LIMSI-CNRS@ CLEF 2014: Invalidating Answers for Multiple Choice Question Answering.

Martin Gleize, Anne-Laure Ligozat, Brigitte Grau

LIMSI-CNRS@CLEF 2014: Invalidating Answers for Multiple Choice Question Answering

Martin Gleize[1,2], Anne-Laure Ligozat[1,3], and Brigitte Grau[1,3]

[1] LIMSI-CNRS, Rue John von Neumann, 91405 Orsay CEDEX, France
[2] Université Paris-Sud, Orsay
[3] ENSIIE, Evry
gleize@limsi.fr, annlor@limsi.fr, bg@limsi.fr

Abstract. This paper describes our participation to the Entrance Exams Task of CLEF 2014's Question Answering Track. The goal is to answer multiple-choice questions on short texts. Our system first retrieves passages relevant to the question, through lexical expansion involving a structured use of the Simple English Wiktionary and WordNet. Then it extracts predicate-argument structures (PAS) from each answer choice and aligns them to PAS found in the passages retrieved in the first step. Finally, manually crafted rules are applied to those alignments to try to invalidate answer choices. If enough answer choices are thus invalidated, we make a decision on the remaining answer choices based on their alignment scores with the passages. We submitted several runs in the task, only one of which reached the random baseline (c@1 of 0.25). In the last section, we provide an analysis of the differences between our relatively good results obtained on trial data and the poor performance of our test run.

Keywords: Question Answering, Passage Retrieval, Textual Entailment

1 Introduction

The task focuses on the reading of single documents and identification of the correct answer to a question from a set of possible answer options. The identification of the correct answer requires various kinds of inference and the consideration of previously acquired background knowledge. Japanese University Entrance Exams include questions formulated at various levels of complexity and test a wide range of capabilities. The challenge of "Entrance Exams" aims at evaluating systems under the same conditions humans are evaluated to enter the University. Previously the evaluation campaign Question Answering For Machine Reading Evaluation (QA4MRE at CLEF) [8] focused on multiple-choice questions designed to evaluate computer systems, but this new task takes on challenges typically offered to humans. It naturally translates into more complex inference phenomena to solve [2], and thus usually lower performance of systems: QA4MRE 2013's best run [1] on the Main task outperformed by a large margin its counterpart on the Entrance Exams pilot task (c@1 of 0.59 on 5-choice questions, compared to 0.42 on 4-choice questions) [6].

2 System Architecture

The overarching goal of our system is to essentially invalidate as many incorrect answer choices as possible. Although we also introduce some elements of validation of the correct answer, our study of the trial question set revealed that we might have a better chance of finding the right answer by elimination of the wrong candidates. The architecture of our multiple-choice question-answering system is described in Figure 1. Its pipeline is composed of mainly four modules: preprocessing, passage retrieval, predicate-argument extractor and validation/invalidation. The remaining of this section is dedicated to the detailed description of those modules.

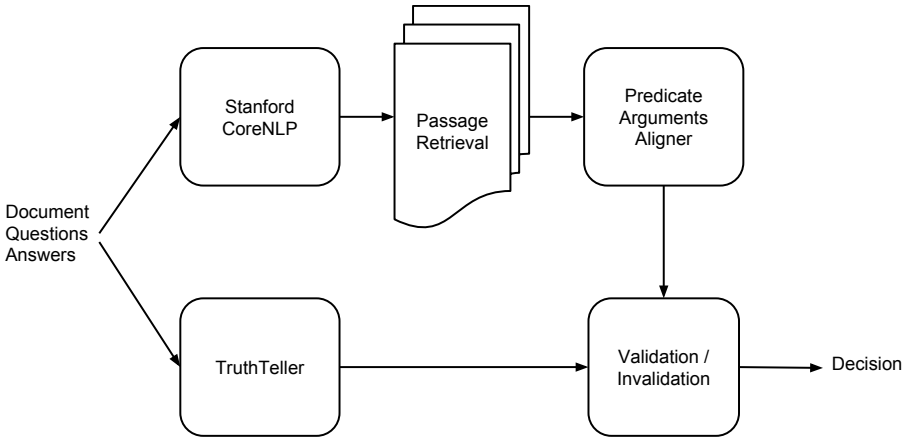

Fig. 1. System Architecture

2.1 Preprocessing

We use Stanford CoreNLP as our main Natural Language annotation tool. Each sentence from the document, questions or answer choices is tagged with Part-Of-Speech [9] and syntactically parsed [4]. In addition, we apply coreference resolution [7] on the whole document. We did not use the Named Entity Recognition component.

We also use TruthTeller [5], a semantic annotator that assigns truth values to predicate occurrences, on all the sentences. More details about the annotation types are available in its presentation paper, but we generally only use the Predicate Truth annotation, which is the final value assigned by TruthTeller. It is one of P (Positive), U (Uncertain) or N (Negative), and indicates whether the predicate itself is entailed by its containing sentence, in the classical sense of textual entailment.

2.2 Passage Retrieval

The passage retrieval module aims at ranking document snippets of 3 to 5 sentences by relevance to the question. Words of the question act as the query. However, it is very rare that words of the question exactly appear in the relevant passage of the document, so we have to use some form of query expansion. We use a method similar to [3]: we basically expand each query and document word with the words of their definition in the Simple English Wiktionary [10]. We can do that recursively, expanding definition words with their own definition, thus defining a kind of word tree. We compute for each (query word, document word) pair a weighted number of common words in their tree. We weigh the words according to their depth in the definition tree.
Figure 2 shows a portion of the definition trees for the words *cat* and *wolf*. The word *animal* is found in the Simple English Wiktionary definition of both words: we say that it is found at depth 1 –depth 0 being the word itself. The word *pet* on the other hand is found in the definition of *dog*, a word of the definition of *wolf*: we say that it is of depth 2. We add the depth of common words in both trees: *animal* is a common word of depth 2 and *pet* is a common word of depth 3 for the pair (*cat*, *wolf*). This roughly captures the intuition that the word *animal* is more accurate to describe common traits of a cat and a wolf rather than the word *pet*.

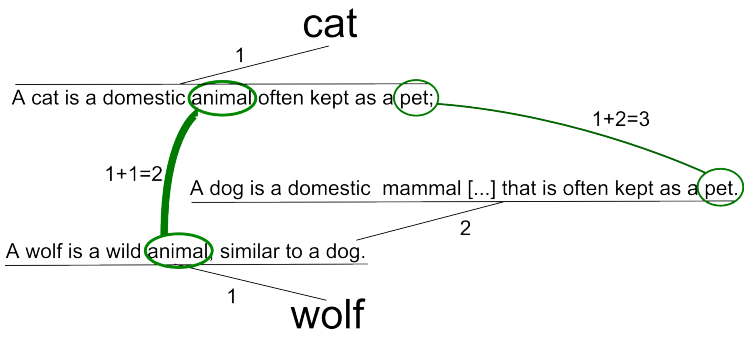

Fig. 2. Simple English Wiktionary Definition trees of *cat* and *wolf*

We enrich the definition trees with coreference information and WordNet synonyms and antonyms –to complete the coverage of the Simple Wiktionary, and weigh all the words in the tree by the IDF score of the word in the document. We ran TruthTeller on all the Simple English Wiktionary definitions, so we can propagate predicate truth values in the definition tree to compute whether a word in the tree is entailed by the root word. The propagation rules are simple: if a word is of truthvalue P or U, each of its direct definition words keeps the truth value assigned by TruthTeller, if it is N, we take the inverse TruthTeller annotation for the definition word (inverse of P is N, inverse of N is P, and inverse of U is U). We do not use predicate truth information for scoring, it will

be used in the validation/invalidation step.
With the resulting word-to-word semantic relatedness scores, we compute a 1-to-1 maximum sum alignment of the words using the Integer Linear Programming solver lpsolve. We then rank the aligned passages by alignment scores. This makes up our basic ranking system and it is also used in the predicate-argument alignment described later in the paper.
One interesting property of our data is that in general, the order of the passages in the text preserve the order of the questions they're relevant to: if question 2 is after question 1 in the reading test, then its relevant passage in the document is most likely after that of question 1. We use a simple dynamic programming algorithm –omitted here for space– to compute the best sequence of passages conserving this property, and place the passages computed this way at the top of their respective per-question list.

2.3 Predicate-arguments Aligner

Predicate-arguments Extractor Predicate-arguments structures (PAS) aim at capturing essentially "Who does what to what?" in a shallow way. We use the Stanford dependency graph from the parsed sentences and take the verbs (Penn POS tag starting with V) as predicates and we classify their dependencies between subject and arguments. We take as subject the nodes in a *nsubj* or *nsubjpass* dependency with the verb, and consider the rest as arguments. We enrich subjects and arguments with their own dependencies to not lose out on adjectives, certain numerical determiners and prepositions. When considering dependencies, we filter out determiners (with the DT tag).

Alignment of answer PAS and passage PAS With the previously described method, we extract PAS for the sentences of each answer choice. We also extract PAS for the sentences in the relevant passages obtained in section 2.2. We align answer PAS and passage PAS with the same alignment method using our semantic relatedness method and the ILP formulation, and rank the alignments by alignment score. Let us note that we align regardless of the function the word plays in the PAS: we can align predicates with subjects and arguments, and subjects with arguments.
At this point, we summarize the information available to us in those alignments. For each aligned word pair, we know:

- the function that each word plays in its respective containing PAS: subject, predicate or argument.
- a semantic relatedness score.
- truth value annotations, as annotated by TruthTeller.

2.4 Validation/Invalidation

Algorithm Our goal in this last module is to eliminate as many answer choices as possible without eliminating the right answer. Let K be the maximum number

of answer choices we are allowed to keep to take the final decision. The following algorithm computes whether we take a final decision for the current question.

```
ALL_ANSWERS := all answer choices
AnswersChoices := all answers
Passages := all relevant passages
While (|AnswersChoices| > K && |Passages| > 0) {
  Passage = Passages.pop()
  Validated = {}
  Invalidated = {}
  Foreach (AnswerChoice in AnswerChoices) {
    PASAlignments = align(AnswerChoice, Passage)
    Foreach (PASAlignment in PASAlignments) {
      if (validate(PASAlignment))
        Validated.add(AnswerChoice)
      if (invalidate(PASAlignment))
        Invalidated.add(AnswerChoice)
    }
  }
  ToRemove := {}
  if (|Invalidated| < |ALL_ANSWERS|) {
    ToRemove := Invalidated
  if (|Validated| > 0) {
    ToRemove := ToRemove U (ALL_ANSWERS \ Validated)
  AnswersChoices := AnswersChoices \ ToRemove
}
return |AnswersChoices| <= K
```

We provide a rough explanation of what goes on in this pre-decision process. We explore all the ranked relevant passages as long as we have not eliminated enough answers. When faced with a new passage, we PAS-align each answer choice with the passage (section 2.3). The ranked alignments go through validation and invalidation rules and the corresponding answer is invalidated if a PAS alignment is found invalid.

Final decision If we go through all the passages without invalidating enough answers, we choose not to answer the question. If we have K or less answers remaining, we pick the one with the strongest PAS alignment score found in the passages.

The rules We manually built 2 validation rules and 3 invalidation rules. They operate on PAS alignments.
In the current iteration of our system, the validation rules must be fired simultaneously to validate an answer choice, whereas only one invalidation rule fired is enough to invalidate an answer choice. We consider that it is indeed usually

much more difficult, even for a human, to invalidate a wrong answer than it is to validate a correct one with accuracy.
In the description of the rules, *polarity* means the predicate truth value of the common word found in the semantic relatedness score. Compatible polarities are P with P, N with N, and U with P,N,U.
Strong alignment means that the word-to-word alignment score is above a threshold, manually set in this system. The validation rules are as follows:

- **Rule 1**: Subject and Predicate are strongly aligned in both PAS, and all polarities are compatible.
- **Rule 2**: Predicate and one Argument are strongly aligned in both PAS, and all polarities are compatible.

The invalidation rules are as follows:

- **Rule 1**: One polarity mismatch is found in a strong alignment.
- **Rule 2**: Predicates are strongly aligned, but their Subjects are not aligned at all.
- **Rule 3**: The alignment is located at least 2 sentences before the best (question, passage) alignment. We noticed on the trial corpus that the correct answer was usually found after the mention of the question in the document.

3 Data and results

3.1 CLEF 2014 QA Track: Entrance Exams data and evaluation

Our data consist of the trial and test sets at CLEF 2014 Question Answering Track, Task 3: Entrance Exams. Both trial and test sets feature the same format: a series of 12 texts, and for each of them, 5 multiple-choice questions to answer: 60 questions in total. There are 4 answer choices possible for each of the questions. This corpus has been extracted from the Tokyo University Entrance Exam in English as a foreign language.
Systems are evaluated according to their c@1, defined in equation 1.

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

with n the total number of questions, n_R the number of correctly answered questions, n_U the number of unanswered questions.

3.2 Results

In this section, we report the results of our best run on both trial and test question sets. This system uses $K = 2$ as the maximum number of answer choices allowed to make the decision. Passages are 5 sentences long and the algorithm described in section 2.4 uses a maximum of 10 passages before reaching non-decision. We tried other values of those parameters, but the corresponding runs

were performing worse on both trial and data sets, so we do not feel the need to further expose results about them. We focus instead on the differences on trial and test results for the described run. Both datasets contain 60 questions. Table 1 reports global results of our system on both sets of questions. As we can see, performance is quite satisfactory on the trial dataset, but really poor –at the level of the random baseline– on the test dataset.

Table 1. Results on trial and test

	Trial	Test
Questions answered	42	36
Errors	22	25
Accuracy	0.48	0.31
c@1	0.43	0.25

We also analyzed in a fine-grained way the accuracy and efficiency of our validation and invalidation rules. Table 2 describes the error rate on questions where invalidation rules ended up eliminating a correct answer choice. In each cell, we find the number of questions where the particular rule eliminated a correct answer, and we also find the number of questions it fired on –where it eliminated at least one answer choice. As we can see, all rules misfire more on the test dataset, which seems to correlate well with the overall poor test performance. Validation rules did not fire as often as the invalidation rules (4 times for trial, and 5 times for test).

Table 2. Invalidation error on trial and test

	Trial	Test
Rule 1	2 / 21	4 / 13
Rule 2	6 / 25	8 / 22
Rule 3	8 / 26	10 / 28

It is possible to frame our rule system as a kind of information retrieval system and evaluate it in term of validation precision and recall, and invalidation precision and recall. For validation, relevant items are the correct answers, and for invalidation, relevant items are the incorrect answer choices. We compare our results in both trial and test datasets with a random baseline, which basically has a 50% chance of validating and invalidating each answer choice and stops under the same conditions as our system (when K or less answer choices remain). Results in term of precision and recall are shown in table 3. As we can see, surprisingly, validation/invalidation does not seem to perform significantly differently from random guesses on test data, while it is not the case at all on trial data.

Table 3. Precision/recall of validation/invalidation

System	Random	Rules (Trial)	Rules (Test)
Validation precision	0.24	**0.36**	0.25
Validation recall	0.40	**0.57**	0.42
Invalidation precision	0.75	**0.83**	0.74
Invalidation recall	0.60	**0.66**	0.56

We finally report the performance of our passage retrieval system compared to several baselines. Prior to this participation, we had annotated the correct passages on 45 of the 60 questions of the trial dataset, this allows us to compute the MRR (Mean Reciprocal Rank) of our passage retrieval method. Results are shown in table 4. Baseline 1 is a simple word-overlap-based measure. Baseline 2 is Baseline 2 with TF-IDF weighting and WordNet keyword expansion. We then ran our system with and without taking into account the order preserving property of the reading tests, mentioned at the end of section 2.2. As we can see, our passage retrieval method vastly outperforms typical baselines, and the assumption about order of questions being preserved in the order of passages seems to hold and help quite a bit, at least on trial questions.

Table 4. Passage retrieval performance

System	MRR
Baseline 1	0.36
Baseline 2	0.45
System without order	0.49
System with order	0.58

4 Conclusion

Our system has been developed with the invalidation of wrong answer candidates in mind, specifically to answer multiple-choice questions. In the CLEF 2014 evaluation campaign, Question Answering track, the submitted run performed at the level of the random baseline in the Entrance exams task and thus did not reproduce satisfactory trial performance.
In further works, we plan to study the reasons behind the differences in performance of our system on trial and test results and to better integrate predicate-arguments structure selection with passage retrieval, the module of our system which seemed to perform well. Other features considered include a character name resolver addition to coreference resolution, and analyzing discourse relations.

References

1. Banerjee, S., Bhaskar, P., Pakray, P., Bandyopadhyay, S., Gelbukh, A.: Multiple choice question (mcq) answering system for entrance examination. CLEF 2013 Working Notes (2013)
2. Gleize, M., Grau, B.: A hierarchical taxonomy for classifying hardness of inference tasks. In: Chair), N.C.C., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., Piperidis, S. (eds.) Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). European Language Resources Association (ELRA), Reykjavik, Iceland (may 2014)
3. Gleize, M., Grau, B., Ligozat, A.L., Pho, V.M., Illouz, G., Giannetti, F., Lahondes, L.: Selecting answers with structured lexical expansion and discourse relations
4. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1. pp. 423–430. Association for Computational Linguistics (2003)
5. Lotan, A., Stern, A., Dagan, I.: Truthteller: Annotating predicate truth. In: Proceedings of NAACL-HLT. pp. 752–757 (2013)
6. Peñas, A., Miyao, Y., Hovy, E., Forner, P., Kando, N.: Overview of qa4mre 2013 entrance exams task. Working Notes, CLEF (2013)
7. Recasens, M., de Marneffe, M.C., Potts, C.: The life and death of discourse entities: Identifying singleton mentions. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 627–633 (2013)
8. Sutcliffe, R., Peñas, A., Hovy, E., Forner, P., Rodrigo, Á., Forascu, C., Benajiba, Y., Osenova, P.: Overview of qa4mre main task at clef 2013
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. pp. 173–180. Association for Computational Linguistics (2003)
10. Wikimedia Foundation: The simple english wiktionary, `http://http://simple.wiktionary.org/`