Machine Comprehension Using Dependency-based Attentive-BiLSTM

Ragheb Al-Ghezi
University of Arizona
Final Project for CSC585
raghebalghezi@email.arizona.edu

Abstract

Machine Comprehension has received great attention from the NLP community not because solving will bring us closer to understand humans better, but it also has significant applications in the field of education. The rest will be built on along the way.

1 Introduction

Machine Comprehension (MC) is the task of automatically answering comprehension questions based on information derived from short texts. While this task is relatively easy task for the humans to do, it is a hard task for the machine because it requires not only linguistic information, but also cognitive, world-knowledge skills as well. MC has been of great interest to NLP community because of its practical applications in education, especially in the field of automated assessment and intelligent tutoring systems. Working on similar tasks will bring us a step closer to understand how human is able to perform such cognitive tasks so easily while machine is not.

For this task, we will be using the second release of Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2018). SQuAD 2.0 is a reading comprehension dataset, consisting of 150K questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable. The difference between SQuAD 2.0 and its previous release is that it has 50K unanswerable questions in addition to paragraph-supported questions. To do well on SQuAD2.0, systems must not only answer questions when possible, but also determine when no answer is supported by the paragraph and abstain from answering.

The strategy will be used to tackle this task is mainly driven by a simple linguistic intuition. Wh-question often requires named entities as answers. For example, a question with *Who* requires a person entity, *Where* requires a location, *What* requires object, idea or action; or whatever name follows it (e.g. what kind), etc. While this is not always the case, this assumption can help reduce choices of candidate answers span. In addition, information from Named Entity Recognizer (NER) can be augmented with dependency parser labels. Thus, for this task, we propose a two-step system. The first step is to train a logistic regression classifier with rich linguistic features to detect the sentence containing best answer. A null-answer category is accounted for in at this step. Features between q and p may include word-overlap, conditional pos-word overlap, average word embedding similarity. Second, within the best candidate sentence, we find the answer span, which is most likely a named entity using named-entity-to-label matching heuristics.

2 Related Work

There has been a large number of studies tackle traditional Machine Comprehension tasks, where answers are extractive and explicitly stated in the passage. The difficulty of this task, however, lies in abstaining from answers when no answer span exist in the passage. Thus, we will focus on models that not only answers the questions, but also the one that predict the probability of questions being answered.

Seo et al (2016) introduced bi-directional attention flow (BiDAF) that uses an RNN to encode contextual information in both question and passage along with an attention mechanism to align parts of question to the sentence containing the answer and vise versa. The model offers context representation at multilevel of granularity: character-level, word-level and contextual embedding. What sets this work from others is that it does not represent the context paragraph into fixed-length vector. Instead, it dynamically computes vector at each time step, combines it with the one from previous layer and allow *flow* through to the next layers. The model outputs confidence scores of start and end index of all potential answers. One problem with this model is it is not designed to handle unanswerable questions. Levy et al (2017) extends the work by assigning a probability to null-answers to account for the questions whose answers do not exist in the corresponding paragraph. It achieves 59.2% EM score and 62.1% F1 score.

Hu et al (2018) propose a read-then-verify system that is able to abstain from answering when a question has no answer given the passage. They introduce two auxiliary losses to help the neural reader network focus on answer extraction and no-answer detection respectively, and then utilize an answer verifier to validate the legitimacy of the predicted answer. One key contribution is answer-verification. The model incorporates multi-layer transformer decoder to recognize the textual entailment that support the answer in and passage. This model achieves an ExactMatch EM score of 71.6% and 74.23% F1 on SQuAD 2.0.

Wang et al (2018) proposes a new hierarchical attention network that mimics the human process of answering reading comprehension tests. It gradually focuses attention on part of the passage containing the answer to the question. The modal comprises of three layers. a) **encoder layer**: builds representations to both the question and the passage using a concatenation of word-embedding representation (GloVe)@ and a pre-trained neural language modal ELMo @ b) **Attention layer**: its function is to capture the relationship between the question and the passage at multiple levels using self-attention mechanisms. c) **Matching layer**: given refined representation for both question and passage, a bi-linear matching layer detects the best answer span for the question. This method achieves the state-of-the-art results as of September 2018. Their single model achieves 79.2% EM and 86.6% F1 score, while their ensemble model achieves 82.4% EM and 88.6% F1 Score.

While these models achieve astonishing results, they are needlessly complex in terms of architecture design and implementation, and very resource-intensive. This complexity results due the lack of linguistic assumptions that can arguably constrain and direct the problem. The baseline for the task is a simple logistic regression with a set of features, and I would argue that adding more feature could achieve good results at a fracture of cost and complexity.

2.1 Baseline Implementation

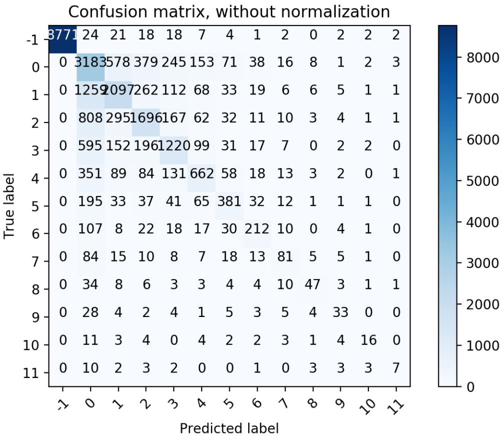
The new version of SQuAD 2.0 task adds a new constraint to competing question-answering models. In addition to identifying the extractive answer spans, a question-answering model should abstain from answering if the passage does not contain an answer. To this end, we implement a simple multinomial logistic regression classifier to address this task. At this level, the classification task is to predict the sentence, in the paragraph, containing the right answer, or declaring that the question is unanswerable. Detecting the correct answer span within the candidate sentence is not dealt with in the phase.

2.1.1 Feature Design

To find the sentence containing the answer, a classifier must determine the sentence that is most similar to the question, and by similar we mean that a good candidate sentence i) shares more words with the question. ii) has high cosine similarity with the question. iii) shares syntactic similarity with the question. Thus, three main features have been selected to this end:

- **Cosine Similarity**: for every sentence in the paragraph as well as the question a word vector representation is created via InferSent [Conneau et al, 2017], which is a pre-trained sentence embeddings method that provides semantic representations for English sentences. Cosine distance score is calculated for each sentence-question pair.

- **Word Overlap**: this calculates the Jaccard score between each sentence-question pair. Jaccard index is a method of computing the



	-1	0	1	2	3	4	5	6	7	8	9	10	11
-1	8771	24	21	18	18	7	4	1	2	0	2	2	2
0	0	3183	578	379	245	153	71	38	16	8	1	2	3
1	0	1259	2097	262	112	68	33	19	6	6	5	1	1
2	0	808	295	1696	167	62	32	11	10	3	4	1	1
3	0	595	152	196	1220	99	31	17	7	0	2	2	0
4	0	351	89	84	131	662	58	18	13	3	2	0	1
5	0	195	33	37	41	65	381	32	12	1	1	1	0
6	0	107	8	22	18	17	30	212	10	0	4	1	0
7	0	84	15	10	8	7	18	13	81	5	5	1	0
8	0	34	8	6	3	3	4	4	10	47	3	1	1
9	0	28	4	2	4	1	5	3	5	4	33	0	0
10	0	11	3	4	0	4	2	2	3	1	4	16	0
11	0	10	2	3	2	0	0	1	0	3	3	3	7

Figure 1: Confusion matrix shows which classes were predicted correctly

class	**precision**	**recall**	**F1-score**	**support**
-1	1	0.99	0.99	8872
0	0.47	0.68	0.56	4677
1	0.63	0.54	0.58	3869
2	0.62	0.55	0.58	3090
3	0.62	0.53	0.57	2321
4	0.58	0.47	0.52	1412
5	0.57	0.48	0.52	799
6	0.57	0.49	0.53	429
7	0.46	0.33	0.38	247
8	0.56	0.38	0.45	124
9	0.46	0.37	0.41	89
10	0.48	0.32	0.39	50
11	0.35	0.21	0.26	34
avg / total	0.72	0.71	0.71	26013

Table 1: This table shows the results of running a multinomial regularized logistic regression. Class column represents the index of sentences within the paragraph, and -1 represent the unanswerable question case.Unpredicted Classes are removed.

explicit similarity between two sets as follows:

$$J(Q,S) = \frac{|Q \cap S|}{|Q \cup S|}$$

where Q and S are sets of words in question and sentence respectively.

- **POS Overlap**: This feature computes the Jaccard score over the part-of-speech-tag representation of sentences. In other words, instead of the word tokens, it checks similarity over POS tokens.We use the default POS-tagger in the Spacy library of Python programming language to obtain the POS representation for the sentences and questions alike.

Using the three features above every question-sentence pair will have three scores and an additional binary feature indicating whether or not the question is answerable. This latter feature is provided by the dataset.

2.1.2 Training and Result

We train a logistic regression classifier with L2 regularization (‘newton-cg’ solver) using scikit-learn library of Python programming language, and we get the results shown in table 1. Numbers in class column represents the index of the sentence in paragraph containing the answer, and -1 indicates that the question has no answer in the paragraph. We also limit the number of sentence to 10. The results show that with simple features we get an F1 score of 0.71.