Title Generation Based on LSTM With Attention Model

Samuel Steiner

Background

Data summarization is a topic which peaked my interest in natural language processing I had originally looked at a project which would create meaningful captions for images but decided to avoid projects which had any element that dealt with non-text data. I changed my project to headline or title generation for a body of text. This form of text summarization is important. We expect headlines or titles of papers to give us enough information to make a decision on whether or not we want to read the article or paper. They are the first piece of information we get about something we choose to read. Through searching I found many different models that generate titles but many of them were not doing any summarization, rather generating titles based on a list of titles[2, 3 ,4]. I was able to find a paper which described a LSTM model which use attention to generate titles based off of text from an article. I decided to try and recreate the findings of this paper by Konstantin Lopyrev using white papers[1].

Dataset

The reason I chose white papers as a focus was due to their ease of access through services like arxiv. As well as titles of these papers should somewhat be a summarization the paper in order to entice readers to choose and read the paper. This project focuses on taking the abstract from a white paper, and generating a title. This does assume that the titles of the papers that were selected are summaries of the paper, in order to get better results a check of all data should be done but for time the assumption was made that this was true. This also assumes that the abstracts are good summaries of the article. This is another assumption which should be audited for a better dateset. Articles that were submitted to arxiv were chosen from the computer science (CS) category and were selected based on their last

updated date, dates between January 1st 2020 and October 31st 2020 were chosen. This yielded a dateset of about 63,000 papers. Each abstract and Title went through a pre-processing phase to remove any unwanted data this was done by removing any punctuation marks and removing stop words from the abstracts. Then each Title and Abstract was tokenized down to a world level.

The Model

Once the data has been processed we can move on to the model. The model closely followed the architecture which is described in the paper. That being an encoder-decoder architecture which are both recurrent neural networks. The encoder takes the text of an abstract one token at a time. Each token is first passed through a word2vec embedding layer that transforms the word into a vector representation. This is then fed into four hidden layers which as well with the previous word or all 0s the first word is combined.

The decoder takes the hidden layers generated after feeding in the last word of the input text as input. Using an embedding layer again it transform the symbol into a word2vec. Then, the decoder generates each of the words of the headline ending with an end-of-sequence symbol, using a softmax layer and the attention mechanism. After generating each word that same word is fed in as input when generating the next word. The final title is then fed through a beam search to reduce repeated words. Throughout training we calculate the loss based on the log loss function [1]:

$$-\log p\left(y_1,\ldots,y_{(T')}|x_1,\ldots,x_T\right)=\sum_{t=1}^{(T')}\log p\left(y_t|y_1,\ldots,y_{(t-1)},x_1,\ldots,x_T\right)$$

As in the paper we use a learning rate of 0.01 along with the RMSProp gradient ,for the RMSProp we use a decay of 0.09. Diverging from the paper instead of training for 9 epochs we train for 7 this was due to a technical error and due to time constraints the model could not be trained again. Due to limited hardware I had to use batch sizes of 32 rather than 384.

The architecture of the attention model is shown in the original paper with the following figure.

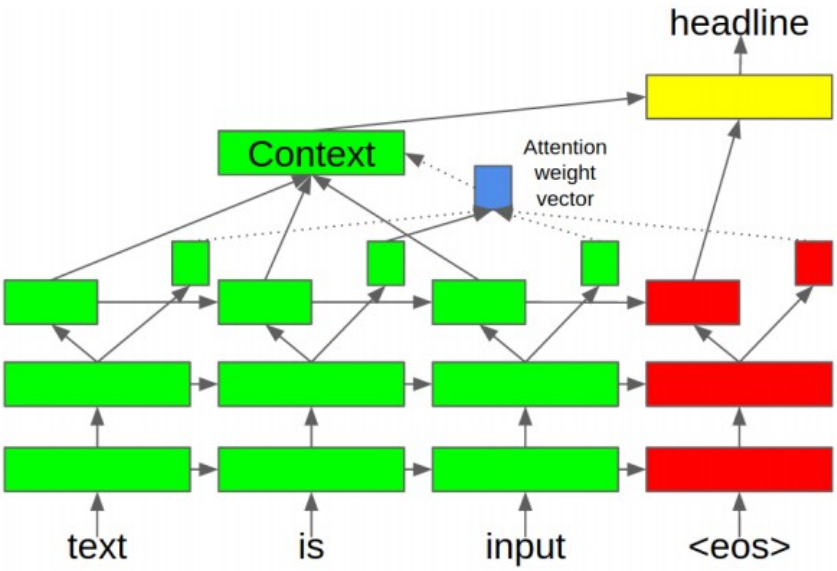

Figure 1: Simple Attention Architecture

Evaluation & Analysis

The model will be evaluated based on the BLUE score over the holdout set as well as the loss just as in the paper. These evaluations will be used to compare the results of this project to the original paper.

While the Blue score did increase over each epoch, and the loss decreased over each epoch we do not see the changes observed in the original paper, even over the first 7 epochs in the paper in the BLUE score and the loss is less comparable. Each of the graphs from this project will be shown but only the BLUE score from the original paper will be shown.

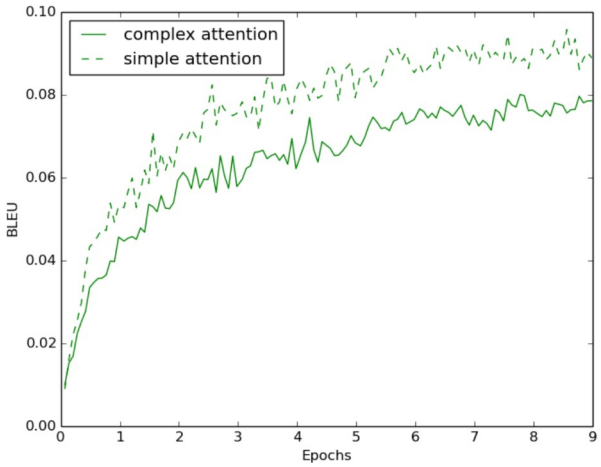

Figure 3: The original paper's BLUE vs epoch

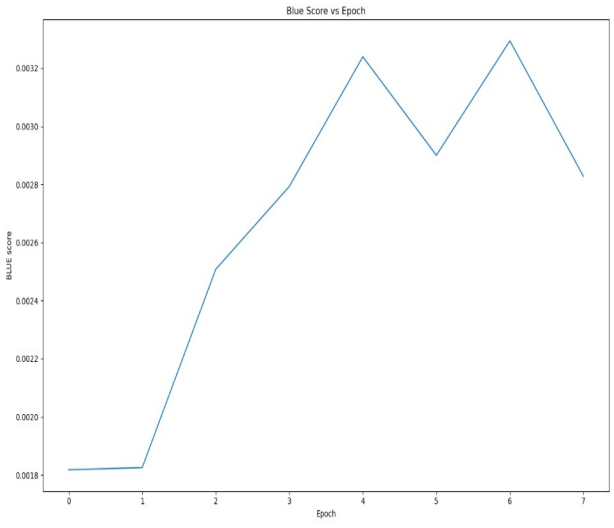

Figure 2: This model's BLUE vs epoch

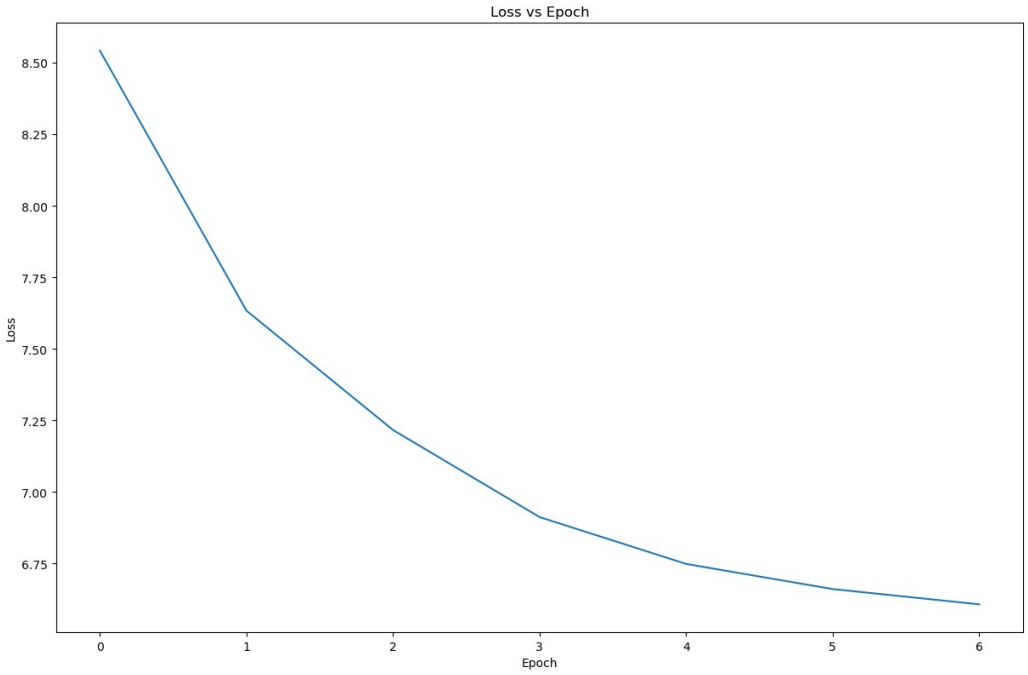

Figure 4: Loss Vs Epoch

After the model was trained it was tested with trying to predict several thousand abstracts to see if and comparing the actual title with the predicted title. The following table shows some of the titles generated and the original title. These have been handpicked as some of the best.

Predicted Title	Actual Title
development with software engineering projects	a practical guide towards agile testdriven development for scientific software projects
networks with multiple information constraints	design of periodic scheduling and control for networked systems under random data loss
smart health detection using deep learning	recent advances in wearable sensors with application in rehabilitation motion analysis
twitter using social media on the twitter	facebook political ads and accountability outside groups are most negative especially when

	microtargeting or hiding donors
networks using online learning with advice	multigraph convolutional network for relationshipdriven stock movement prediction
networks with deep reinforcement learning	multiple access in aerial networks from orthogonal and nonorthogonal to ratesplitting
adversarial with graph data for domain adaptation	deep adversarial domain adaptation based on multilayer joint kernelized distance
natural language comprehension using deep learning	protoqa a question answering dataset for prototypical commonsense reasoning
airborne vehicles using deep reinforcement learning	understanding realistic attacks on airborne collision avoidance systems

From the examples we can see that model was not able to accurately predict titles for these abstracts. There are several reasons which I want to explore for why this may have happened an offer future investigations can be used to remedy these problems.

Problems

The two factors which biggest problems and the ones ones which I will offer solutions too are that of the word2vec embedding and the dateset size. These two things I would argue would create the most problems when training the data. Obviously there is a huge difference in the dateset size from the original paper and this project. They used a news dateset which included about 5.5 million news articles across several years and several different news publications. My dateset was much much smaller I think including papers from multiple different categories and over multiple years could of made the dateset large enough. This is an easy fix but takes time to collect the data.

The other problem which could be explored is the word2vec embedding. This embedding it self is a model which can be trained. The word2vec used in this project was trained on Google News articles. I suspect that there is a lot of optimization which can be done here, I would propose a project which we train the model several times with different word2vec

models. While many models exist based off Wikipedia articles or Google News, there are no pretrained word2vec models based on academic papers or more specifically arxiv. I would say that the model would work much better if the embedding was based around white papers.

Conclusion

While I was unable to reproduce the results of the original paper I think this project gave me a lot of insight into recurrent neural nets in terms of text generation and text summarization. This project allowed me to explore the topic in a way highlighted what parts of the process are important. The results of my project have also shown me many ways which I can further research this topic which future projects could be based on. It would be worthwhile to explore the ideas presented in the problems section to try and get results which are closer if not reproduce the results of the original paper.

Citation and References

https://arxiv.org/abs/1512.01712

https://github.com/sallamander/headline-generation

https://thecleverprogrammer.com/2020/10/05/title-generator-with-machine-learning/

https://towardsdatascience.com/nlg-for-fun-automated-headlines-generator-6d0459f9588f

https://medium.com/@ishita19013/title-generation-using-nlp-440fa1156e97