Milestone 3: Working Notes

Team 9: Napat Luevisadpaibul, Dishan Gupta, Callie Vaughn, Wenyi Wang

1. Introduction

Given a question, selecting the correct answer from a set of potential choices is a difficult task for an automatic system. In this project we refined a system for selecting answers to questions based on the information provided in documents. Although such a system is a long way from matching the expertise of a human, we were able to use some natural language processing and statistical techniques to achieve an accuracy that was significantly higher than chance.

2. The Data Set

The data set that we used for this task was the QA4MRE Alzheimer's data set from 2012. The data consist of a set of documents with associated multiple-choice questions. Each question has five potential answers. In our development set each document had 10 associated questions, and in our blind test set each document has either 10 or 15 questions associated with it. The domain of all of the documents is medical in nature and deals with the disease Alzheimer's in some form or another.

3. Our Approach

We began this project with the initial baseline system as given to us by the TAs for this course. To improve the system, we made the following modifications: implementing cosine similarity between the questions and answers as part of the answers' scores; implementing both PMI scoring and an alternative similarity scorer as part of the answers' scores; discarding obviously incorrect answers using rules; changing the criteria for defining noun phrases; eliminating the possibility of a "null" choice as our system's answer; and several other techniques.

3.1 Pointwise Mutual Information:

The original system used a default similarity metric. We substituted the provided pointwise mutual information (or PMI) scorer for this default metric and saw a substantial increase in our accuracy on the development set.

3.2 Alternative similarity scorer:

In addition to the PMI scores and the default similarity scores in the given code, we want to implement extra methods, since the two existing similarity scores may not perform well on some documents and we want to find a more accurate and appropriate way to calculate the similarity score. Inspired by the previous homework, we come up with this idea, including implementing cosine similarity, Dice coefficient and Jaccard coefficient. Cosine similarity is a well-known measure in information science. We combine it with Dice coefficient and Jaccard coefficient and we call them alternative similarity scorer. Later on, we also calculated the cosine similarity between question and its answers.

3.3 Adjusting weight to combine scores:

Basically, we have three scores, namely the original similarity score, the alternative similarity score and PMI score. Our goal is to combine the three of them in a reasonable way. For the first two scores, they are already modified and normalized to the range (0,1). Meanwhile, the PMI score in the given code was a sum of the PMI scores of all tokens in sentences and its range is neither (0,1) nor (-1,0). In order to better combine the three of them, we want to normalize the PMI score.

After making this change, the performance of using merely PMI score stays the same (better than the other two scores combined). However, when using the sum of the three scores, the performance drops 2.5% (1 out of 40). This is reasonable since the PMI score is better and the new combination reduces the weight of PMI score. Finally, we may give up using the other two similarity scores and may use only PMI score.

3.4 Answer Discarding:

We used a rule-based method (in the annotator AnswerDiscardAnnotator) to discard answers that clearly did not fit with the questions. There were several types of these:

- "How many…" questions:

  If a question begins with "How many", its answer must be numeric. This means that the correct answer is a phrase made up only of integers and number words such as "million", "thousand", or "hundred". All answers that do not fit this pattern were discarded.
- "What are…" questions:

  Any question that begins with "What are" must have an answer that is plural. That is, the answer should either be a plural noun (i.e. end with "-s") or it should contain the word "and".
- "What [pluralNP]…" questions:

  Any question that begins with "What" followed by a plural noun phrase must also have a plural answer. As described above, this means that all answers that don't end with "-s" or contain the word "and" should be discarded.
- "Which…" questions:

  As we read through our output, we found that many questions start with "which X", where X is a category, and that some of the choices may not even belong to the category. In light of this observation, we want to discard some invalid choices based on the category, using some background information.

  The first and a fundamental step of this modification is to find appropriate knowledge base, in which we have the correct choices and have no incorrect choices. We started with using Wikipedia. We found the pages of "amino acid", "histone deacetylase inhibitor", "enzymes", "human hormone" and "peptide" on Wikipedia. With these knowledge bases, we got a relatively 20% increase in performance when using "simpleQuestionRunCPE" (from 15 to 18 correct answers). Later on, we also included the knowledge base for "protein". The protein is a more complicated concept with many sub-categories than other. Moreover, there is no naming rule for protein, so we need to find a comprehensive list of proteins.

3.5 Noun Phrases:

In the baseline system, noun phrases are defined as any contiguous string of nouns, adjectives, and cardinal numbers. By removing adjectives from this definition so that only nouns and cardinal numbers are considered noun phrases, we improved our accuracy on the development set.

3.6 Null Choice Elimination:

During error analysis, we realized that many of the questions that we were getting wrong were questions that we were not answering at all. Our highest answer choice in these cases was null, and this was true even for some questions for which we had discarded all but one answer (the correct answer). Therefore, we modified the code to initialize every candidate answer's score to a small number so at least we would choose one of the answer choice even if all answers got a score of zero from their candidate sentences. This eliminates "null" selections and gives the system at least a 20% chance (up to 100% chance, depending on how many obviously incorrect answers were discarded) of guessing the correct answer.

3.7 None of the Above:

In the final blind test set, there are many questions with one "none of the above" candidate answer. Since the system didn't have any concrete way of dealing with such a possibility before, and since the tokens in the phrase "none of the above" are not more likely than incorrect answer tokens to appear in the document, we decided to select "none of the above" if it exists as a potential answer choice and the other answer choices score below a certain confidence threshold.

3.8 Term Frequency:

We noticed that a number of the incorrect answers that our system was choosing were mentioned far less frequently in the document than the correct answer. Therefore, we added the term frequency of the answer candidate in the document to that answer candidate's score. In the cases where the answer candidate did not appear in the document, this had no effect on the score (as was the case for answers that were long phrases rather than individual words). In all other cases, however, this proved to be a useful technique in creating a part of the answer choice score.

3.9 Ignore invalid answer removal if we remove all answer:

If by any chance we remove all the answers, that means we have no chance to select the correct answer. So we should not use invalid answer removal logic in that question. Therefore, if all answers have been discarded for a given question, all of those discards are ignored, and the answer is selected from the original five answer choices.

3.10 Add score to answers which have PMI or similarity score match with question:

We noticed that in the development set some questions begin with "Which CLU" and the correct answer is CLU1. So if we augment the scores of answers that have high PMI with their question we can answer this type of question correctly.

4. Results

The initial system as provided by the TAs got the correct answer 7 times out of 40 (17.5%) on the development set, giving a c at 1 score of 0.1925. With our changes as described above in section 3, our system improved on the original data to answering 12/40 (30%) of the questions correctly, for a c at 1 score of 0.3.

5. Error Analysis

Many of the questions for which we still select the incorrect answer require expert knowledge that is deeper than syntactic or even semantic category. For example, there are several questions that require selecting the correct DNA sequence from the list of answers. Since there is no comprehensive list of DNA sequences available online, we cannot even definitively remove non-DNA sequences from the list. An additional problem we noticed is that one question asks about the incidence of Alzheimer's in the future. The answer is included in the document as a percentage of the world's population, but the answer choices given are only in raw numbers.

6. Conclusion

Question answering is a difficult problem requiring extensive background knowledge even when searching a document for answers to multiple choice questions. There are certain linguistic and statistical tricks that can help, though, as our increase in accuracy shows.

7. Work Distribution

We all contributed to writing, testing, and running the code. Individually, our contributions were as follows:

Callie Vaughn:

- AnswerDiscardAnnotator (for the first three of the four types of answers)
- Term frequency module
- Working notes (wrote introduction, error analysis and conclusion, edited the final version)
- Parameter selection for our final 10 output versions

Napat Luevisadpaibul:

- Null Choice Elimination
- None of the Above handling
- Ignore invalid answer removal
- Add score to answer which have PMI or similarity score match with question
- Integrate OutputRunCasConsumer to pipeline
- Change criteria to seleck k candidate sentence (but we don't include this in final system)
- Write program to extract protein list from protein database
- Annotate discard answer choice in phase1
- Working notes (wrote some of the approaches)

Wenyi Wang:

- Removed some minor bugs in the original code
- Added normalization to original similarity scorer
- Implemented and integrated the alternative similarity scorer (including cosine similarity, Dice coefficient and Jaccard coefficient)
- Adjusted weights of different scores
- Found and organized knowledge base for biology terminologies (protein, enzyme, etc.)
- AnswerDiscardAnnotator for the "which" type questions
- Ran experiments and generated output files for submission
- Maintained wiki pages (meeting notes, individual contribution, etc)

Dishan Gupta:

- Noun Phrases criteria (NN, JJ, CD etc.)
- PMI Scoring with respect to background corpus
- Parameter Tuning (Minimum score threshold, Term frequency weight, Noun phrase criteria etc.)
- Tried synonym generation but couldn’t train on Alzheimer’s corpus with the timeline.