Project 3 - Report

Team

Akash Peri (ap659)
Ishaan Jain (irj4)
Vinayaka Suryanarayana Bommathanahalli (vb247)

Format Correctness

A sample from the dev output json:
{"57296eb01d04691400779436": "such loss is also rarely observed in other plastids and prokaryotes", "572736bff1498d1400e8f4b8": "Genghis Khan ", "572735a15951b619008f86c2": "Demand ", ...

Preprocessing on the data

In all cases, we stripped out stop words, and made all words lower case. This was to focus on the content words. Also, to improve performance, we created a Pickle file containing word vectors representing questions and a dictionary mapping question intents to categorical features.

The words vectors were of length 301 - 300 dimensions and 1 label

The intent dictionary was of the format *'Intent' : feature* to facilitate lookup and substitution.

We also had a Json file that contained some important features generated by our system for a question. This was also done so that we did not have to recalculate these multiple times. The json file was of the format:

```
{
    question_id : (
                    “CANDIDATE SENTENCE with maximum cosine similarity”,
                    “INTENT”,
                    “QUESTION”
                    )
}
```

This brought the execution time down from half an hour per dataset to 5-6 minutes per dataset. This let us experiment with more tweaks to our code.

Project 3 - Report

Description of baselines implemented, and enhancement made for this project

Our goal was to build a baseline for answering questions. We started by running evaluate.py to see how the evaluation was done, then looked at the data, to figure out what it looked like. Since it resembled the data used in SQuAD, we decided to adopt the sliding window mechanism used as a baseline in that paper. The intuition behind the sliding window is that the answer is guaranteed to be found in our context, and we can attempt a best guess at a possible answer to the question without involving the semantics of the words themselves yet. Once we filter out stopwords, we are then left with content words that we can rank by proximity to content words from the question.

To elaborate, each essay was split up into multiple paragraphs or contexts. Each of these contexts had multiple questions to be answered for it. We decided to build a sliding window of 10 for each of the contexts. In our baseline, we computed a naive similarity measure with a question by counting the number of shared words between the two strings. We decided to swap that for a cosine similarity measure, taken from the SPaCY library. Spacy also takes into account synonyms, which is convenient for us. We then consider windows that have a high similarity as ones most likely to contain the answer.

To find the best window from this subset, we use the last word from the question as an anchor. This is because generally in the context, that last word is followed by the answers. We then find the occurrence of the last word in the candidate windows. The assumption is that the later in the window the word appears, the fewer candidate answers words we can glean from the window. Therefore, we add the fraction of number of words after the anchor to total words in the window to the similarity count we've found. Hopefully, this allows us to break the tie. If not, we choose a window randomly.

At this point, we now have a successful window in which our answer can lie. After removing any other stop words, we then a) either submit the entire window, or b)randomly sample a word from it and return it as the answer. We postulated that approach a) would perform better on the F-1 score, since we have a larger span which could contain the answer at the cost of the EM score. As for approach b), the EM would be higher at the cost of F-1 because of the inherent randomness of picking a candidate. Our theory was proven correct through running our baseline on the testing set.

Based on the feedback received from the proposal, we decided to implement all of the following:

1. implementing the "smarter approach" for question identification.

2. finding a non-trivial way to compute the "similarity score" for each chunk and the question.

3. using more features for answer extraction

Project 3 - Report

1. We used two different classification models on a particular question - intent dataset (from Glove), details below and used it to classify the intent of the questions from SQuAD on-the-fly. If we couldn't find an appropriate intent, we then fell back to the naive, rule-based approach. If this failed as well, we then used the last-resort technique of looking up all matching POS tags for intent groups. Eg, if the question is '*HUM*' for human, and we do not find it, we then scan for all NNP POS tags in the candidate sentence, or in the context instead.
2. For similarity, we use the window based-approach with cosine similarity between the window and question (without stop words). This similarity metric is a better indicator than our baseline system - albeit with a few issues of its own (discussed below)
3. Apart from NER and POS tags, we tried to implement two things: span identification for multi-word answers and incorporating dependency graphs from spaCy into our feature vectors. However, due to abstractions in the paper, we did not fully understand how to integrate these features into our systems.

Question Identification:

The heart of our Question Answering system revolved around question identification. We decided that by first determining what the question was asking, we could then filter the context to only look at applicable terms, via rule based classification. To identify the question, we first focused on the 'w' words. If we saw a who, what where, when or why, we could immediately narrow down what kind of answer we were expecting. The mapping is summarized below. If we saw "when" for instance, then we would respond with a "date" tagged entity from the answer.

When:	Date, Time
Who:	Person
How:	Many: Person
	Much: money, quantity, percent
Which:	person, quantity, percent
Where:	gpe, loc, facility
Whose:	person, org, facility, gpe, loc
percent:	percent

The next problem we faced was how to identify key terms in the question and context. For the question, we did a simple word match to see if the keywords highlighted above were present in the question. For the context, we used libraries from spaCy, an NLP library. First, we used spaCy to break our context into sentences. On these sentences, we were originally running the naive similarity mechanism we'd developed in our baseline. It barely worked, so we'd tried switching to cosine similarity, with slightly improved performance. Finally, we decided to utilize spaCy's similarity measure. For the most common words, spaCy maintains a word vector representation that it can do a direct comparison on. For less common words, it uses the document itself, with a four word window on either side as context. We figured this would be a

more effective similarity measure to employ. With this similarity measure, we were able to rank the sentences to find the sentence closest to the question. This sentence likely contains the response to the question.

Here, we come to the second problem. Our representation above relies on us being able to assign named entities to spans in a sentence. That is, we need to be able to detect that 'New York' is a location, similar to our NER project. In this case, we leveraged spaCy's built in NER system. We did actually try to use our own, but found that it's performance was horrendous, and that we were waiting too long for it to complete for it to be worth the hassle. With this, we can access tags such as B-PER that we can use to complete answering our posed question. At the end of this stage, we were able to make predictions.

We weren't happy with this performance however. When analyzing the predictions, we noticed that our rules did not adequately enumerate the possibilities well. Rather than building more elaborate rules, we decided to implement a classification model to capture the intent of a question. We found a dataset online (Experimental Data for Question Classification http://cogcomp.org/Data/QA/QC/) that had labeled questions with their corresponding intents. We used this dataset to train our classification models. For the features, we used word2vec to generate 300 dimensional vectors for each word in the question. We then summed up those vectors together and normalized by the length of each question to build a vector representation for every question.

After training our classification model (Logistic Regression and kernelized SVM), we then used it to identify the most likely question intent. We kept using the spaCy similarity measure to identify the most likely candidate sentence for the answer. We also used spaCy to assign NER and POS tagging for spans within the sentences. We then retrieved the first matching NE within the candidate sentence, and returned that as our proposed answer. There were challenges with this as well. Often, the most likely question intent had no matching NE tags within the candidate sentence. In this scenario, we then fell back to POS tags, and returned the first proper noun we encountered. Failing that, we would then return the entire sentence.

Example Dry-Run

Below, we go over an example from the data set, and how our various models classify it.

Considering our intent classification model based system:

Example of Correctness:

Question: "What was the name of the free music promotion on Kanye's website in 2010?"
Intent Classification: Entity
Max similarity sentence: GOOD Fridays through his website, offering a free download of previously unreleased songs each Friday, a portion of which were included on the album.

Project 3 - Report

Answer picked by system: GOOD Fridays
Correct Answer: GOOD Fridays

The SVM correctly classifies the question to expect an entity based answer (possibly due the What at the beginning and the word - name in the sentence). The system then identifies the above sentence as the one with maximum cosine similarity after removing stop words. It then picks the closest entity-based NER from the sentence (like PERSON, LOCATION etc.) and pick GOOD Fridays which was tagged as a PRODUCT.

Example of Incorrectness
Question: What place on the Billboard chart did Kanye's album debut at?
Intent Classification: Entity (Actual Intent: Number)
Max similarity sentence: The College Dropout received near-universal critical acclaim from contemporary music critics, was voted the top album of the year by two major music publications, and has consistently been ranked among the great hip-hop works and debut albums by artists.
Answer picked by system: College Dropout
Correct Answer: 2

We can immediately see what went wrong in this case. The SVM expected an ENTITY (based on the logic in the previous example), and the maximum similarity sentence picked was also incorrect. Hence, it picks an highly incorrect answer after falling back to NER tags - since the system is sure it requires an entity.

Considering our intent classification model based system:

Example of Correctness:
Question: Who broke the world record for simultaneous fireworks?
Intent Classification: PERSON
Max similarity sentence: Since 1992 the Music of the Night has been performed in the Royal Citadel by the 29 Commando Regiment and local performers to raise money for local and military charities.
Answer picked by system: Commando Regiment
Correct Answer: Commando Regiment

The Major difference here is that we used hand-written rules to identify the intent of the question. Since the question started with Who, we can say with certainty that it requires a PERSON answer.

Example of Incorrectness:
Question: 'What festival was included in the scenes shot in Mexico?'
Intent Classification: 'PRODUCT'

Project 3 - Report

Max similarity sentence: 'With filming completed in Rome, production moved to Mexico City in late March to shoot the film\\'s opening sequence, with scenes to include the Day of the Dead festival filmed in and around the Zócalo and the Centro Histórico district.'
Answer picked by system: 'filming Rome production Mexico City March film opening sequence scenes Day festival Zócalo Centro Histórico district'
Correct Answer: Day of the Dead

Since we wanted a What - based answer, we specialized to a PRODUCT, however, while it is correct that we are looking for an Entity, the specialization of PRODUCT is incorrect - resulting in the fallback model activating.

Why is our model successful?

Questions are generally fairly predictable to answer. If the question starts with "When", then the answer is most likely to be a date. If it starts with "How much", and has words like "cost" or "spend" or "money", then the answer is likely numerical and monetary. A few such rules are sufficient in capturing the some of questions.

However, we found that many of the questions were HOW, WHY and WHAT based. These need a description in the answer with a sound reasoning. We made our best attempt to find the answer in such a case and in the worst case return the whole candidate sentence with the context word removed.

Indeed, we observed that the rule based QA system had a decent performance. We decided to switch to logistic regression to better handle the edge cases, for which we would have had to design more and more narrow rules. This is an effective model because the words in the question are generally sufficient to classify the intent, and so, we can use the words of the question as features.

Project 3 - Report

Results:

Here are our final classification results:

	Dev		Test	
	Exact Match	F1 Score	Exact Match	F1 Score
Sliding window, entire span *(Baseline)*	0.99	18.5	0.987	18.53
Sliding window, random word *(Baseline)*	5.01	11.81	5.08	11.81
Intent-based rule classification	8.55	17.81	6.907	16.783
Best model combining all the above models in a backoff technique	8.533	21.752	9.133	20.14

Analysis:

We can see that our intent based classification was better when it comes to exact matches than the other two baselines. However it wasn't much better than the sliding window. This went contrary to our expectations, which were that the intent based classification mechanism should be much better at classifying NE based questions. The F1 score also rose a similar unimpressive amount. Comparing to the original SQuAD paper, it maps to the sliding window baseline with distance extension. We suspect it's because we don't do a great job classifying intents, as seen in the examples

Observations made:

The performance on Named Entities is on par with the paper. This makes sense, as we focused the most on this, and our model was trained with Named Entities. The performance for non-Named Entities is also terrible, as expected. It's more challenging to train our model adequately for non-Named Entities, and harder to take them from the sentences.

Potential Issues:

The way we construct our logistic regression features does not consider word order, just the words themselves. Thus, it would treat the following two questions the same.

Why did the chicken cross the road?

Project 3 - Report

Why did the road cross the chicken?

Is it how you can sing? (referencing an object that allows the user to sing)
How is it you can sing? (expressing disbelief the subject can sing)

The example given is contrived, but there could conceivably be a similar question pair where the intent changes given a different ordering. A potential solution for this could be to employ something similar to how objects are hashed in Java. We can multiply each word vector by a prime number to ensure the order matters as well.

The next potential issue is that when we choose the NE to return from the sentence, we return the first one that matches the intent. We didn't know of any better measure than this, and found that returning a random matching entity resulted in a worse performance. This could be an area of improvement.

Work Split:

Akash: preprocessing, baseline system and experimenting with spacy
Ishaan: classification model for question intent classification, generating data files, attempted Match LSTM
Vinayaka: Intent based answer finding, NER based Answer finding, POS Based answer finding, Attempt to work with Stanford NLP APIs