SI630 Project
Bi-Directional network and Attention Reader Models for Question Answering on the CNN Dataset

Yuqing Xia, Mei Fu
{felizxia, meif}@umich.edu

Abstract

Question Answering is becoming more and more popular in recent years, while statistical and neural network models have been implemented in this NLP task, advanced sequential models like LSTM and attention based network reached the highest performance. This paper reimplemented those state-of-art machine learning architectures, including Attention Sum reader model by (Rudolf Kadlec and Kleindienst, 2016) and Bi-directionally Attention Flow models by (Sordoni et al., 2016) on CNN dataset. The goal is to compare those models' performance on different scale of the dataset and try different vector representation methods. Our baseline model - logistic regression achieved 41% accuracy, while the best deep learning approach achieved 67.2% accuracy using 175,000 training data with option sum reader model.

1 Introduction

As machine comprehension is gaining popularity recently, the path to highly intelligent chat bot still left with a long journey to go. The top goal of this research is thus by implementing advanced network algorithms to teach machine better understand text meanings, and thus serve as the foundation of chat bot.

News as one of the most dominant information resources for the public,training machine to understand news according to user queries and news contexts can help people in different fields find facts and capture new events easily. Training models on such dataset covers variety of topics in society and thus help the machine to understand texts fundamentally. With the development of other dataset in the future, we can feed more domain-specific data into our current model to serve for different purpose, for example, building an economic information retrieval system.

As one of the challenges of question answering system is long text understanding and captures the relationship between news/contexts between questions/queries, thus the main goal of this is to implement the state-of-art attention models, and consider it's reliability in different task. We compared time cost, model complexity cost with final model accuracy on both validation dataset and test dataset. We concluded that although Bidirectional Attention Flow may receive higher accuracy, but it requires 12 hours for one epoch on a 89,000 dataset, and thus is as not applicable as attention sum reader, where it can train one epoch using only around 40 minutes, and reached 66.9% accuracy after 34 epochs using only 89,000 training data.

We also observe that there is a slightly difference when calculating entities probabilities in news using sum method or average method. When using a small model like attention sum reader with only one layer of GRU, sum method converges quickly and part of the reason is that in CNN dataset, entities appeared frequently have higher probabilities of being selected as the correct answers. While for bidirectional attention flow model, if we use sum method it will have over-fitting problems after 5 epochs, and we found train accuracy reaches around 80% of accuracy while validation accuracy drops to 62% of accuracy. Thus we use mean attention reader method for our bidirectional attention flow model and used sum method for our simpler model.

2 Problem Definition and Data

Question answering system is a simple evaluation of how machine understand the complex relationship between the user queries and contexts. A successfully implementation of QA system can improve information retrieval performance and en-

able users to get concise results back with less filtering efforts. QA is also a start point of training a chat bot, as an intelligent chat bot would be a successfully QA that posses larger memory network to enable the conversation continue logically. Exploration on implementing different models and feature engineering on QA can benefit future research on chat bot.

This project aims at applying machine learning and deep learning techniques to teach machine to perform question answering task. The evaluation matrix we will use accuracy. Currently, the Attention Sum Reader model developed by (Rudolf Kadlec and Kleindienst, 2016) reached 69.5% accuracy on the CNN dataset, the model developed by (Chen et al., 2016) reached 73.6% accuracy, and the BiDAF model developed by (Seo et al., 2017) reached 76.9% accuracy.

The datasets we are going to use are the CNN dataset released by Google DeepMind. The CNN dataset is one of the largest open source QA dataset. It contains over 90,000 CNN news, averagely has 4 queries (highlights) per story, which gives 1.2mm story-question pairs. The vocabulary size is 120,000 and 210,000. The proportion of train/dev/test is about 9:1:1. This dataset focuses on the noun entity recognition. One interesting attribute of this dataset is that it anonymized the noun entities (e.g. @entity01) to force the model to learn from the context instead of the entity itself. Also, the news stories provide more sufficient background information compared with other dataset such as SQuAD, which normally contains only several sentences.

The CBT is also a very popular QA dataset, which contains around 700,000 questions in the training set. We are going to compare the model performance on both datasets.

3 Related Work

While traditional question answering systems reply on hand labeled dataset to do classification task, neutral question answering method nowadays has developed an alternating attention mechanism (Sordoni et al., 2016) by setting up encoding phase of both memory of document and query, and creating steps pf inference to explain the semantic relationship between documents and queries, and then used the updating attention weights to find the weights that maximized proba-

Story	@entity0 , @entity1 (@entity2) @entity3 lure @entity5 and @entity6 migrants by offering discounts to get onto overcrowded ships if people bring more potential passengers , a @entity2 investigation has revealed . a smuggler in the @entity1 capital of @entity0 laid bare the system for loading boats with poor and desperate refugees , during a conversation that a @entity2 producer secretly filmed ……
Query	@placeholder investigation uncovers the business inside a human smuggling ring
Answer	@entity2
Words	@entity3:Smugglers @entity2:CNN @entity1:Libyan @entity0:Tripoli ……

Figure 1: Sample CNN-DM Story-Question Pair

Story	1 So they had to fall a long way . 2 So they got their tails fast in their mouths . 3 So they could n't get them out again . 4 That 's all . 5 `` Thank you , '' said Alice , `` it 's very interesting ……
Query	`` Boots and shoes under the sea , '' the XXXXX went on in a deep voice , `` are done with a whiting ''.
Candidates	Alice|BOOTS|Gryphon|SHOES|answer|fall|mouths|tone|way|whiting
Answer	Gryphon

Figure 2: Sample CBT Story-Question Pair

bility of finding the correct answer.

(Xiong et al., 2016) used dynamic memory network that is able to jump out of the local optima by comparing potential answer spans in that document. More specifically, the DMN they developed is able to iteratively estimate the start and end point of the potential answer spans.

While those two articles, one test their dataset on cloze style test set (CNN) and another one outperforms on Stanford Question Answering Dataset (SQuAD), (Seo et al., 2017) Bi-directional Attention Flow model standout in both data sets. Their memory less attention system only compute attention between query and context bi-directionally at the same time stamp without taking attentions in previous time stamp into consideration. By doing so, the attention layer can focus on learning interactions between query and context and negative results from previous time stamp will then have limited effect on the attention weights. Also their attention weights can flow from sequential layers without summarizing context words into fixed vector can avoid unnecessary information loss as well.

4 Methodology

4.1 Baseline Model

For conventional baseline model, we chose logistic regression for this multi-class prediction task. The reason we want to choose these is that, first, we have a very large dataset, so we want to choose a simple and computational saving models as a baseline; the second reason is that we want to understand our datasets better by picking features and see their performances.

We also relabeled entities name according to their appearance in both questions and answers dataset in order to make the labeling more consistent.

With reference to the results from (Chen et al., 2016) of their most important features, we tested on our datasets with different feature filter systems, we came up with 4 features that lead to highest priority.

Our features include:

1) Frequency

By counting all entities' frequencies in the news sets, we found that words that be chosen as target entity has average of 8 times of appearance in the context. Thus we added this as one of our key features.

2) First index location

As the driven by the writing norm of news, content that appear before are more important as word that appear later. Although this not always true, but 80% of entities show up in the 1/3 of the whole contexts.

3) Bi-gram Exact Match

Since there is some uncertainties when predicting candidate sentences matching clues in the questions, we also find that there are some exact match patterns, where tokens at the end or around (2 indexes around) the matched results have high probabilities as correct answers, thus we point those candidates entities as 1 and others as 0.

4) Distance

Since it is very hard to implement semantic practices in logistic regressions, and exact bi-grams are too strict to create results, we also create a feature called distance that records the minimum distance between each entities in current news set each entities or clues in the question data. After observing our dataset, we found that entities that have distance between 1 - 10 tend be a correct answer.

We add all feature values and normalize them for every entity, so every entity has distinct combined features values and the output of logistic regression is a binary value.

4.2 Deep Learning Approaches

Besides the baseline model, we decided to use 2 dominant methods to train our question answering system bidirectional attention flow model and option sum reader model. We will reference methodology from (Seo et al., 2017) and (Rudolf Kadlec and Kleindienst, 2016), but basically option sum reader model is simpler version compared to bidirectional attention flow model, but we want to include this model to tune parameters and experimenting different strategies of entities vector representation and news length truncating lengths and whether to context window around entity as contextual embeddings of entities.

4.3 Data Pre-processing

For the deep learning models, we used a pre-trained GloVe 100-dimension word embedding. As for the character-level embedding, we tried two

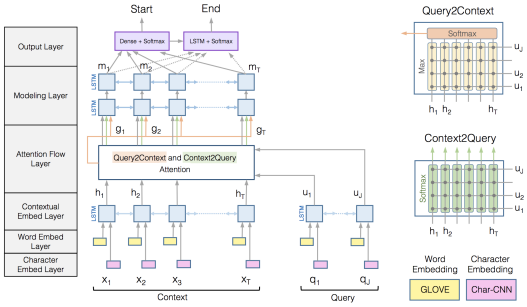

Figure 3: Bi-Directional Attention Flow Model

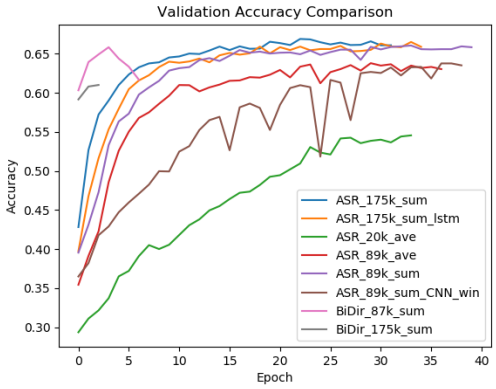

Figure 4: Validation Accuracy

different approaches and compared their performance. One is to use Word2Vec to train another 25-dimension character embedding on our train set, and take average of the character embedding to represent a word. Another approach is to use CNN to extract features during training. For CNN, each word is truncated or padded into 15 characters, and we applied 1D filters 3, 4, 5 to learn different lengths of characters. We used concated max pooling results to represent a word.

We kept top 20,000 words and label the rest of words as 'UNK'. Similarly, we have 'UNK entities'. Since we used character embedding for entities, for example, 'entitity 1' will have character embeddings for 'e','n'.. till '1', and we average each character's embedding values to create it's final 'character embedding' values. We also compared the performance of using CNN to extract character-level features, however it does not work fundamentally different with original embedding concatenate method and it causes extra time to train the model.

Before implementing the models, we selected 10,0000 news as our training data, and we truncated each news to a fixed length of 300 words, and each questions to 46 words long. We also filtered training data if the answer not show up in the 300 words length, and we ended up with around 80,000 news as our training dataset.

4.4 Bi-Directional Attention Flow Model

After data preprocessing, the next step we used two bidirectional LSTM with input from word and character embedding layer on context words and queries words, thus we obtained H hidden states values for context words and U hidden states values for queries word. Similar to embedding layer, the contextual embedding layer is still encoding words according to context in order to better suit for longer text representation.

We then encoded our feature vectors with the similarity matrix between the context and query as attention vectors, then we combined the contextual embedding vectors which captures the context word interactions,with this attention vectors, where context words interactions with queries have been encoded, we can have each column vector in this combine vectors have corresponding query-aware weights for each context word.

In the end, we used another two bidirectional LSTMs with inputs from our attention layer, thus we can get each context word representations conditioned on current query/question and on it's current paragraph/news.

Finally, we need an output layer to predict which entity is the correct answer. Although it's easy to implement a softmax layer in order to predict every word probability of being the desire answer, since we only care about which options is the correct answer, thus we added a masking layer called option layer, where we only input entities' character embeddings values. Thus we ended up having only each options' probabilities in the last step. Because each entity may appear multiple times, we sum up or average each entities probabilities in every location as their final probabilities and then use L1 normalization method to normalize them as output. Thus the cost of this model is each entities probabilities with their corresponding labels.

5 Evaluation and Results

Here, we examined the models, accompanied with different tunning parameters and embedding struc-

Model (Train Size) \ Accuracy	Train	Validation	Test
Logistic Regression (87k)	-	38	41.4
BiGRU + Attention x 2 (20k) (simplified R-Net)	46	-	-
BiGRU + Attention + Masking (ave) (20k)	60.5	54.6	54.9
BiGRU + Attention + Masking (ave) (87k)	64.1	63.8	63.6
BiGRU + Attention + Masking (sum) (87k)	67.3	66.1	66.9
CNN +(win)+ BiGRU + Att + Masking(sum) (87k)	62.9	63.8	64.7
BiGRU + Attention + Masking (sum) (175k)	65.2	66.9	67.2
BiLSTM + Attention + Masking (sum) (175k)	66.3	66.5	67.1
Bidirectional_Attention + Masking (87k)			
Kadlec et al. (2016)[1]	-	68.6	69.5
Chen et al. (2016)[2]	-	73.8	73.6
Minjoon et al. (2017)[3]	-	76.3	76.9

Figure 5: Test Results

tures, including:

1) Training size of 13,700 versus training size of 89,000 versus training size of 175,000.

2) Word2Vec character-level embedding versus CNN character-level embedding.

3) Sum versus average method to calculate probabilities for each entity

4) LSTM model versus GRU model.

We found that Attention Sum reader with 175,000 training data have the highest validation and test accuracy.

6 Discussion

The main findings of the project are that we tried two state-of-art models and understand the key components of model that can produce most accurate results.

First of all, our logistic regression model only reached 38% of accuracy on validation data and 41% of accuracy on test data. The main reason is that we did not create enough variables for logistic regression to solve this complex problems. We could have added dependency parse match, sentence co-occurrence and other variables in the model to see accuracy changes.

For deep learning models, the first challenge is that the model complexity strongly affects training efficiency. We could have tried adding more data in Bi-directional Attention Flow model with 175,000 training data in order to prevent over-fitting issues, however it will take around 28 hours for a Mac with 16G memory to train this model per epoch, and thus we decided to experiment tuning mostly on Attention Sum Reader model.

We found out that CNN character embeddings did very trivial impact on entity representation as compared to contextual embedding averaging character-level Word2Vec to represent a word. Actually the performance went down after we used the CNN character level embedding.

We also found that average attention pointer method is more suitable for complex models like Bi-directional Attention Flow, while sum attention pointer converges quickly with one layer GRU with attention layer model. The main reason here is that since we trained a relatively smaller dataset(89,000) on the complex model, and thus it tends to over-fit and have higher probability to pick entities that appear more frequently as candidate answer while it is not always true on validation and test dataset. Thus sum attention pointer method will lead to over-fitting problems under this circumstances.

What's more, the simpler GRU layer outperformed LSTM layer on the Attention Sum Reader in terms of both speed and accuracy, although LSTM outperformed GRU on the Bi-directional Attention Flow.

If there is enough computation resources for us, we will further compare Bi-directional Attention Flow with mean attention pointer on a larger training data to see if it can outperform our current model.

7 Other Things We Tried

For the logistic regression, we also tried to use local embedding to select the words having high similarity with "@placeholder", but turns out that adding this feature will only hurt the performance.

Also, we tried multi-classification for logistic regression, which means predicting one answer out of 307 candidates, but the accuracy is only around 7%, so we turned to binary classification.

For deep learning models, we tried to split each news/context into 19 words long sentences that are centered with entity. By doing so, our original input expanded from 300 words to 1902 words because there are lots of overlapping words included when applying the context window. When including this expanded inputs to our cur-

rent model, unfortunately, the computer memory crashes because of memory issue.

We also tried to filter our 300 words long news input only including sentences with entities, however this to some extent missed the contextual information in news, thus performed worse than original dataset.

8 Future Work

In this project, we compared the performance between deep learning approach and the logistic regression baseline, as well as the contribution of different parts of the deep learning architecture. Constraint by computation resources, the complex deep learning model did not beat the performance of simpler deep learning model. Given more computation power, we will be able to study more complex models with more data.

9 Group Effort

Yuqing:

Model source code adjustment; Run bidirectional attention flow model; Create dataset for context window inputs.

Mei:

Model source code adjustment; Run R-net and ASR hyper-parameter tuning; Pre-process dataset.

Acknowledgments

We are thankful to Dr. David Jurgens and Rohail Syed for their excellent guidance during the office hour and through out the semester.

References

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task .

Martin Schmid Ondrej Bajgar Rudolf Kadlec and Jan Kleindienst. 2016. Text understanding with the attention sum reader network .

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ICLR* .

Alessandro Sordoni, Phillip Bachman, and Yoshua Bengio. 2016. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245* .

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604* .