CS 512: Advanced Machine Learning-

Long Form Question Answering Using Seq2Seq Models

Guide: Sathya N. Ravi Student: Sajal Chandra UIN: 659145709

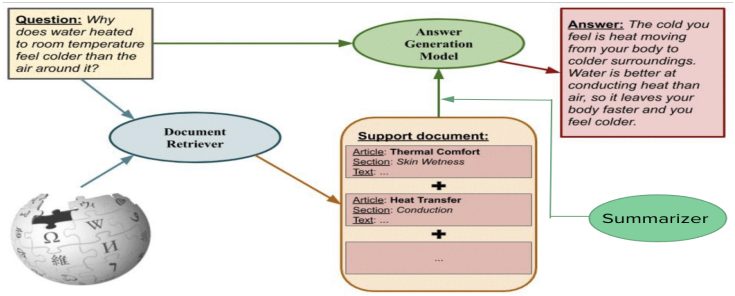
This is the final report of my project titled *Long Form Question Answering using Sequence to Sequence Models*.

1 Inspiration

The paper titled "ELI5: Long Form Question Answering"[1] was the major source of inspiration for this project. The authors introduce a large corpus(collected from Reddit) which comprises of 270,000 question-answer sets which are ideal for training a model to perform a long-form question answering task. Additionally, they introduce a procedure to improve the accuracy of the answers provided by the model- using a supporting document to answer each question.

2 Pipeline

The ELI5 authors propose a simple approach to answering the given question. As illustrated in the diagram, a document retriever is used to retrieve Wikipedia articles that potentially contain useful information pertaining to our question. The support document is then passed to the Answer Generation Model along with the original question pair.



Since Wikipedia articles are detailed and contain 3500 words on average, creating support documents using entire articles leads to slow performance. Further, articles that are not relevant to the question being asked can even have a negative impact on the prediction. To mitigate this problem, I introduced a text summarizer into this pipeline with the aim of reducing the complexity of the model and boost its performance.

3 Seq2Seq Text Summarizer

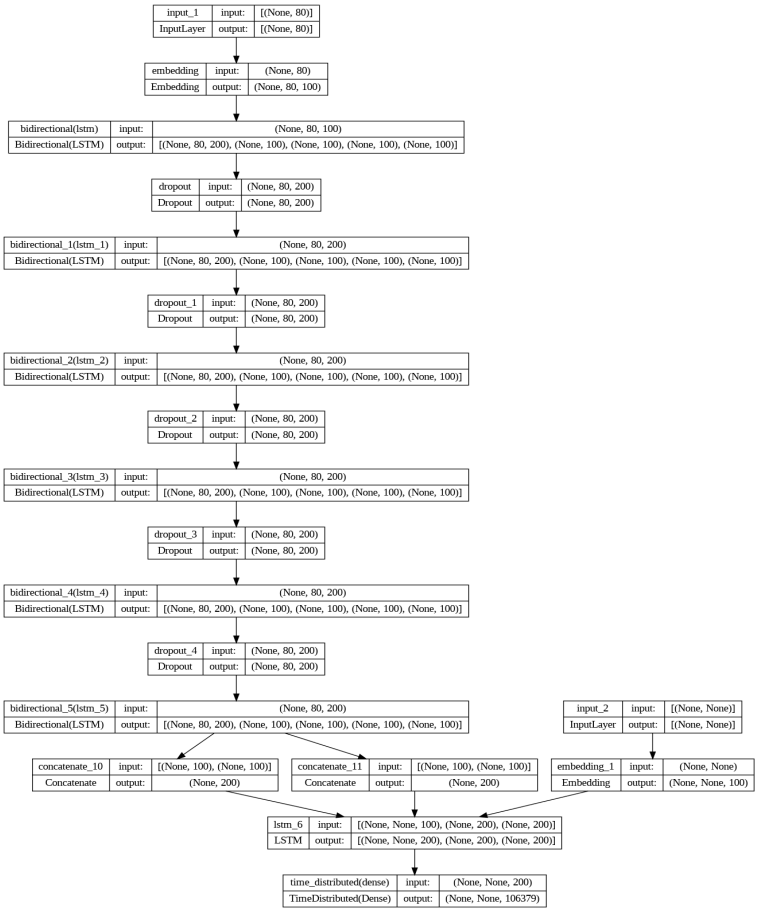
3.1 Dataset

To train a sequence-to-sequence model for summarizing Wikipedia articles, I used the wikipedia-summary dataset[2] from Hugging Face. It contains over 3M article-summary pairs. The dataset is in English, and contains extracted summaries and articles- which makes it easier to train for out specific task.

3.2 Preprocessing

The preprocessing steps are similar to the other text summarizartion tasks in this report. I performed the following operations in the raw text- removed HTML tags, converted to lowercase, removed stopwords, and eliminated punctuations. After cleaning the text, I used nltk's tokenizer to vectorize the text.

3.3 Summarization Model Architecture



Starting with a simple LSTM model, I trained and fine-tuned it b testing various changes such as adding hidden layers, changing the learning rate, adding dropout layers, etc. Eventually I landed on the model shown next. It is a bidirectional LSTM model with a total of 16 vertical layers. This is a sequence-to-sequence model that takes input sequences of length 80 and maps them to dense vectors of length 100 using an embedding layer. The model then processes the embeddings using six bidirectional LSTM layers to capture past and future contexts of each word. Dropout layers are included for regularization, and the target sentences are concatenated with context vectors before being fed to the decoder LSTM. The output layer applies a softmax activation to produce a variable-length output sequence.

4 Evaluation

After training this model on the wikipedia-summary data, the model achieved a BLEU score of 0.6338. Plugging this model into the pipeline and evaluating the Answer Generation Model's performance, I got the following results:

	rouge1	rouge2	rougeL	rougeLsum
Precision	0.2781	0.0505	0.1691	0.196628
Recall	0.2607	0.0399	0.1665	0.182340
F-Score	0.2403	0.0414	0.1453	0.165588

Figure 1: Full Support Document

	rouge1	rouge2	rougeL	rougeLsum
Precision	0.2721	0.0391	0.1600	0.193046
Recall	0.2371	0.0295	0.1493	0.163854
F-Score	0.2191	0.0307	0.1310	0.150292

Figure 2: Summarized Support Document

Performance with No Support Document

	rouge1	rouge2	rougeL	rougeLsum
Precision	0.2213	0.0254	0.1313	0.158176
Recall	0.2147	0.0259	0.1386	0.153314
F-Score	0.1893	0.0210	0.1169	0.134525

Figure 3: No Support Document

5 Conclusion

Upon observing the results, it is clear that summarizing the support documents did not lead to a significant loss of information- as the ROUGE scores are only slightly lower than the original model's performance. Further, when compared with the model's performance without any support document, it is confirmed that summarizing the support document indeed helped the Answer Generation Model come up with better answers.

References

[1] Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. Eli5: Long form question answering, 2019.

[2] Thijs Scheepers. Improving the compositionality of word embeddings. Master's thesis, Universiteit van Amsterdam, Science Park 904, Amsterdam, Netherlands, 11 2017.