

Abstractive Text Summarization

Yalamaddi Abhinav*

Student, School of Computer Science and Engineering, Vellore Institute of Technology, Amaravati, India

Abstract: The objective of this project is to imitate the fundamental concepts of this research of state-of-the-art abstract text repeated models to examine various processes until the work had a reasonable functional basis. This work was motivated by various research papers with several novel features that have achieved remarkable achievement. In multiple iterations, this study will enhance the adoption of words, complexity of decoders, and attentiveness. In addition, a bilinear care mechanism adds the last model, increasing the rate of loss of training. Text Summarization is one of the most experimental subjects in natural language processing that reduces the size of a document while keeping its meaning. Summary techniques are classed as extractive or abstractive based on whether the precise phrases in the original text are produced or whether new phrases are constructed using natural language methods. Extractive summaries have been carefully examined and a developed state has been obtained. Abstractive summary is the focus of the research. Due to the ins and outs of the text, abstractive summarization is challenging.

Keywords: Text summarization, Natural Language Processing, Abstractive.

1. Introduction

In recent years the summarization of the text has been noticed by the web content overflow. This information abundance enhances the need for dynamic and able text summaries. It is of immense importance, because there are many different applications including journals, books, magazines, stories about this same topic, events, scientific articles, weather forecasting, stock exchange, news, novels, music, plays, movies and speech. Thanks to its huge expansion, numerous world-class universities including Aarhus University-Denmark, NaCTeM-Manchester University and others worked hard to develop it.

As the amount of information and data available on the Internet grows every day, it is becoming constantly an open research issue to acquire and understand the information needed as soon as feasible. The collection of all the information and subsequently the production in summary form is a tiresome operation. The Internet is a database information platform. But this data still has to be processed massively. Thus, a text-based synthesis was required, which conserves the meaning and content of the document into a shorter version. A brief is therefore helpful in saving time and retrieving huge data from papers.

In general, the text resume can be classified into two categories: extractive summary and abstract summary.

Extractive type selects significant sentences or phrases from source materials and organize them into a synopsis without modifying the original text. The phrases are generally followed as in the original text. However, abstraction of the original text is studied and examined using the approach of linguistic use. A comprehensive summary aims to provide information accurately that usually calls for advanced language production and strategies of compression.

Abstractive summary is an effective kind of summary in comparison with extractive summary because it collects information from several texts in order to construct an accurate summary. This has becoming more popular since new sentences may be developed to convey vital information from text documents. In a consistent and linguistically accurate form, an abstractive summarizer delivers summarized information. Readability or language quality is a vital driver to improve a summary's quality.

2. Background and Literature Survey

The summarization is about providing a brief summary that highlights the major concepts of the text of the input. For this project, the work was focused on abstract composition that creates a paraphrase summary. The extractive summary works by extracting only the input terms by comparison.

Abstraction models fall within the more profound learning category known as models that map from input sequence to target sequence. While sequences of sequences have been applied effectively for several NLP difficulties, including machine translation, the abstract summary of cutting-edge models still offers lots of chance for improvement. While state of the art models can achieve high ROUGE scores in tiny input summaries, the model often loses its capacity to sum up key points once large inputs are assessed. The abstract model's standard metric. While hierarchical models have managed to summarize large inputs, they are still a long way away. The task of summarizing papers is thus an unresolved difficulty for natural languages to be processed.

Despite the clear distinction between abstract summary and machine translation, an international standard of abstractive summary has become the attention-grabbing model of the RNN encoder-decoder as suggested [2]. The encoder generates an input representation for both activities, and the decoder generates the final output using these encoding. [6] This paper proposes, as a base model, a two-directional encoder, a one-

*Corresponding author: abhi.yalamaddi@gmail.com

directional decoder, an attention mechanism for the cached state of the source and a layer from SoftMax over the vocabulary. Then combines several elements to focus attention, including a huge quantity of vocabulary, feature-rich keyword coding and the use of pointer networks for modelling unusual words. [5] The vocabulary trick restricted the decoder's vocabulary to terms in batch input files, increasing convergence and addressing the bottlenecks of SoftMax computing throughout the entire vocabulary. It adds words from the objective dictionary to the vocabulary of the decoder until a default size is reached. This approach concentrates the model on the source words, which are helpful in summary.

3. Proposed System

A. Preprocessing and Dataset

Within this particular project, we trained the model on Kaggle's Amazon Fine Food Reviews dataset, containing 500,000 reviews, generally used for training models for abstractive summarization from 1999 through 2012. We have drawn up 80% data and summaries during preprocessing and trained the model to predict the abstract summary following the user review.

Ref: <https://www.kaggle.com/snap/amazon-fine-food-reviews/download>

B. Model

A Three Layer Stacked LSTM Encoder-Decoder model with Global Attention Mechanism was used during implementation. On the training set, the algorithm was able to achieve an accuracy of 77.27 percent using this model (constituting 80 percent of the dataset). This model also achieved a cumulative BLEU-4 score of 0.8800 on the test set.

C. Working Methodology

As stated before, this project is implemented using 3 models as a stacked layer.

1. The first model was a basic one-way LSTM encoder decoder with randomly initialized words. As mentioned later, discovered that using preexisting word embedding would be better, and hence iterated Model 2 using this model.
2. The model has been built with a two-way LSTM encoder and one-way LSTM decoder.
3. Implementation of a two-way LSTM encoder in the last model and incorporated the LSTM decoder worldwide attention. Previously provided attention scoring functions. The final secret state (forward and reverse) of the encoder is linked and utilized as the initial hidden state of the decoder.

Model 1 randomly deploys and upgrades the Word Embedding model. Here it is assumed the words vector representations that were learned would be more easily customizable than Word2Vec or Glove vectors that were previously trained. But it soon became known that limits on machine and time would prevent the attainment of such an objective, as many iterations on huge data sets are necessary to develop accurate word embedding. Glove vectors are loaded for

the model 2 and model 3 embedding matrices because it is not possible to train in words on a huge dataset.

Used the same integration and terminology to facilitate encoding and decoding. In the context in which a word appears, however, different patterns such as Rush et al. (2015) provide a variety of embedding arrays for diverse functionalities, thereby increasing the concept of vector semantics.

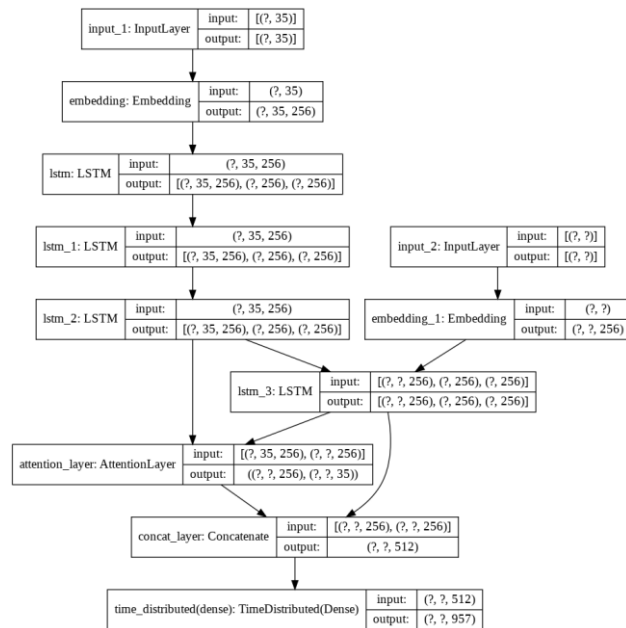


Fig. 1. Flow structure of the algorithm implemented

D. Analysis

At training time, each step of the model decoders incorporates article truth as an input. This enables the machine to learn predictions given inputs x and past predictions, y . However, during the test, the decoder sends its result into the next decoder cell as a word embedding input. For the gradient descent technique, implemented using RNN algorithm. The reason to pick RNN because it processes sparse data well and makes greater updates for small frequency parameters and smaller updates for large frequency parameters. This is extremely beneficial in handling larger vocabulary sizes, as common words appear far more frequently than unusual terms.

The need to start exploring new attention functions, after creating and successfully testing basic model models. To boost the model's capacity to recognize critical data, allowing the encoder to offer scaled 'size values' in over-sized states. You must either update your attention scores directly or you must connect them to your covert vector encoder.

Therefore, ROUGE had to be used to examine the results, but time to construct a competitive model was not available. Unfortunately. However, a totally different strategy was tested in the last effort. In 3000 vocabulary data points and 600 epochs there were 1,000 instead of 100,000 training data points with 5,000 vocabulary-size 10 epochs during implementation. At the period of 80 we saw the summary predicted, the first word of which was based on essential truth, but the article or title was not immediately replicated by the first word.

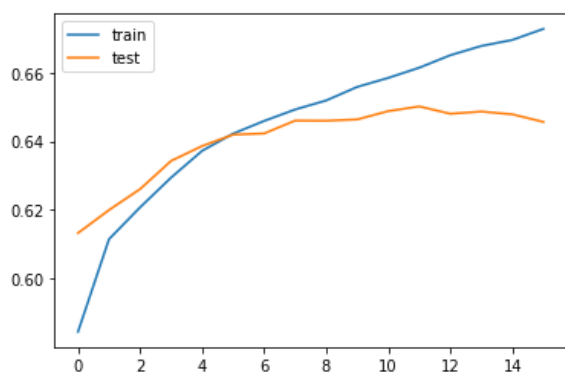


Fig. 2. Accuracy graph

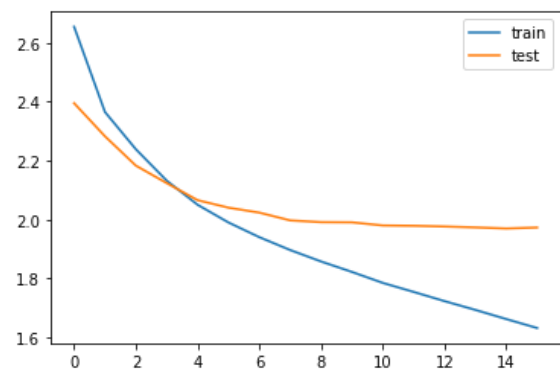


Fig. 3. Loss graph

4. Results and Discussions

Although Nallapati et al. (2016) [6] work has successfully built the RNN summarizer-decoder as the basis of the model, there was insufficient time for training, Model 3 was mostly trained on misdate, and had several challenges with the Collab GPU. For the first time 10 times, Model 3 was trained on 100,000 sentence-headline combinations. The 200 Glove vector and vocabulary are used with roughly 200 encoders and decoders. The loss decreased between 8,667 and 6,584, however the model started creating completely UNK toks after evaluation of the development forecasts, for maximum output length. Later examined the data used to train the model and the manner through which initialized the vocabulary in the model to detect the problem.

The most common V-words from the entire data package are used to construct a buffer matrix that starts with a V-size vocabulary. The V = 10,000 words most typically referred to are still significantly less than the first 100,000 single words of the complete unnoted English mega word Corpus. Many truth values have therefore been declared as UNK tokens. UNK was the largest term in truth, and UNK's recurrent output may minimize losses.

The RNN grabber model decided without sufficient time to utilize a large data set to generalize a model to unnoticed

information, using a small dataset in which model vocabulary is identical to the words in learning & development. This solves the problem of repetitive UNK token impressions and even prints intelligible words but only at the 80th of the 600s the results were inspected.

5. Conclusion and Future Work

This has been the first experience in establishing a large-scale data project, and the work that I done is quite different from any other works that I previously done. The creation of every starting code was a significant problem. The need to grasp what each functions call accomplished, but to rely on the code of earlier.

I had never built scripts to analyze large volumes of text, thus it was incredibly fascinating to think about each design decision. Came across very bad documentation and weird constraints in contrib libraries when used TensorFlow, which had to solve in a self-manner. Then finally modified the source code numerous times to try debugging one or two problems. The most interesting thing, once I had the baselines, was to explore the model.

I can't go as far as my guide would like, But I do know how vital it is to give up time for all the variables - as seen above, I seem to be in the right direction with the final version, a very modest collection of data and many more epochs Out of 600 80 times and the results of the test and train data can be observed in the fig. 2 and fig. 3. The project is optimistic that if we run the final model across the epochs of this data set (this will happen shortly after this submission), a meaningful summary will work much better.

References

- [1] Aref, I. Moawad and M. 2012. "Semantic graph reduction approach for abstractive text summarization." Computer Engineering System (ICCES), 2012 Seventh International Conference.
- [2] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. "Neural Machine Translation by Jointly Learning to Align and Translate."
- [3] S. M. Harabagiu and Finley L'ac'atusu, 2002. "Generating Single and Multi-Document Summaries with GISTEXTER." In U. Hahn & D. Harman, Proceedings of the workshop on automatic summarization 30-38.
- [4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. 2018. "Bert: Pre-training of deep bidirectional transformers for language understanding."
- [5] Jean, Sébastien, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2016. "On Using Very Large Target Vocabulary for Neural Machine Translation – Sampled Softmax." The Neural Perspective.
- [6] Nallapati, Ramesh, Bowen Zhou, Cicero Dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. "Abstractive Text Summarization Using Sequence-to-sequence RNNs and Beyond." Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning.
- [7] S. N. Munot and P. N. P. I. Sharvari. 2015. "Conceptual framework for abstractive text summarization." International Journal on Natural Language Computing (IJNLC).