Reading Comprehension on Short Passages for Question Answering

JAKE PITKIN
CS 6390 - Information Extraction
April 25, 2017

Task Description

The task I worked on was trying to understand short passages and use this understanding to answer questions. This is a micro-reading task as the answer to each question is a segment of text from a short article. This requires trying to get a deep understanding about the semantics of a short article and a question. Once trained, my system takes as input a short article and a collection of questions about the article. For each question, it outputs its best guess at the answer to the question.

External Resources

I leveraged some external NLP and machine learning resources to design my system.

SQuAD (*https://rajpurkar.github.io/SQuAD-explorer/*) - Stanford Question Answering Dataset (SQuAD) is a large collection of articles (500+) with accompanying question-answer pairs (100,000+). The dataset was created from Wikipedia articles by crowdworkers. Additionally I used the provided evaluation script to produce some of my evaluation metrics.

NLTK (*http://www.nltk.org/*) - I used the NLTK library for common NLP tasks. These tasks included tokenizing words, sentence splitting, and creating syntactic parse trees of sentences.

Scikit-learn (*http://scikit-learn.org/stable/*) - Scikit-learn is the machine learning library I chose to use for my logistic regression model. I additionally

used the library to generate tf-idf scores for words and to perform cross-validation on the logistic regression model.

Technical Design

The provided dataset came with a wealth of labeled examples (100,000+) so I decided to train a supervised machine learning classifier. Features were created for positive and negative examples and a logistic regression classifier was trained. Come prediction time, a collection of candidate answers were passed through the classifier to produce a confidence score for each candidate (YES this is a correct answer or NO this is not a correct answer) for a given question. The candidate with the highest confidence score was predicted to be the correct answer.

Populating a Candidate Pool - The answer to each posed question is a relatively short span of text from the corresponding article. As such there exists $O(L^2)$ (where L is the number of words in the article) possible answers to a question, this list needs to be culled. To cull this list down to a workable size I used a heuristic: the answer is a noun phrase. I generated syntactic parse trees for each sentence in the article and added each found noun phrase to a candidate pool. It's extremely unlikely that the answer to a factoid question would be a pronoun so these were removed and articles were stripped from remaining candidates.

This candidate pool was generated at two different times in my system. The dataset provided by SQuAD only consisted of positive or correct answers to a given question. Candidate answers from the pool that had no word overlap with the ground truth answers were used as negative or incorrect answers. Additionally this candidate pool is generated at prediction time as a list of possible answers to a question.

Extracting Features - Given a collection of candidate answers and the provided ground truth answers, features for examples were extracted. Each training example is based on a candidate answer or ground truth answer, a question, and the context of the answer. The following features were generated for each example.

Description	Extraction Process
Sum of the TF-IDF of words occurring in both the question and the sentence containing the candidate answer.	Scikit-learn used to generate TF-IDF scores for all words in the training corpora.
Sum of the TF-IDF of the words occurring in both the candidate answer and the question.	Same as above.
Sum of the TF-IDF of the words surrounding the candidate answer also in the question (separate feature for left and right).	Same as above.
Binary feature indicating the presence of a w-word in the question (who, what, where, when, which, or none.)	Parsing of the question.
Lexical overlap of sentence containing the candidate and the question.	Tokenized both using NLTK and took the set difference.
The length of the sentence containing the candidate answer minus the candidate.	Simple parsing I wrote.
Candidate contains a capital letter.	Same as above.
First letter of all the words in the candidate are capitalized.	Same as above.
Candidate contains a digit.	Same as above.
Head noun of the candidate.	Same as above.
Word preceding the candidate.	Same as above.
Word succeeding the candidate.	Same as above.

Table 1: Extracted features for each training and test example.

Training a Logistic Regression Model - Used scikit-learn to train a logistic regression classifier with an equal number of positive and negative training examples. Due to the large size of the SQuAD dataset, I only used about 20% of the training data to keep training time reasonable. Using another partition of the dataset, scikit-learn was used to perform cross-validation to optimize the hyper-parameters of the model.

Making Predictions - For each article, a candidate pool is created using the same technique used during training. Each candidate answer is passed

through the trained logistic regression model to obtain a confidence score. The candidate with the highest confidence score is predicted as the answer to the given question.

Evaluation - To evaluate my system, I made predictions on about 20% of the development set. The creators of the SQuAD dataset also provide an evaluation script. The script reports an *exact match* (exact match with one of the ground truth answers) and a *macro-averaged F1 score*. The F1 score is the average bag of tokens overlap between the prediction and the ground truth answer.

Evaluation Results

The SQuAD dataset has an accompanying paper (Rajpurkar et. al, 2016) with some provided baselines. Two I found interesting was the performance of taking a random guess and the performance an average human achieves on answering these questions. Additionally I report the performance of my initial system, my final system, and the performance of the state-of-the-art.

Approach	Dataset	Exact Match	F1 Score
Random Guess	Dev	1.1%	4.1%
My Basic System	Dev	5.8%	10.7%
My Final System	**Dev**	**14.57%**	**22.12%**
State-of-the-Art	Test	76.92%	84%
Human Baseline	Dev	80.3%	90.5%

Table 2: Performance of various systems and baselines.

Successes and Regrets

Successes - The thing I was most proud of with this project was my performance gain over my basic system. I was able to triple the exact match performance and double the F1 score. This was attributed to a more complex approach of using a machine learning classifier to reason about the context in which answers appear. The strongest feature was using TF-IDF scores, which greatly helped my system identify words that were important to an article and likely to be an answer to a factoid question. This was my first time building feature vectors from scratch for an NLP task and using scikit-learn for machine learning. I learned a lot there that I can carry over to future projects.

Regrets and Next Time - A large bottleneck my system had was finding candidate answers. The heuristic of only considering noun phrases generated by an syntactic parse tree only found the correct answer about 55% of the time. Next time I would use a stack of multiple algorithms and heuristics for finding candidates. Using a NER classifier I imagine would be useful for finding candidates as many answers to factoid questions are named entities. Additionally, having a coreference layer would allow me to couple pronouns with a corresponding proper noun and would give more contextual information to work with.

The questions posed were mostly short factoid questions. It would be quite plausible to categorize questions based on what the question is looking for (a time, a place, a person, etc.) and then train a separate classifier for each question "type". I think using a separate model for each question type would give a performance gain.

Finally, I think I could of taken an approach that is better suited for a micro-reading tasking. I read some related work on reading comprehension for biological processes (Berant et al., 2014). The work aims to model the relationships between entities and events in a short biological process to answer multiple-choice questions. The authors achieved good results and in hindsight aiming for a "deeper" understanding of the articles would of been advantageous as the articles are short in length and the answer usually only appears once.

Appendix

Below is an example of an article, a question, and what my system extracts. There is a training example created for most candidates in the candidate pool and each ground truth answer. I'll show the features created for the predicted candidate (in this case a mostly correct prediction but not an exact match).

Sample Output

Article - On April 20, Kennedy sent a memo to Vice President Lyndon B. Johnson, asking Johnson to look into the status of America's space program, and into programs that could offer NASA the opportunity to catch up. Johnson responded approximately one week later, concluding that "we are neither making maximum effort nor achieving results necessary if this country is to reach a position of leadership." His memo concluded that a manned Moon landing was far enough in the future that it was likely the United States would achieve it first.

Candidate Pool - opportunity, Kennedy, Johnson, programs, NASA, memo, April, Vice President Lyndon B. Johnson, America, space, status, program, position, results, leadership, Johnson, maximum effort, week, this country, landing, States, future, memo, Moon, United.

Question and Ground Truth Answer - Who was Kennedy's vice president? *Lyndon B. Johnson*

Features for Candidate 'Vice President Lyndon B. Johnson'

Feature Description	Feature Value
Sum of the TF-IDF of words occurring in both the question and the sentence containing the candidate answer.	**kennedy, vice, president** 0.344
Sum of the TF-IDF of the words occurring in both the candidate answer and the question.	**vice, president** 0.2179
Sum of the TF-IDF of the words surrounding the candidate answer also in the question (separate feature for left and right).	*left:* **kennedy** 0.1261 *right:* 0
Binary feature indicating the presence of a w-word in the question (who, what, where, when, which, or none.)	who *true* and the rest *false*.
Lexical overlap of sentence containing the candidate and the question.	2
The length of the sentence containing the candidate answer minus the candidate.	31 words
Candidate contains a capital letter.	*true*
First letter of all the words in the candidate are capitalized.	*true*
Candidate contains a digit.	*false*
Head noun of the candidate.	*Johnson*
Word preceding the candidate.	*to*
Word succeeding the candidate.	*asking*

Table 3: Extracted features for 'Vice President Lyndon B. Johnson'.

Prediction - My systems final prediction is *Vice President Lyndon B. Johnson* when the ground truth is *Lyndon B. Johnson*. This is counted as a partial match and improve the F1 score but is not an exact match.

References

Berant, Jonathan, Srikumar, Vivek, Chen, Pei-Chun, Huang, Brad, Manning, Christopher D, Vander Linden, Abby, Harding, Brittany, and Clark, Peter. Modeling biological processes for reading comprehension. *In Proc. EMNLP, 2014.*

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *In EMNLP, 2016*