DOCUMENT SUMMARIZATION

By

Ratnadeep Mitra
Sooeun Oh

Submitted to
Dr. Keegan Hines
in partial fulfillment of the
requirements for the degree

of
Master of Science

Georgetown University
Neural Networks and Deep Learning (ANLY-590-02)

December 14th, 2018

Abstract

Sequence to sequence model has been developed for translating a language into a different language. The goal of this research to apply Seq2Seq network to develop a text summarization model that summarizes news articles into headlines. Following the goal, we developed a deep learning model for abstractive text summarization using Gated Recurrent Units and trained it on US news articles.

1. Introduction

Text summarization is the process of generating a short, fluent, and most importantly accurate summary of a respectively longer text document. The goal of automatic text summarization is to present the source text into a shorter version with semantics. The most important advantage of using a summary is that it reduces the reading time. Text Summarization methods can be classified into extractive and abstractive summarization. An extractive summarization method consists of selecting key sentences, paragraphs, or passages in the original document and concatenating them, and reproducing them into a shorter form of summary. An Abstractive summarization is an understanding of the main concepts in a document and generating a short summary consisting of a few sentences that represent the main idea of the source document. The abstractive method is not merely a simple selection of a few sentences extracted from the source; rather, it is a compressed summary that delivers the main contents of the document, possibly using the vocabulary unseen in the source document.

2. Related Work

In approaching the problem, we considered the findings of several papers that provided a good understanding of the research already done. Text summarization techniques have been widely explored over the past decades such as corpus-based approaches, cohesion-based approaches and graph-based approaches. However, recent studies indicated that the encoder-decoder recurrent neural network architecture developed for machine translation has been proven to be effective when applied to the problem of text summarization. Sequence to sequence (Seq2Seq) learning has been used for abstractive and extractive summarization. In more recent studies, researchers proposed a novel document-context based Seq2Seq models using Recurrent Neural Networks (RNN) for summarizations. Most studies used BLEU and ROUGE scores to optimize evaluation metrics.

3. Datasets

We evaluated our model on Kaggle's All the news dataset, which contains 143,000 news articles from 15 American publications. The dataset contained the following fields - *id*, *title*, *publication*,

author, *date*, *year*, *month*, *url* and *content*. Out of all the fields, only *title* and *content* were kept and the rest were discarded. Due to a limited computation power, we had to work with a subset of random sample of 20,000 articles from the original dataset.

4. Methods

4.1 Data Pre-Processing

Data was pre-processed using the *ktext* utility package. The *ktext* package performs common pre-processing steps associated with deep learning, including steps of cleaning, tokenization, padding and truncation using process-based threading in parallel. Cleaning involved removal or replacement of unnecessary characters with generic tags and lower-casing the text in order to reduce the noise in our data. Then, the document was split into a list of words during tokenization, so that the mapping generated unique tokens and assigned integer value to each of the tokens. After data cleaning and tokenization, pre-padding and post-truncation was done such that all documents were of the same length and each document length is the heuristic of the 70th percentile of the document length. Only the top 8,000 words in the vocabulary were retained and the remaining words were set to the index 1 which corresponded to rare words. '*_start_*' and '*_end_*' indicators were added to the beginning and end so that each document can be identified.

4.2 Model Architecture

Sequence to sequence network architecture was implemented for the analysis. The diagram below shows the model architecture.

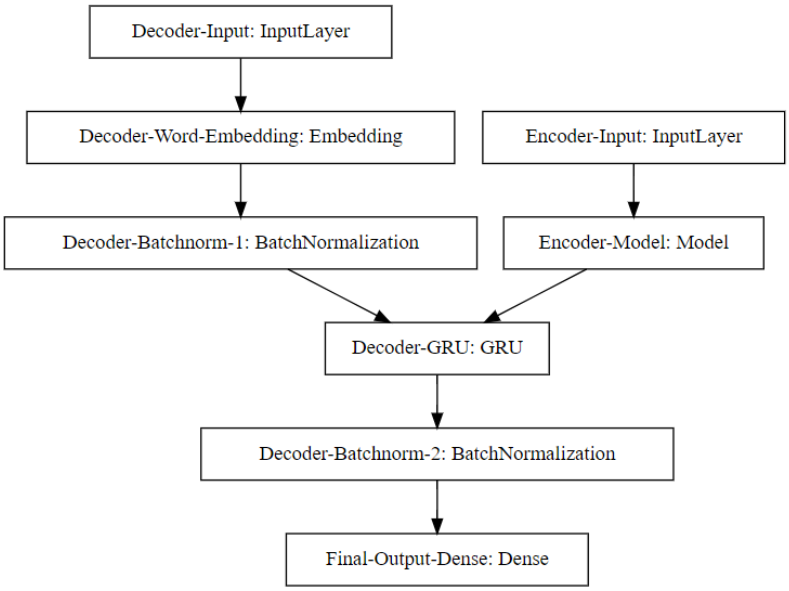

Figure 1: Encoder-Decoder Model Architecture [1]

The model put together encoder and decoder with Gated Recurrent Units. Teacher forcing was used as well to allow the model to train faster. The loss function used was sparse categorical cross-entropy because it allows integers as targets. It is more memory efficient before the targets need not be one-hot-encoded.

5. Results

Due to computation limitations, we were able to train the dataset for only 5 epochs. The graph below (Figure 2) shows that the loss in the validation dataset did not reduce significantly. This observation indicated that the model was possibly not trained enough.

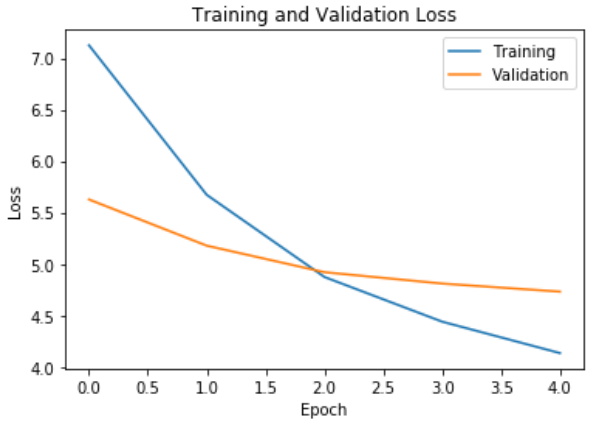

Figure 2: Track of Training and Validation Loss

Not surprisingly, several predictions were not accurate, and a few phrases were repeated. However, a few predictions were accurate as well. For example, when the original title and generated title were compared in the test set, it was found that the model summarized a news article as "*America by air calm before the storm*", whose original headline was "*America by Air: Descending into a Dust Storm*". Another instance of a decent prediction was "*donald trump your wednesday evening briefing the new york times*" and the original headline was "*Donald Trump, John Glenn, Oakland: Your Thursday Evening Briefing - The New York Times*". On the other end of the spectrum, the model generated predictions which were in no way related to the news article. For example, the model summarized a news article as "*The pope francis allow married man to become priests*", whose original headline was "*How Climate Change Unleashed Humans Upon South America's Megabeasts*".

6. Discussion of Results

As stated earlier, the results were not consistently accurate. This could be due to several reasons. One of the major issues we faced was the lack of computation power. Training such a large dataset on 8GB CPU was not feasible. We were finally able to run a subset of 20k rows on a 16GB CPU but with only 5 epochs. Typically, training a model for document summarization

requires a large dataset and/or a high number of epochs. Furthermore, our model was not complex enough. A stacked RNNs or an attention-based decode would have yielded better results. Another issue was the interpretation of special characters in the text. The program was not able to read the special characters with either '*Latin-1*' or '*UTF-8*' encoding, which have negatively affected the accuracy of the predictions.

7. Conclusions

Our model was able to perform well on some articles, but it also gave average and some below average performance. We believe that, with our limited computing resources, we have not been able to explore the full potential of the Seq2Seq model. Attention mechanism would have been helpful for summarization. To pay attention to a part of the source sequence, is to assign a high attention weight relative to the rest of the sequence. It means that the dynamic context built by the decoder contains more information from the encoder states corresponding to the parts of the source sequence that are interesting. In other words, attention mechanism allows the network to refer to the input sequence, instead of forcing it to encode all information into one fixed-length vector. Seq2Seq is very well-suited for document summarization, and we believe that a better trained and more complex model would have given us more accurate predictions.

References

[1] Husain, Hamel. "How To Create Data Products That Are Magical Using Sequence-to-Sequence Models." *How To Create Data Products That Are Magical Using Sequence-to-Sequence Models*, Towards Data Science, 18 Jan, 2018.