CS 4740 – NLP

Team Members: Michas Szacillo (mas744), Alexandra Ward (amw349), Brandon Sweezer(bas329), and Abiy Tibebu (agt33)

Project 3 Proposal

Baseline System:

For the baseline, we decided to use a simple Random Guess model. To do this, instead of returning a random word from the given paragraph, we decided to use IOB tags to make the selection process easier, and to return an entire IOB phrase instead of a single word. We used Stanford’s NER model to generate the IOB tags, this

Our decision to use random guess is mostly so that we have an extremely simple model to start from, but to also start gaining intuition on the QA problem itself. However, we did go a step further than returning just a random word or phrase, since we are classifying the paragraph contents by IOB tag. The random guess will have relatively decent odds of guessing the correct answer, as there are only a handful of noun phrases per sentence, and a handful of sentences per paragraph.

Baseline Results:

|  |  |
| --- | --- |
| Exact Match | F1-Score |
|  |  |

Proposal:

To begin, we plan on trying some more simple models such as the Window Sliding model used in the critique article. We also plan on playing around with tagging systems to make picking out the answers in the paragraph easier (for example POS tags, or IOG tags). As we continue to improve our model, we will look into possible linguistic NP filters as well, since we know that most of the answers to the questions are phrases and not just words.

Similarly, we plan on classifying question types and semantic class types so that we can narrow down the possible answers even further to a corresponding return type (e.g. Person, Place, etc). We would do this by searching questions for identifying words such as “who” and “where” which would indicate the question is looking for a person or a place, respectively.

This answer classification could be improved from hard-coded rules by using a machine learning approach. We could train a neural network to identify the most likely NER tag which a question is looking for using a training corpus found online, or worst case, hand generated by us. Using this trained neural network, we could give it our questions and then (hopefully) get the correct NER tag to look for in the sentence. Done correctly, this would vastly improve the answer parsing process, for we would have an accurate and reliable way to determine what sort of thing we are looking for in the paragraph. Once we know this, it should be simple to identify which of the possible answers in the paragraph is the right one using the following method.

After we have found an answer, we plan to evaluate the validity of that answer using a method we will call Statement Rearrangement Perplexity (SRP). SRP will rearrange the phrasing of the question and the previously identified answer possibilities so that they form a statement. Then it will evaluate the perplexity of each of these three sentences, using N-gram (we will not use anything higher than trigrams as the answers will not be very long) models generated on the paragraph and question. The least perplex answer will be selected as our final answer. We believe this will improve our accuracy, for the correct answer will have extremely low perplexity, as the rearranged statement will have almost identical phrasing to that of the answer in the paragraph.

Group Contribution:

Brandon: Wrote writeup, came up with some ideas for part 2, setup git repo and slack channel.

Alexandra:

Michas:

Abiy: