Coattention-Based Neural Network for SQuAD Question Answering

Xizhi Han
Department of Physics
Stanford University
hanxzh@stanford.edu

Yue Hui
Department of Mathematics
Stanford University
yueh@stanford.edu

Abstract

Reading comprehension has long been a challenging task in natural language processing, possibly because semantic relations between contexts and questions can be complicated. Recent release of SQuAD has seen increasing efforts in achieving better performance on this problem. In this report, we combine ideas from two high-performing SQuAD models, namely the Dynamic Coattention Network and the DrQA, and implement a model achieving 72.2% F1 and 61.7% EM on SQuAD test set. Performance of our model and effect of different features is analyzed in detail.

1 Introduction

Our central focus in this report will be reading comprehension on SQuAD [4] with deep learning. The SQuAD dataset has three splits: 80% data in the train set, 10% data for dev and 10% unpublished data for test. Each sample consists of a context paragraph (which is an excerpt from Wikipedia) and a question whose answer is contained in the context. The algorithm is asked to find location of the answer in the context. A large variety of types of questions and topics are covered in the dataset, so the SQuAD challenge provides a platform to compare how well computer systems perform in understanding natural language texts, which can be useful in real life situations such as automatic question answering in natural languages and digital personal assistants.

There has been much literature tackling this task. Our method is mostly inspired by the Dynamic Coattetion Network [2], of which two essential novelties are the coattention layer, which encodes interaction between question and context, and the dynamic pointing decoder, which decides answer spans iteratively. We also gain much intuition from [3], where its Document Reader model describes many interesting features to be implemented. It is also exciting to see that recent models are achieving F1 scores (approximately 89%) comparable to human performance (approximately 91%). We refer readers to https://rajpurkar.github.io/SQuAD-explorer/ for more details.

The rest of this report is organized as follows: section 2 describes the model and features we have implemented, and its training details. In section 3 we analyze the results and errors, and show effectiveness of our feature implementations. Section 4 is the conclusion.

2 Experiments

In this section, we introduce our question-answering model with details of implemented features. Processes of training and tuning hyperparameters are presented as well.

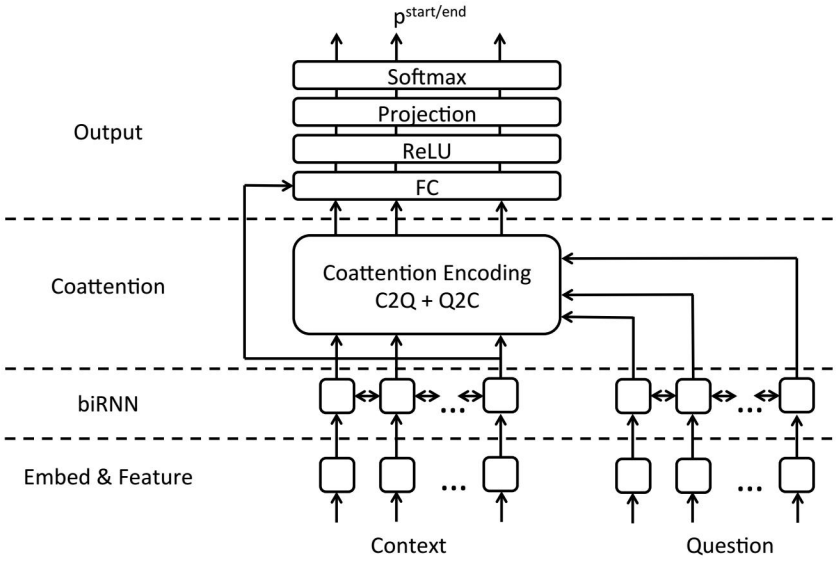

Figure 1: Architecture of our model.

2.1 Model Architecture

Following instructions in the "*CS 224N Default Final Project: Question Answering*" handout, we first performed a baseline model with an RNN encoder layer, an attention layer and an output layer implemented, which achieves an F1 score of 41.5 and an EM score of 32.7 on the course dev leaderboard. Then the simple attention layer in the baseline model is replaced by a coattention layer, and a feature vector is concatenated with GloVe word embeddings. Details of our model is as follows with an illustration in Fig. 1.

RNN Encoder Layer Our model is trained and tested on SQuAD where each examples is a (context, question, answer) tuple. The context is represented by a sequence of vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$ and similarly for the question $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M \in \mathbb{R}^d$. Each vector $\boldsymbol{x}_i$, $\boldsymbol{y}_j$ is a concatenation $[\boldsymbol{e}; \boldsymbol{f}]$ of a 300d GloVe embedding $\boldsymbol{e} \in \mathbb{R}^{300}$ and a binary feature vector $\boldsymbol{f} \in \{0, 1\}^{48}$, which will be elaborated in the next subsection.

The vectors are then fed into a one-layer bidirectional GRU (with shared weights). The forward hidden states $\overrightarrow{\boldsymbol{c}_i} \in \mathbb{R}^h$ and $\overrightarrow{\boldsymbol{q}_j} \in \mathbb{R}^h$ and the backward hidden states $\overleftarrow{\boldsymbol{c}_i} \in \mathbb{R}^h$ and $\overleftarrow{\boldsymbol{q}_j} \in \mathbb{R}^h$ produced by the GRU are then combined to obtain the context hidden states $\boldsymbol{c}_i = [\overrightarrow{\boldsymbol{c}_i}; \overleftarrow{\boldsymbol{c}_i}] \in \mathbb{R}^{2h}$, $i = 1, \ldots, N$ and the question hidden states $\boldsymbol{q}_j = [\overrightarrow{\boldsymbol{q}_j}; \overleftarrow{\boldsymbol{q}_j}] \in \mathbb{R}^{2h}$, $j = 1, \ldots, M$ respectively.

Coattention Layer First a pair of sentinel vectors $\boldsymbol{c}_\emptyset, \boldsymbol{q}_\emptyset \in \mathbb{R}^{2h}$ are added into the list of context hidden states and question hidden states respectively and denote $\boldsymbol{c}_{N+1} = \boldsymbol{c}_\emptyset$, $\boldsymbol{q}_{M+1} = \boldsymbol{q}_\emptyset$. Define the affinity matrix:

$$\boldsymbol{L}_{ij} = \boldsymbol{c}_i^T \boldsymbol{q}_j \in \mathbb{R}, \quad i = 1, \ldots, N+1, \quad j = 1, \ldots, M+1,$$

and the Context-to-Question (C2Q) attention distribution α^i and outputs $\boldsymbol{a}_i$ for $i = 1, \ldots, N+1$ are

$$\alpha^i = \text{softmax}(\boldsymbol{L}_{i,:}) \in \mathbb{R}^{M+1}, \quad \boldsymbol{a}_i = \sum_{j=1}^{M+1} \alpha_j^i \boldsymbol{q}_j \in \mathbb{R}^{2h}.$$

Similarly compute the Question-to-Context (Q2C) attention distribution β^j and outputs $\boldsymbol{b}_j$:

$$\beta^j = \text{softmax}(\boldsymbol{L}_{:,j}) \in \mathbb{R}^{N+1}, \quad \boldsymbol{b}_j = \sum_{i=1}^{N+1} \beta_i^j \boldsymbol{c}_i \in \mathbb{R}^{2h},$$

where j runs from 1 to $M+1$. Next the second-level attention outputs are evaluated

$$\boldsymbol{s}_i = \sum_{j=1}^{M+1} \alpha_j^i \boldsymbol{b}_j \in \mathbb{R}^{2h},$$

where, however, we are interested in only $i = 1, \ldots, N$. Finally we feed the sequence $[\boldsymbol{s}_i; \boldsymbol{a}_i]$ through a bidirectional GRU to get the coattention encoding $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N\} = \mathrm{biGRU}(\{[\boldsymbol{s}_1; \boldsymbol{a}_1], \ldots, [\boldsymbol{s}_N; \boldsymbol{a}_N]\})$ where $\boldsymbol{u}_i \in \mathbb{R}^{2h}$.

Output Layer The coattention encodings $\boldsymbol{u}_i$ are combined with the context hidden states $\boldsymbol{c}_i$ into blended representations $\boldsymbol{d}_i = [\boldsymbol{c}_i; \boldsymbol{u}_i] \in \mathbb{R}^{4h}$, which is then fed into a fully connected layer followed by a ReLU nonlinearity, i.e., $\boldsymbol{d}'_i = \mathrm{ReLU}(\boldsymbol{W}\boldsymbol{d}_i + \boldsymbol{v}) \in \mathbb{R}^h$. Start/end logit for each context location i is computed through a linear projection layer $\mathrm{logits}_i^{\mathrm{start/end}} = \boldsymbol{w}_{\mathrm{start/end}}^T \boldsymbol{d}'_i + u_{\mathrm{start/end}} \in \mathbb{R}$. Finally the probability distribution of start/end locations of answers is $p^{\mathrm{start/end}} = \mathrm{softmax}(\mathrm{logits}^{\mathrm{start/end}}) \in \mathbb{R}^N$. And our loss function is the sum of the cross-entropy loss for the start and end locations.

Prediction At test time, given a context and a question, the answer span is predicted as (l, r) such that $0 \leq r - l \leq 15$ and $0.95^{r-l} p_l^{\mathrm{start}} p_r^{\mathrm{end}}$ is maximal. Here 15 and 0.95 are hyperparameters tested to give the best performance on the dev dataset. Intuitively this makes succinct answers favorable (and empty answers impossible).

2.2 Feature Engineering

Coattention networks yield high performance even without additional features; however in this project we hope to investigate how extra inputs could still boost its performance. Effectiveness and tradeoffs are discussed with examples and data in the next section; here we simply introduce the additional features that we have implemented.

Exact Match For each context location $i = 1, \ldots, N$, $t_i = 1$ if the context word at location i exactly appears in the question; otherwise $t_i = 0$. This helps our model better recognize appearances of questions in the contexts.

Embedding Refinement It is observed that GloVe word embeddings don't differentiate interrogatives very well, possibly because they appear in similar semantic contexts and have similar syntactic functions. To improve let $s_{j,k} = 1$ if the word at question location j matches element k in the following list: ["what", "why", "how", "when", "who", "where", "which", "no", "not", "and", "or", "besides"], otherwise $s_{j,k} = 0$. Hence $\boldsymbol{s}_j$ is a binary vector of length 12.

Part-of-Speech Tag We have also used NLTK 3.2.5 to assign to each word in the context or question one of the thirty-five Part-of-Speech tags, so that our model can understand structure of sentences better. This yields a binary vector $\boldsymbol{p} \in \{0, 1\}^{35}$ indicating which category the word lies in.

For context words, the first and the last features are concatenated into a feature vector $\boldsymbol{f}_i = [t_i; \boldsymbol{0}; \boldsymbol{p}_i]$ and for question words, the second and the last features are combined $\boldsymbol{f}_j = [0; \boldsymbol{s}_j; \boldsymbol{p}_j]$. Zeros are padded so that feature vectors for context and question words have the same length $(= 48)$.

2.3 Training and Tuning

Our model is trained on a Microsoft NV6 Standard Azure VM. We have used an Adam optimizer with initial learning rate 0.001 and gradients clipping with the maximal gradient norm 5.0. Contexts are truncated to length 400 and questions to length 30 if they are too long. Batch size is 100. In the following we focus on three hyperparameters: whether to have sentinels, hidden size h and dropout rate (used in biGRUs).

Sentinels To better understand the role of sentinels, we have tried different initialization of sentinel variables (xavier or zero) and they show no significant (≈ 0.1) difference in final F1/EM scores. However if sentinels are disabled, F1/EM scores drop by about 4. Visualizing C2Q attentions, we see sentinels are necessary for contexts to attend to none of the question words, which reduces error if most of the context is irrelevant.

Overfitting Deep learning should be capable of learning end-to-end; however we think feature engineering is still important because of the overfitting problem, at least when data is limited. In Fig. 2 we have plotted loss curves of models with or without feature vectors and with different hidden sizes and dropout rates. The figure on the left shows training process of the original model with feature vectors. In the middle figure, we give up feature engineering and hope that with more neurons (a larger hidden size) the model could discover those features by itself; however this is not the case because overfitting becomes more severe as there are more parameters. Our tactic is then to increase the dropout rate in the right figure, where indeed overfitting is cured however because the effective number of neurons is decreased the model gets harder to fit the data. As a result the final F1/EM scores are not improved. In this toy case the model with feature embeddings yields better performance than the model simply with more hidden neurons, who struggles with overfitting.

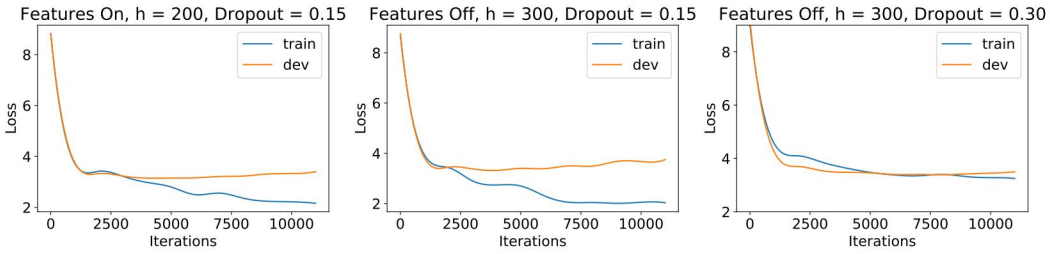

Figure 2: Train/dev loss curves of models with/without feature vectors, with different hidden sizes and dropout rates.

3 Results and Analysis

Our model reaches an F1 score of 71.9 and an EM score of 60.9 on SQuAD dev dataset and F1 of 72.2 and EM of 61.7 on test set. A comparison with other high-performance models on the leaderboard is summarized in Tab. 1. Our F1 is slightly lower than the original dynamic coattention network possibly because the dynamic pointer decoder is not implemented. Performance of our model with features turned off is presented in Tab. 2 to show effectiveness of the features.

Table 1: Comparison with other models (test).

Model	EM	F1
Human Performance	82.304	91.221
QANet+ (single)	82.209	88.608
R-net+ (single)	79.901	86.536
Dynamic Coattention Networks (single)	66.233	75.896
Our Version of Coattention Model (single)	61.733	72.187
Match-LSTM with Ans-Ptr (sentence)	54.505	67.748

Behavior of our model and effects of additional features will be analyzed in four scenarios that we think are important and exemplary:

3.1 Long Answers

Our model is having a hard time finding long answers, for example,

Table 2: Comparison of our model with different features (dev).

Model	EM	F1
Original	60.9	71.9
Without Embedding Refinement	60.5	71.4
Without Exact Match	59.9	71.0
Without PoS Tag	57.7	69.3
No Features	58.0	69.5

- **Context** 20th century fox , lionsgate , paramount pictures , universal studios and walt disney studios paid for movie trailers to be aired during the super bowl . fox paid for deadpool , x-men : apocalypse , independence day : resurgence and eddie the eagle , lionsgate paid for gods of egypt , paramount paid for teenage mutant ninja turtles : out of the shadows and 10 cloverfield lane , universal paid for the secret life of pets and the debut trailer for jason bourne and disney paid for captain america : civil war , the jungle book and alice through the looking glass .
- **Question** paramount paid fo , 10 cloverfield lane and which other film trailer to be aired during the game ?
- **True Answer** teenage mutant ninja turtles : out of the shadows
- **Predicted** super bowl

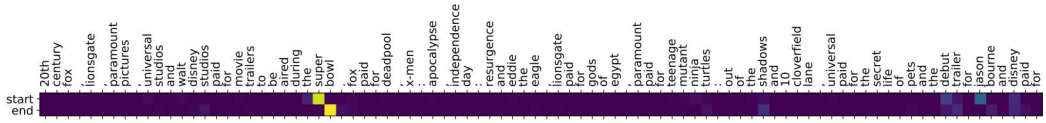

Figure 3: Start and end location probability distribution predicted for the example.

At first we guess this may be the result of our prediction strategy, which penalizes long answers. However, this is not the case, as indicated by predicted start and end location probability distributions in Fig. 3, where the probability is significant only near the wrong answer "super bowl". We think in this example our model cannot recognize the true answer as the name of a movie in parallel with "10 cloverfield lane" in the question, because it is lengthy for an RNN and with punctuations in it.

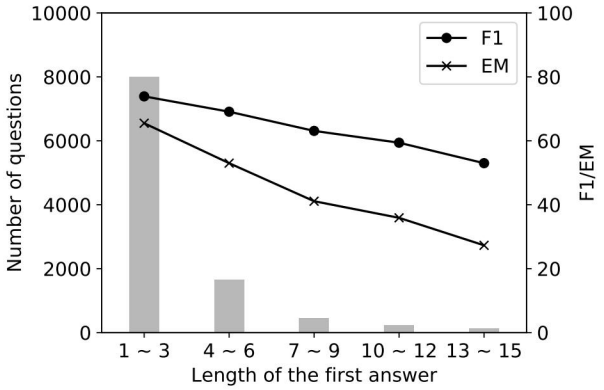

Figure 4: Averaged F1/EM scores of our model versus length of the true answer (up to 15 tokens).

To be more quantitative the effect of answer length on prediction accuracy is plotted in Fig. 4, where a consistent drop of accuracy due to increasing answer length is indeed observed. Note the maximal possible answer length predicted by our model is 15 tokens. Our additional features cannot help

with this problem, which possibly requires different model architectures. We should also note that this effect might also be due to the fact that there are fewer data points with longer answers.

3.2 Imprecise Spans

Another major source of accuracy loss is imprecision in predicting start/end locations. As an example in Fig. 5 we have plotted imprecision (distance from true locations) of predicted start locations of answer spans. Note there is a significant improvement in predicting exact start locations (distance 0 in the figure), after enabling features. So indeed features are helping the model better recognize beginning of answer spans.

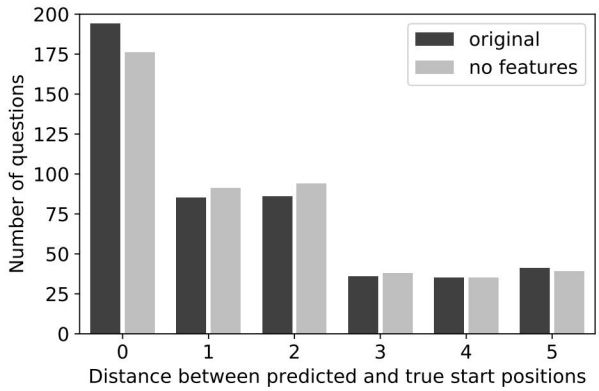

Figure 5: Number of predictions versus minimal distance between predicted start location of the answer span and that of true answers in the dev dataset, for our model (original) and our model without feature vectors (no features).

To gain more intuition C2Q attention distribution of the following example is visualized in Fig. 6.

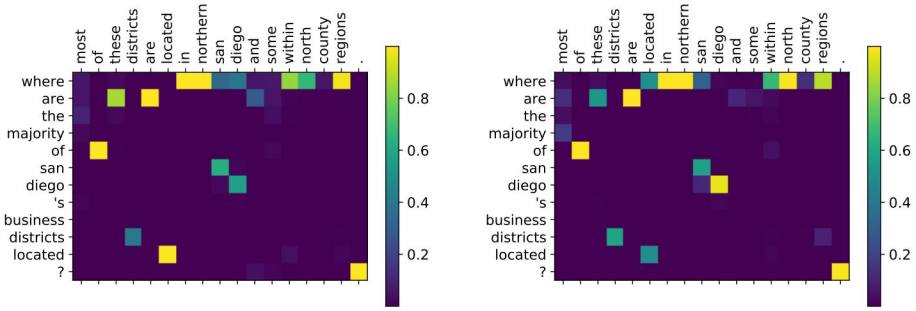

Figure 6: C2Q attention distribution of our model without (left) and with (right) feature vectors.

- **Context** downtown san diego is the central business district of san diego , though the city is filled with business districts . these include carmel valley , del mar heights , mission valley , rancho bernardo , sorrento mesa , and university city . most of these districts are located in northern san diego and some within north county regions .
- **Question** other than its main central business district , where are the majority of san diego 's business districts located ?
- **Predicted** (without features) northern san diego and some within north county regions
- **Predicted** (with features) northern san diego

From attention visualization it is observed "diego" in the context attends more to "diego" in the question after enabling features, which may be a result of the exact match feature. So the model

can focus more on the question and know where to stop. It is also interesting to observe that with features "located" and "north" in the context attend more to "where" in the question, which we think is because of the embedding refinement. Overall we think features provide more information of relevance between contexts and questions so that attention can be more focused.

3.3 Question Types

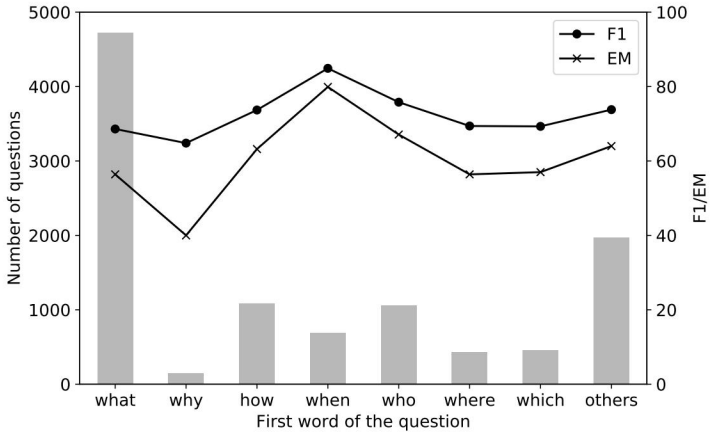

Figure 7: Averaged F1/EM scores of our model on different types of questions (dev).

Precision of our model on answering different types of questions (starting with "what", "why", etc.) is also different. As is shown in Fig. 7, our model gives best predictions on questions starting with "when", possibly because numbers (times, dates, years, etc.) are easier to identify. Our model perform least satisfactorily on questions starting with "why". Indeed, these may be the difficult questions that require some reasoning. Overall, all F1 scores are above 60 so there is no significant weakness.

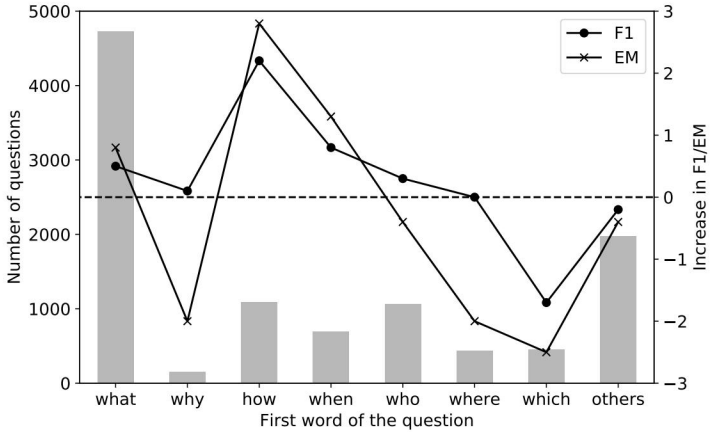

Figure 8: Increase in averaged F1/EM scores after adding the embedding refinement feature, on different types of questions (dev).

As introduction of embedding refinement is aimed at improving capacity of dealing with interrogatives (see "feature engineering" subsection) it is natural to ask, how this additional feature affects model's performance on different types of questions? The results are summarized in Fig. 8, where increase means F1/EM of our model minus F1/EM of our model without embedding refinement. Except for "which", the F1 scores are improved and the F1 score for questions starting with "how"

(possibly "how many") is increase by 2.2! This is evidence that our embedding refinement feature is effective.

3.4 Difficult Questions

Finally there are some questions requiring thought even for humans. For example, focus on the example in paragraph "Long Answers" one more time. Now the question is

- **Question** what famous july fourth holiday movie did fox pay to advertise a sequel of during the super bowl ?
- **True Answer** independence day
- **Predicted** apocalypse

It would be impossible to know the answer without knowledge that the "july fourth holiday" is the "independence day". However, this information does not appear in the context so it is impossible for our model to discover this commonsensical connection.

Here is another difficult example, and its difficulty lies in understanding and comparison.

- **Context** southern california is home to many major business districts . central business districts (cbd) include downtown los angeles , downtown san diego , downtown san bernardino , downtown bakersfield , south coast metro and downtown riverside .
- **Question** what is the only district in the cbd to not have " downtown " in it 's name ?
- **True Answer** south coast metro
- **Predicted** central business districts

To correctly answer this question, one first has to know what "name" exactly means and then to compare different districts present in the context. Our model only contains one coattention layer, which may be insufficient for such comparing work.

4 Conclusion and Future Work

As the analysis shows, our implementation of coattention and features is effective in SQuAD question answering, but the model can get confused in some difficult situations, e.g., when the answer is long or when the answer requires some form of reasoning. To improve the model, we could implement the dynamic pointer decoder as well, which will help to recover from local maxima and improve predictions of start and end locations. Also, if we could obtain more data with lengthy answers, we might be able to obtain better training results on long answer spans.

Acknowledgments

We would like to thank Dr. Richard Socher, and all the TAs for a rewarding class experience. We would also like to thank Microsoft for providing GPU VM on Azure to train the models.

References

[1] Seo, M., Kembhavi, A., Farhadi, A. & Hajishirzi, H., Bidirectional Attention Flow for Machine Comprehension *arXiv preprint arXiv:1611.01603*, 2016

[2] Xiong, C., Zhong, V., & Socher, R. Dynamic Coattention Networks For Question Answering *arXiv preprint arXiv:1611.01604*, 2016

[3] Chen, D., Fisch, A., Weston, J., & Bordes, A. Reading Wikipedia to Answer Open-Domain Questions *arXiv:1704.00051*, 2017

[4] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text *arXiv:1606.05250*, 2016