Question Answering System Based on HOTPOTQA

Jiaming Li, student ID 300233503
CSI 5180 - Topics in AI: Virtual Assistant
Master of Computer Science, University of Ottawa
Final Project Report

1 Project summary

Question answering (QA) is a significant Natural language processing (NLP) problem as well as a long-standing AI milestone. A user can ask a question in natural language and receive a rapid and concise response using QA systems. The capacity to read a piece of literature and then answer questions about it is known as reading comprehension. Reading comprehension is challenging for robots because it necessitates a combination of natural language comprehension and world knowledge. In this project, I modeled HotpotQA dataset as a seq2seq problem. Then, combined context and question to form one single context called context_with_question which is used to feed to the encoder. To demarcate the between context and question, I used ' @q_pad '. The decoder's job is to come up with the answer to this question.

2 Methodology

2.1 HotpotQA Dataset

The HOTPOTQA is a dataset which contains 113k Wikipedia-based question-answer pairs with four key features. A datapoint looks like this:

- Context: A list of different topics. Each topic has multiple sentences explaining each concept.
- Question: A question that asks a question from one particular topic from the context. So the job of the model is to find the answer from all the different topics.
- Supporting facts: this mentions which tell exactly which topic the answer is derived from. But I ignore this information in the model in this project.
- Answer: The answer to the question which is usually a phrase, name, place or thing.

For my project, I employ HotpotQA, a question answering (QA) dataset with difficult, multi-hop questions. There are 112,779 questions in the dataset that require reasoning over many hops to answer. The dataset includes two types of benchmark settings: distractor mode, in which the model must find supporting facts in order to retrieve answers in the presence of noise paragraphs, and fullwiki mode, in which the model must identify supporting facts from the entire Wikipedia text.

2.2 Convolutional Seq2Seq Networks

In a convolution seq2seq model, the most important deviation from RNN based sequence to sequence model is that the input is sent in one go. There is no unrolling in sequence dimension. Here are some important concepts:

2.2.1 Positional Embeddings

Since there is no concept of unrolling, we provide sequence information in the form of positional embeddings. We add these positional embeddings with the input dims.

2.2.2 Residual connections

- Help pass on the input information to each part of the architecture. So each layer has the option to look at previous layers outputs/features in conjunction with the original input.
- Help in gradient flow. Each layer of the architecture gets a good enough magnitude of gradients. In networks with no residual connections, the gradients keep diminishing with each layer resulting in difficult training.

2.3 Project Settings

- **Software Platform**
  PyCharm, macOS, Github
- **Programming Environment**
  Python 3.9.0
  PyTorch 1.10.2
  pytorch-transformers 1.2.0
- **Datasets - HotpotQA**
  The reason to choose this dataset is the length of contexts. In a way, the context can be thought of as a document containing multiple topics and the model's job is to find the answer to the question based on the document. Initially, I only focused on loss and perplexity values and not exact matches with answers while ignoring the 'pad' token in the decoder. In this project, I added an attention mechanism over context+question hidden states. The attention mechanism is later used by the answer decoder. The model is simply learning the train data as train loss is decreasing but validation loss is not.

2.4 Parameters Settings

The parameters settings of two models are shown as Table 1:

Settings	Baseline	With Attention
Batch Size	128	32
Hidden Dim	100	50
Encoder Embedded Dim	100	50
Decoder Embedded Dim	100	50
Encoder Dropout	0.5	0.5
Decoder Dropout	0.5	0.5
Epochs	10	10
Clip	1	1

Table 1: Parameters Settings for the Models

3 Activity table

Activity	Why	Time Taken	Deliverable
Research and study current articles and achievements, build background knowledge	Gather knowledge about QA system models	5 hours	None (Online resources and free video tutorial)
Explore and compare different datasets	Decide which dataset is the most suitable to the project	2 hours	Chose HOTPOTQA
Install the libraries and packages that used in this project and configure the environment	QA system requires a proper environment to run the program	2 hours	The projects can worked well on my laptop
Build and run the model (baseline)	Turn the QA process into a function so that the examples can be easily tried	6 hours	Some result plots should be generated
Build and run the model (with attention)	Improve the model and compare the results with baseline	4 hours	Some result plots should be generated
Test some examples of the model and compare the result both technically and artificially	See how accurate the model could be	3 hours	Answers of questions that I give to the model
Fix and improve the model	Get better results	2 hours	Output could be more accurate
Write report and record the presentation for final project	Summarize what I do for this project	7 hours	Video and paper
Total hours: 31 hours			

Table 2: Activity Table

4 Challenges and Resolutions

4.1 Dataset Changing

Many prominent datasets, such as SQuAD [1], have been created with the goal of providing a dataset to train reading comprehension models. SQuAD has questions that can be answered by analysing a single paragraph, but it is not indicative of the more difficult issues we can encounter, which require extra reasoning.

To get around these restrictions, HotpotQA creates questions that involve reasoning across numerous sources in natural language, rather than relying on a pre-existing knowledge base [2]. It also provides strong supervision in the form of supporting facts so that the system can grasp how the result was arrived at, which will aid systems in performing meaningful, explainable reasoning.

HotpotQA differentiates out from other QA datasets because it delivers a more realistic and tough dataset that more closely resembles how real-world QA systems should be constructed. The distractor and fullwiki settings are available for the HotpotQA dataset. The authors utilise bigram TF-IDF to extract eight paragraphs as "distractors" given the question query to test the model's ability to detect true supporting facts in the presence of noise. The system is then fed these eight paragraphs, along with the two gold paragraphs, to verify its robustness. The first paragraph of all Wikipedia articles is given into the system in the fullwiki option. This "tests the system's ability at multi-hop reasoning in the wild," according to the scientists [2].

4.2 Researches on HotpotQA are limited

The number of papers on HotpotQA is very limited, I did some research on building a model based on HotpotQA dataset.

In Simple and Effective Multi-Paragraph Reading Comprehension, Clark and Gardner suggest a model [3]. The model employs a CNN-based combination of GloVe-based word embeddings and character embeddings.

Attention Is All You Need is another major study that impacted architectures for typical NLP activities, including QA [4]. The authors of this work propose that "transformers," an attention-based unit, be used to replace standard recurrent architecture. The authors replace the typical RNN architecture with a completely attention-based architecture, which enhances model performance on a variety of tasks, including SQuAD, and aids parallelization during training.

5 Demo, Results and Outputs

5.1 Demo

Figure 1 shows what I tested for the model. I input a text file which contains the introduction of the University of Ottawa, it describes uOttawa's location, history, campus floor area, campus size, student ratio and so on. Then, I asked couples questions like "Where is University of Ottawa located in?", "What is the history of University of Ottawa?". Then, the model can answer the questions as "Ottawa, Ontario" and "the university of Ottawa was first established as the college of bytown in 1848" according to my input text. Even I checked the answers, they are both correct.

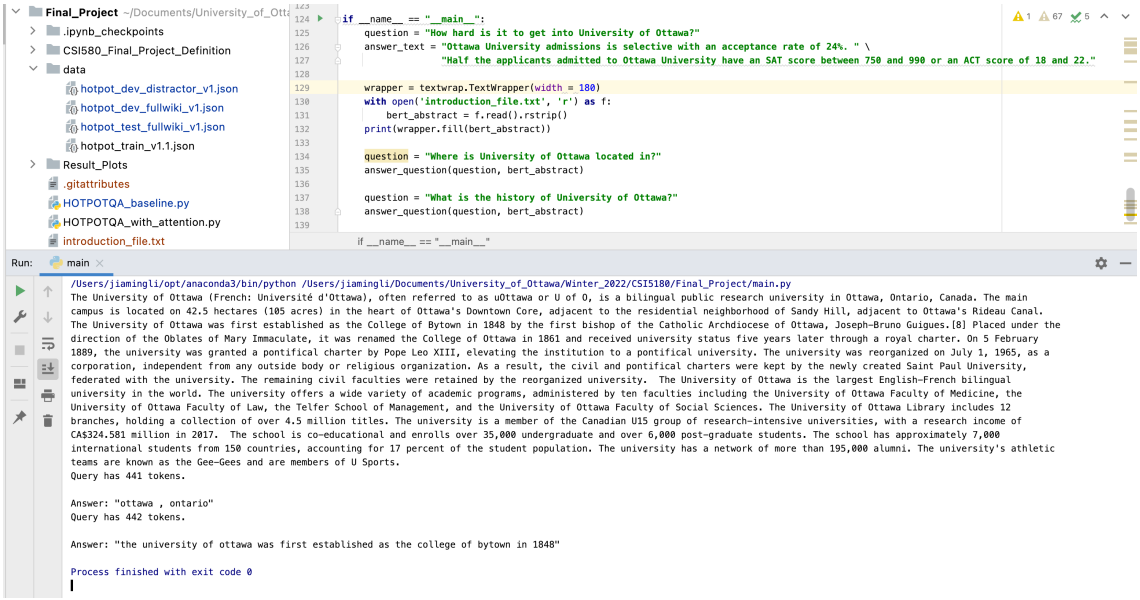

Figure 1: Demo uOttawa

5.2 Results

The following tables show the results of loss and perplexity values. From Table 3, we can see that both train loss and valid loss (no matter it is with attention or not) decrease. Perplexity is a metric used to judge how good a language model is and a lower perplexity score indicates better generalization performance. So as the results shown, the train and valid perplexity values have the same trends as loss value – the overall trend is down, which indicates that a better performance the model is training to get.

Epochs	Baseline Train Loss	Valid Loss	with Attention Train Loss	Valid Loss
1	5.834	5.733	5.737	5.732
2	5.130	5.653	5.142	5.632
3	4.836	5.541	4.841	5.533
4	4.560	5.435	4.565	5.461
5	4.338	5.418	4.359	5.489
6	4.145	5.432	4.195	5.521
7	3.963	5.453	4.056	5.507
8	3.795	5.493	3.926	5.542
9	3.595	5.582	3.816	5.589
10	3.401	5.687	3.693	5.597

Table 3: Comparison of Train and Valid Loss of Two Models

Epochs	Baseline Train PPL	Valid PPL	with Attention Train PPL	Valid PPL
1	341.765	308.765	310.068	308.443
2	168.942	285.170	171.088	279.133
3	125.912	254.845	126.599	252.864
4	95.623	229.224	96.104	235.350
5	76.550	225.499	78.191	242.006
6	63.122	228.493	66.321	249.995
7	52.631	233.348	57.764	246.381
8	44.495	242.908	50.704	255.139
9	36.429	265.648	45.424	267.489
10	29.986	295.011	40.158	269.701

Table 4: Comparison of Train and Valid Perplexity Values of Two Models

5.3 Outputs

The plots of train and valid loss with/without attention, train and valid perplexity values with/without attention are displayed below:

From Figure 2 and Figure 3 shows above, we can see that both train loss decrease directly without increasing, even a little bit. And the whole trend of train loss is obviously doing down. However, for both of the valid loss, they decrease first, then when epoch = 5, they start to going up, and the loss values become greater and then keep increasing until the training is over.

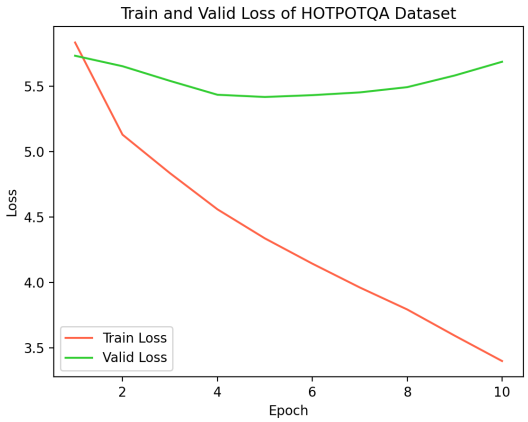

Figure 2: Train and Valid Loss

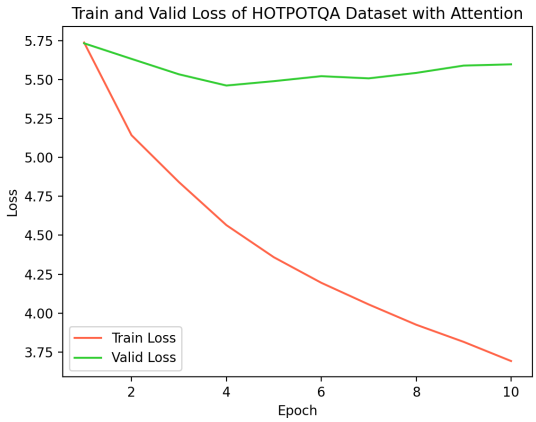

Figure 3: Train and Valid Loss with Attention

When it comes to train and valid perplexity values, we all know that perplexity measures how well the model predicts the test set data; in other words, how accurately it anticipates what people will say next. The results indicate most of the variance in the metrics can be explained by the test perplexity. From Figure 4 and Figure 5, we can see that train perplexity values show a downward trend, but the valid perplexity value declines at the beginning. Combined with the data content in 4.2 Tables, they start to rise at the when epoch = 6 with or without attention, but the final perplexity value is still lower than the value at the initial training.

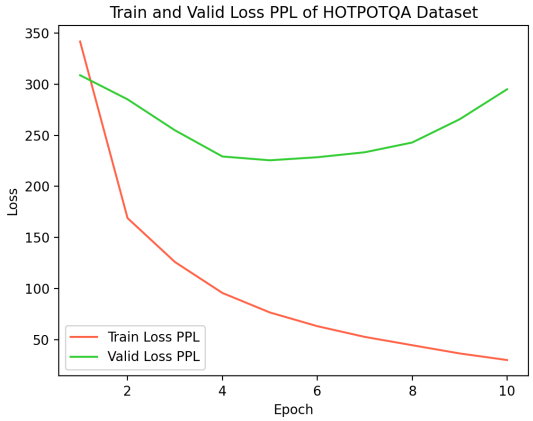

Figure 4: Train and Valid Perplexity Values

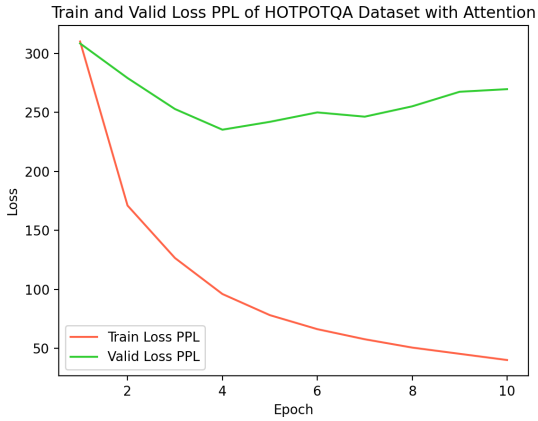

Figure 5: Train and Valid Perplexity Values with Attention

6 Conclusion

In this project, I identified some issues with the baseline model based on the HotpotQA dataset, and then proposed and implemented some learning and architectural changes to that model. With attention, loss value and perplexity value are higher than the values of baseline, which does not change efficiently.

In addition, we know that hyperparameter tuning is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model argument whose value is set before the learning process begins. The key to machine learning algorithms is hyperparameter tuning. Although I explored a variety of improvements to the baseline architecture, I did not explore additional hyperparameter tuning beyond learning and optimization, and I believe that there is still room to do more in this area.

7 GitHub Code

https://github.com/JaneLi99/CSI5180_Final_Project

References

[1] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2016, pp. 2383–2392. [Online]. Available: http://aclweb.org/anthology/D16-1264

[2] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, "Hotpotqa: A dataset for diverse, explainable multi-hop question answering," 2018.

[3] Clark, C., Gardner, M. (2017). Simple and effective multi-paragraph reading comprehension. arXiv preprint arXiv:1710.10723.

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[5] Rajpurkar, P., Jia, R., Liang, P. (2018). Know what you don't know: Unanswerable questions for SQuAD. arXiv preprint arXiv:1806.03822.

[6] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2019). Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771.

[7] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2020, October). Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations (pp. 38-45).

[8] Su, D., Xu, Y., Winata, G. I., Xu, P., Kim, H., Liu, Z., Fung, P. (2019, November). Generalizing question answering system with pre-trained language model fine-tuning. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering (pp. 203-211).