Building End-to-End Question Answering Model With Reinforcement Learning

Eti Rastogi, Prashant Budania
Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
erastogi@andrew.cmu.edu, pbudania@cmu.edu

Abstract

We propose an end-to-end Question Answering model that can efficiently handle the task of passage selection. Given a query and group of passages related to it, the task is to select a passage such that it can correctly answer the given question. We reformulate the task of passage selection as that of a reinforcement learning task. We propose an agent which is responsible for picking up a paragraph from a set of paragraphs given for the question. We say that the agent was successful in the task if the paragraph picked by it contained the correct answer. We also use a baseline neural network model(BiDAF) for predicting the answer from the selected passage.

1 Introduction

Question Answering (QA) in the field of information retrieval or Natural Language Processing (NLP) refers to the process of building models for automatic answering of the queries posed by humans. A typical QA model takes a query as input and outputs an answer either selected from a knowledge base, or the passages/snippets associated with the query or uses generative models (for example: LSTMs/GRUs) to generate an answer. The Question Answering models can be divided into two broad categories: traditional NLP based models using modular approach or end-to-end trainable models using neural approach. In the traditional QA models, the entire pipeline consisted of modular methods including but not limited to query processing, information retrieval, passage selection or ranking, answer extraction or generation and reformulation, answer ranking etc. The pipeline is shown in Fig [1].

The other category is QA models based on neural networks. The main advantage of the neural-QA models is that they are more robust, domain independent and they don't require explicit feature engineering or special pre-processing and they are more adaptable across similar datasets and can

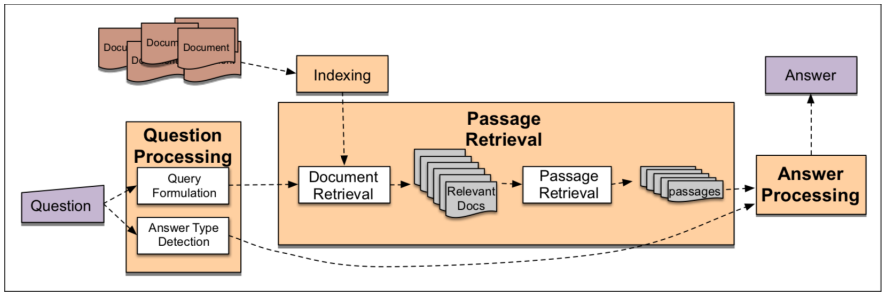

Figure 1: Traditional QA pipeline

even make use of knowledge from other domains compared to the traditional QA models. Our proposed method involves training such an end-to-end model.

One of the main disadvantages of a typical QA model is that it becomes expensive - both in computation and time, with increasing size of the context. Therefore we consider a hierarchical approach where we first select the relevant passage coarsely and then apply the QA module for answer-selection using the selected passage as context.

We treat passage selection as a latent variable trained jointly from the answer only using reinforcement learning. Therefore, we propose an agent that takes in a query and a group of related passages(current state) and chooses a relevant passage(action). The agent receives a positive award if the performance is good, i.e. if the selected paragraph helps in predicting the correct answer. We use REINFORCE algorithm [Wil92] to train our RL model.

To check the performance of our agent, we implemented a baseline question answering model based on bi-directional attention: Bi-Directional Attention Flow (BiDAF) model [Seo+16]. The BiDAF model takes in the original query and gives and answer based on the selected context. We evaluate our model on SearchQA dataset [Dun+17] , which is a collection of questions from the famous game show Jeopardy.

In the field of Natural Language Processing, people have shown limited success using Reinforcement Learning algorithms. More commonly, RL is used where the problem has latent variable(s) and where the reward function can be based on the configuration. In our case also, we use reinforcement learning for the passage ranking task to formulate the passage selection as a latent variable and give rewards on the accuracy of the selection. The introduction of a latent variable in the form of passage selection helps the Question Answering model work faster, since it needs to only observe a smaller context. Also, given a very good passage selection model, the training of the Question Answering model is more likely to lead to a more robust model. This is possible because for longer sentences, RNN finds it difficult to remember the interactions between the words. Therefore it has better ability to capture the meaning from smaller context and find the co-relation between context and query. Also, the lack of golden outputs(correct paragraphs) does not allow us to do direct supervision and hence RL helps by giving an indirect supervision with the help of rewards based on the likeliness to get correct predictions.

2 Related Work

The use of reinforcement learning for Question Answering models is very rare but there have been some recent papers that tries to reformulate the simple question answering pipeline in a manner that allows the use of RL to its advantage. For example, in [Cho+17], the authors use a reward based objective function to first do sentence selection and then generate an answer from that. Our reward function is inspired from their approach only where you get a positive reward if the selected sentence/passage helps you with answer selection/generation. In [Buc+17], the authors also use RL for question answering where the goal of the agent is to reformulate the query in such a manner to elicit the best possible answer. They call their approach: Asking the Right Questions and their error analysis also uses the BiDAF model as their baseline QA model and they were able to beat simple baseline models on SearchQA [Dun+17] dataset. In [WYT17], they use joint learning to do question answering and they were able to show that their model benefits from joint learning to do well on both tasks (query and answer generation).

3 Method

3.1 Dataset

SQuAD : Stanford Question Answering Dataset (SQuAD) [Raj+16] is one of the largest machine comprehension dataset available consisting of around 100,000 questions posted by crowdworkers on a set of Wikipedia articles. The dataset consists of multiple question-pargraph pairs divided into different topics. The answer to each question is always a span in the context. The dataset consists of 90k/10k train/dev split.

Table 1: General Properties of SQuAD

Average passage length	735.78
Average number of questions/passage	4.63
Average number of passages/topic	42.75
Average number of questions/topic	198.18
Average answer length	3.58

Table 2: General Properties of SearchQA

Average passage length	237
Average number of passages/question	50.6
Average number of passages with answers/question	16.78
Average answer length	1.5

SearchQA : SearchQA [Dun+17] is a dataset consisting of around 140k questions extracted from Jeopardy. Each question-answer pair has around 50.6 snippets. The data has been preprocessed by lower casing and removing stop words. The training, validation and test sets contain 99,820, 13,393 and 27,248 examples, respectively.

3.2 Model

BiDAF : Bi-Directional Attention Flow(BiDAF) is a hierarchical multi-stage architecture which models the paragraph representation at different levels of granularity. It performs attention in two directions to obtain a query-aware context representation. The attention is computed for every time step and is allowed to flow through the subsequent layers which reduces the information loss caused by early summarization. It also uses a memory-less attention mechanism, that is, attention at every time-step is independent of previous time steps. This helps the model to compute accurate attention at each time step even if it completely misjudges at previous time steps. The different layers in BiDAF model are :

1. **Embedding Layer(Character + Word)** : Let $x_1, ...x_T$ and $q_1, ...q_J$ represent the words in the input context paragraph and question, respectively. Each word is mapped to vector space by using character-level CNNs as well as pre-trained GLOVE embeddings[PSM14]. Character level embeddings are useful for handling rare or out-of-vocabulary words. Each character is represented as a 16-dimensional vector and fed to a CNN architecture with 100 filters and a kernel size of 5. The outputs of the CNN are max-pooled over the entire width to obtain a 100-dimensional vector for each word.

   The character and word embeddings are concatenated together to obtain a 200D vector for each word.
2. **Highway Layer** : The concatenated word vector is passed to a two-layer highway network which is a MLP. Highway networks allow the model to dynamical learn its depth. This is done by learning a gate T which controls how much of the output is produced by transforming the input and carrying it, respectively. The outputs of the Highway Network are two matrices: $X \in \mathrm{R}^{200\times T}$ for the paragraph/context and $Q \in \mathrm{R}^{200\times J}$ for the query.
3. **Contextual Embedding Layer** : The output of the highway network is further passed through a bidirectional LSTM to model the temporal interactions between words. Thus, we again obtain two matrices : $H \in \mathrm{R}^{200\times T}$ from the context word vectors and $U \in \mathrm{R}^{200J}$ from query word vectors.
4. **Attention FLow layer** : This layer is responsible for combining and connecting the information from the context and the query words. Unlike other models, the attention vector at each time step, along with the embeddings from previous layers, are allowed to flow through to the subsequent layers which helps in reducing the information loss caused by early summarization. We calculate a similarity matrix $S \in \mathrm{R}^{T\times J}$ where S_{tj} indicates the similarity between t-th context word and j-th query word. The similarity matrix is computed as :

$$S_{tj} = W^t[h_t; u_j; h_t \bullet u_j] \tag{1}$$

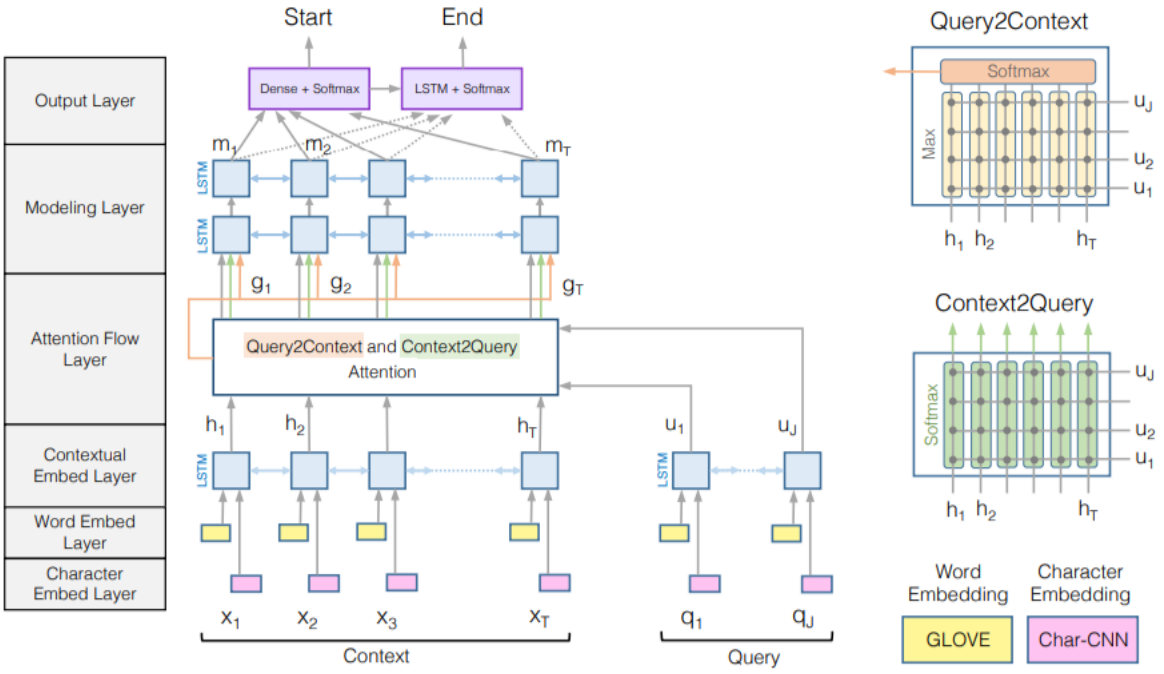

Figure 2: Bi-Directional Attention Flow model

where h_t represents the contextual embedding of t-th context word, u_t represents the contextual embedding of j-th question word, $\bullet$ is element wise multiplication and [;] is vector concatenation across row.

We use S to obtain attentions in two directions : from context to query as well as query to context.

Context-to-query attention : It helps to determine which query words are most relevant to each context word. Let $a_t \in \mathrm{R}^J$ represent the attention weights on the query words by t-th context word:

$$a_t = \text{softmax}(S_{:t}) \tag{2}$$

Thus, we obtain the attented question vector over t-th context word as :

$$c2q_t = \sum_j a_{tj} U_j \tag{3}$$

Hence, we obtain a matrix of attended question vectors $c2q \in \mathrm{R}^{200 \times T}$ for the entire context.

Query-to-context attention : This signifies which context words have the closest similarity to one of the query words and are hence critical for answering the query. The attention weights on the context words are obtained by :

$$b = \text{softmax}(\max_{col} S) \tag{4}$$

Then the attended context vector is :

$$q2c = \sum_t b_t H_t \tag{5}$$

This vector indicates the weighted sum of the most important words in the context with respect to the question and is tiled T times across the column, thus giving $q2c \in \mathrm{R}^{200T}$.

The output of this layer combines the contextual embeddings and the attention vectors to obtain $G \in \mathrm{R}^{800 \times T}$

$$G = [H; c2q; H \bullet c2q; H \bullet q2c] \tag{6}$$

5. **Modeling Layer** : The matrix G, output of the attention layer is fed to a 2 layer Bi-Directional LSTM to capture the interaction among the context words conditioned on the query. Hence we obtain a matrix $M \in \mathrm{R}^{200 \times T}$.
6. **Output Layer** : The function of this layer is to predict the location of correct answer from the context by predicting it's start and end index in the context. The probability distribution of the start index over the entire paragraph is given by :

$$p^1 = softmax(W^T[G; M]) \tag{7}$$

The output of the modeling layer M is again passed to a bidirectional LSTM to obtain $M^2 \in R^{200 \times T}$ which is used to obtain the probability distribution of the end index :

$$p^2 = softmax(W^T[G; M^2]) \tag{8}$$

7. **Training Details** : The loss function to be minimized is simply defined as the sum of the log probabilities of the start and the end indexes of each answer, averaged over all the examples:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{i=N} \log p^1_{y^1_i} + \log p^2_{y^2_i} \tag{9}$$

where N is the number of examples in the dataset, y^1_i and y^2_i are the true start and end indices of the i-th example, respectively.

Each paragraph and question is tokenized by using a NLTK tokenizer and fed into the model. We take a batch size of 60 and train the model on Adam optimizer with an initial learning rate of 0.5, for 12 epochs. A dropout rate of 0.2 is used on all the layers.

8. **Test** : At test time, we use dynamic programming to get the answer span (k, l) where $k \leq l$ such that we obtain a maximum value over their product : $max(p^1_k p^2_l)$

Passage Selection model: For the passage selection module, we have a simple network based on a LSTM and a Multi-Layered Perceptron (MLP) unit to sample one passage/snippet given a query and the associated passages/snippets. The model pipeline is shown in Fig 3.

Given a query and the passages, these are first passed through a pre-trained LSTM network (from the BiDAF network) to get query and passage embeddings respectively. Then, the query embedding is concatenated with each of the passage embeddings and sent through a MLP network to get similarity score between the query and the passage. A passage is sampled based on the weighted probability distribution (after passing the similarity scores through softmax). This is identical to how 'hard' attention works. Note that, the similarity score calculation and the sampling is independent of the order of passages and the number of passages associated with the query. After this, the sampled passage and the query are sent to the BiDAF model to receive a reward from the agent (BiDAF model). The passage sampler network is trained with REINFORCE [Wil92] algorithm with the expected reward as the objective function. The BiDAF model is not trained (pre-trained model is used) and the weights and other learnable parameters are kept frozen.

Training Details: The general details about the combined model are given below and since RL based models are harder to train, we also used distant supervision and baseline to make the training stable and making sure the model is learning properly:

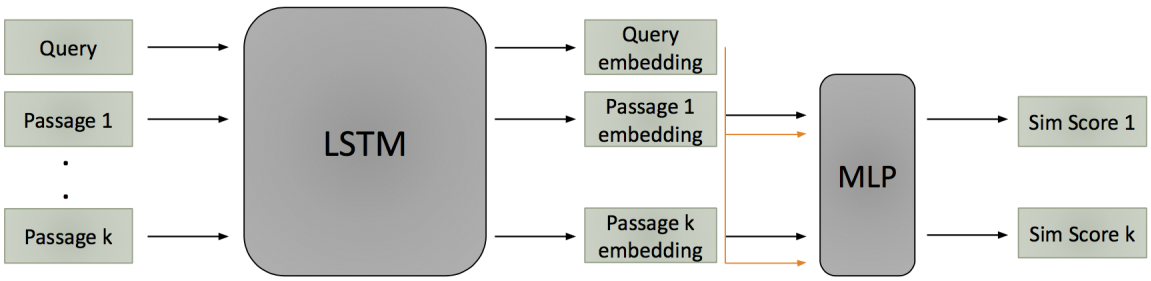

Figure 3: Model Pipeline for RL based Passage Selection Module

1. **Agent, Actions and Reward:** In our case, the environment is the combined model in which the agent samples an action (passage sampler) and receives reward based on the action (from BiDAF model). The action is to sample a snippet/passage given a query and the list of associated passages. For our case, we limit the number of passages to 35 (the average number of passages per query = 50.6) to keep computational complexity low. The reward is $p_1 \times p_2$ where p_1 is the probability of gold start index and p_2 is the probability of gold end index according to the BiDAF model. If the sampled passage does not contain the gold answer, it receives a reward of 0. We chose this as our reward function because we wanted both the start and the end probabilities to be high that's why we didn't use $p_1 + p_2$. Other possibility for the reward function that we could have used: the percentage overlap between the gold and the predicted span.

2. **Objective function for RL task:** To maximize the expected reward i.e.

$$J(\theta) = \sum_{p_k \in \text{passages}} p_\theta(p = p_k|\text{query,passages}) R_\theta(p_k)$$

3. **REINFORCE algorithm:** For training the passage ranking model, we can use REINFORCE [6] through which the gradient of the objective function is approximated with a sample $\hat{p} \sim p_\theta(p|\text{query, passages})$. Although the REINFORCE loss function is not same as crossentropy, we can easily manipulate the target values in order to use the standard cross-entropy loss. Intuitively, $R_\theta(p_k)$ can be thought of as a scaling factor which dictates how the p_k should change in order to maximize the expected future rewards. If action a is good, then, reward will be high and thus, will push p_k for action a by a larger value and vice-versa. Thus, eventually good actions will get higher probability.

4. **Distant Supervision:** To stabilize the training at the beginning of the experiment, we used distant supervision or supervised learning setting to train the model. For the supervised learning setting, the goal was to correctly predict the gold passage given the query. We used a simple heuristic to select the gold passage (since the gold passages are not given): the first passage containing the gold answer. We used the negative log likelihood loss and Adam optimizer with learning rate of 0.001 to train our model for 30 epochs. We started with distant supervision but gradually shifted to RL setting over the epochs. For choosing between SL or RL, we again used a simple heuristic: $\gamma^{\text{epoch-1}}$ where γ or the decay rate = 0.8.

5. **Baseline:** To reduce the variance and to make training stable, we also used baseline. We used the running average of rewards as baseline score. The other reason for use baseline was that in our case, we only had positive rewards and for effective training, we need positive and negative rewards so that our model can figure out how to increase the probability scores of the passages that give positive effective reward and decrease the probability scores otherwise. Here,
   effective reward = reward - baseline
   i.e. actions that perform better on an average end up receiving positive reward

4 Results

BiDAF : We train the model on SQuAD and clearly observe that most of the errors are superficial. Our model accuracy does not exactly match to the original paper as instead of using the moving averages (as done by authors of BiDAF), we directly use the raw weights to evaluate the model. For majority cases, we can observe that, although the predicted and the actual answer are different, both of them can be considered correct. These errors can easily be corrected by either correcting the preprocessing or handling the case of multiple occurrence of answer in the context.

We also test this model on SearchQA dataset. In order to do so, we concatenate the first 10 paragraphs for each question together and generate the true answer span(start and end index) by finding the first occurrence of the answer in the concatenated paragraph. We only obtain an accuracy of 27.8%. On analyzing the errors, we observe that majority of the errors occur because the first occurrence of the answer usually does not correspond to the context of the question. Thus, bolstering our hypothesis for the need of a model which can correctly pick a paragraph capable of answering the given question.

Question	Predicted	Gold Answer	Reason for error	Percentage (Error Category)
Which articles of the Free Movement of Workers Regulation set out the primary provisions on equal treatment of workers?	1 to 7	articles 1 to 7	partially incorrect span	53
What year did BSkyB acquire Sky Italia ?	2014	2014	Wrong span	12
when did French and Indian war ended ?	1754–1763	1763	Splitting on space	3

Table 3: Error analysis on SQuAD

Question	Predicted	Gold Answer	Reason for error
party u singer also plays young lady named hannah	hannah	Miley Cyrus	Wrong answer span (First paragraph talks only about miley, there is no mention of hannah)
ignature appetizer p f changs chicken cups vegetable	crisp lettuce	lettuce	-
4 x 12	4	48	Answer present but context is different.(passage talks about construction)
barbara undershaft	woman	major barbara	Extra information required

Table 4: Error analysis on SearchQA

We believe this is because SearchQA consists of more complex and compositional questions than SQuAD. This can be easily confirmed by looking at the human performance of 80.3% on SQuAD as compared to 47.23% on SearchQA. Also, concatenation of passages drastically increases the context size(from a paragraph of size 700 words in SQuAD to around 2000 words in SearchQA).Since, data in SearchQA is pre-processed, it results in more convoluted language(for example : gandhi deeply influenced count wrote war peace) as compared to formal language in SQuAD. The questions represent keyword-based search queries rather than grammatical questions. We also conjecture that since questions in SearchQA do not contain any wh-words like who , when , where etc. unlike in SQuAD, the BiDAF model is not able to leverage on this and model the questions properly.

Passage selection model: The combined model with the Reinforcement Learning setting did not perform well. The average test reward improved from 0 to 0.3 (plateaued) and there was not a significant improvement in the Exact Match score (in fact the numbers dropped to 6.6). To figure out the reasons why our model was not working that well, we sampled 50 queries and examined the confidence scores of the RL model to figure out the main error categories and to see what improvements could be made to the model. The results are shown in Fig 4. The biggest reason why our current setting is not working that well is that there are too many passages with the gold answer

Table 5: Results on BiDAF

Dataset		Accuracy(Ours)	Accuracy(Original paper)
SQuAD	With character embeddings	64.2	67.7
SQuAD	Without character embeddings	62.3	65.0
SQuAD	Reduced Vocabulary size	63.0	-
SearchQA	With character embeddings	27.8	-

(16.78 per query). We have changed the experimental setting to a multi-label problem now where the goal is to predict any of the answer bearing passages. Other than that, the snippets are very noisy (and they are scrapped from Google and Wikipedia) and some of them don't even make any sense. This hinders the model performance as LSTMs rely on the strong temporal relations between words which sometimes is missing from the snippets in SearchQA. Also, the queries are very short and informal in the sense that they are not typical Wh-queries as seen in SQuAD dataset.

Question	Gold Passage	Predicted Passage (Confidence Score)	Error Type
oil cartel controls 40 world production	oil producing exporting cartels act 2007 even though **opec** controls ``only'' 40 world 's production , influence prices substantial first , opec countries extensive reserves	• … 10 opec nations pump 40 world 's oil supplies … (0.2) • **opec** controls ``only'' 40 world 's production … (0.13) • … opec , cartel controls 40 world 's oil production …(0.25)	Too many passages with the gold answer
name texas city spanish yellow	**amarillo** , texas wikipedia amarillo 14th populous city state texas , united states also city also known yellow rose texas city takes name spanish word yellow , recently rotor city	• …texas city 's spanish yellow answers 8 letters texas city 's spanish … (0.3) • **amarillo**, … yellow rose texas city … (0.25) • amarillo texas cities traveltexas amarillo , means yellow spanish … (0.25)	Passage does not contain the correct answer but exact query
18 became queen 1837 , reigned 63 years	queen **victoria** biography undiscovered scotland last monarch house hanover , ruled 63 years 7 18th birthday , william iv died heart failure victoria became queen idea marriage two , course longer say 1837	• queen elizabeth ii make 63 years 217 days throne (0.1) • queen **victoria** … ruled 63 years 7 18th… (0.14) • queen victoria reigned 63 years , seven months , two days (0.23)	Wording + Noisy snippet

Figure 4: Error Categories and Confidence Scores using Reinforcement Learning

5 Conclusion

In conclusion, we saw that BiDAF works comparatively better on SQuAD dataset (compared to SearchQA) because of the query and passage language style (formal vs informal, noise-free vs noisy). We also noticed that adding character level embeddings increased the performance as they can better handle out-of-vocab (OOV) or rare words. Bidirectional attention helped a lot on doing span selection by capturing the key connections between query and snippets. Also, we think that adding question category along with questions in SearchQA might help for doing conditional span selection because SearchQA answers are shorter (average length = 1.5 vs 3.58 in SQuAD) and most of the times they are entities (person/place/organization). Adding an entitity extractor and conditioning on the answer category would help a lot in improving the scores. The other main reason that our current setting is not working that well is that around 50% queries have 15+ snippets containing the gold answer. So, the heuristic of selecting one passage is not working so well. Instead, if we convert this problem into a multi-label setting where the goal would be to improve any of the gold passage scores and use that for modelling our loss, it could give us better sampled snippets. The bottomline is that the RL setting may not work so well for SearchQA in the current setting without any additional pre-processing (like removing the noisier snippets) and other model adaptations. Finally, we learnt that reinforcement Learning is quite tricky to get working on Question Answering tasks especially on SearchQA.

6 Future Work

For future work, assuming that we get access to computational resources, we would like to try using word embeddings from Language modelling rather than pretrained Glove embeddings. Also, we can use self attention by incorporating multiple hops of attention to allow deeper interaction between context and query. Furthermore, we would like to train the BiDAF model on SearchQA dataset after incorporating the answer category and training embeddings(SearchQA has around 1.2M unique tokens). Also, we would be making adjustments to handle multiple occurrences of the answer in the passage. We can also try to sample multiple passages and see if training improves – but it can harm the performance also. One last thing to try is the techniques presented in the paper [Buc+17] where they use query reformulation to reformulate the query so that the model can return correct answers more accurately and more confidently

References

[Buc+17] Christian Buck et al. "Ask the right questions: Active question reformulation with reinforcement learning". In: *arXiv preprint arXiv:1705.07830* (2017).

[Cho+17] Eunsol Choi et al. "Coarse-to-fine question answering for long documents". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vol. 1. 2017, pp. 209–220.

[Dun+17] Matthew Dunn et al. "Searchqa: A new q&a dataset augmented with context from a search engine". In: *arXiv preprint arXiv:1704.05179* (2017).

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: `http://www.aclweb.org/anthology/D14-1162`.

[Raj+16] Pranav Rajpurkar et al. "Squad: 100,000+ questions for machine comprehension of text". In: *arXiv preprint arXiv:1606.05250* (2016).

[Seo+16] Minjoon Seo et al. "Bidirectional attention flow for machine comprehension". In: *arXiv preprint arXiv:1611.01603* (2016).

[Wil92] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Reinforcement Learning*. Springer, 1992, pp. 5–32.

[WYT17] Tong Wang, Xingdi Yuan, and Adam Trischler. "A Joint Model for Question Answering and Question Generation". In: *arXiv preprint arXiv:1706.01450* (2017).