE	0.254	0.14	0.22	0.31	0.53	**0**
SiteQ	0.466	0.32	0.54	0.62	0.81	0.09
SI(3)	**0.480**	**0.33**	**0.57**	**0.72**	**0.83**	0.15

Table 4. Evaluation of passage retrieval on QA4MRE 2013 Entrance exam task

8 Conclusion and perspectives

This paper describes different experiments we conducted for QA4MRE 2013. We worked on two problems. The purpose of the first one was answering complex questions by recognizing discourse relations. The categorization of questions shows very good results while discourse relation recognition results allow us to see that this approach merits further consideration. Thus we will work on the improvement of this module and the integration of this criterion for selecting an answer. The second problem we studied was passage retrieval, especially for answering entrance exams, as semantic distance between questions, answers and text are important. We proposed indexing passages with expansion of question and answer words computed by accounting for recursive definition of words in a dictionary. This module shows good results. We now have to evaluate this approach on the other tasks and improve answer selection within best passages.

References

1. Grau, B., Pho, V.M., Ligozat, A.L., Ben Abacha, A., Zweigenbaum, P., Chowdhury, F.: Adaptation of limsi's qalc for qa4mre. In: CLEF 2012 Working notes on QA4MRE. (2012)
2. Li, X., Roth, D.: Learning Question Classifiers. In: COLING'02. (2002)
3. Ligozat, A.L.: Question classification transfer. In: ACL 2013. (2013)
4. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1. ACL '03, Stroudsburg, PA, USA, Association for Computational Linguistics (2003) 423–430
5. Levy, R., Andrew, G.: Tregex and Tsurgeon: tools for querying and manipulating tree data structures. In: Proceedings Fifth international conference on Language Resources and Evaluation (LREC 2006). (2006)

6. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2** (2011) 27:1–27:27 Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.
7. Schmid, H.: Improvements in Part-of-Speech Tagging with an Application to German. In: Proceedings of the ACL SIGDAT-Workshop. (1995)
8. Attardi, G., Atzori, L., Simi, M.: Index expansion for machine reading and question answering. In: CLEF (Online Working Notes/Labs/Workshop). (2012)
9. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, Association for Computational Linguistics (2003) 173–180
10. Lee, G.G., Seo, J., Lee, S., Jung, H., Cho, B.H., Lee, C., Kwak, B.K., Cha, J., Kim, D., An, J., et al.: Siteq: Engineering high performance qa system using lexico-semantic pattern matching and shallow nlp. In: Proceedings of the Tenth Text REtrieval Conference (TREC 2001). Volume 442. (2001)
11. Wikimedia Foundation: Simple english wiktionary
12. Gleize, M., Grau, B.: Limsiiles: Basic english substitution for student answer assessment at semeval 2013. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013), Atlanta, Georgia, USA, Association for Computational Linguistics (June 2013) 598–602
13. Tellex, S., Katz, B., Lin, J., Fernandes, A., Marton, G.: Quantitative evaluation of passage retrieval algorithms for question answering. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, ACM (2003) 41–47
14. Light, M., Mann, G.S., Riloff, E., Breck, E.: Analyses for elucidating current question answering technology. Natural Language Engineering **7**(04) (2001) 325–342