# Context Aware Text Summarizer

ANASHKA NAINA U , MOHAMMED SHIRAZ AK
ROHIT P, SARATH SREEDHAR

## Abstract

**Keywords:** Text summarization, Semantic representation, Abstractive model, Neural networks, Word embeddings, GloVe, seq2seq RNN model, Attention layer

There exist two fundamental approaches for text summarization, which are extractive and abstractive summarization. In an extractive summarizer model, important words or phrases are identified from a document and the summary retains this set of words or phrases. Humans summarize text by creating a semantic representation of the content in the brain. The way humans forms this semantic representation is an abstractive model which can be simulated with the help of neural networks. The model that we examine in this project, makes use of word embeddings, initially, word embeddings are created using pre-trained GloVe model. This word embeddings are then used to train a seq2seq RNN model which is used for generating a summary. An attention layer is used which acts as a memory layer for the model to keep track of the content to be tracked. The dataset used for the initial training process is The Signal Media One-Million News Articles Dataset. Further training would be proceeded based on the availability of standard datasets.

## 1 Introduction

Today we are living in a busy world which is overwhelmed by information from different sources like blogs, news article, etc. It is so hard to absorb only necessary information from long articles because it is time-consuming. In such situations, it will be a blessing to get these long articles summarized. While summarizing the text the meaning and important information of the text should be preserved.

Text summarization can be extractive or abstractive. The words phrases or sentences from the document will be selected to generate a summary in extractive summarization. The summaries generated through extractive summarization usually be as perfect as human-written summaries. Abstractive text summarization follows another way. After thoroughly understanding the text, rephrase it into a shorter version. The summary may contain new words. The method of abstractive text summarization is somewhat complex than extractive. It may associate the capabilities of generalization, paraphrasing and may use some external knowledge.

In our model, we use GloVe which is a word-word co-occurrence count that have the potential for encoding some form meaning. Seq2seq model simulates the working brain and the attention model helps to store knowledge.

In the case of GloVe, the counts matrix is preprocessed by normalizing the counts and log-smoothing them. Compared to word2vec, GloVe allows for parallel implementation, which means that its easier to train over more data. It is believed (GloVe) to combine the benefits of the word2vec skip-gram model in the word analogy tasks, with those of matrix factorization methods exploiting global statistical information. GloVe is essentially a log-bilinear model with a weighted least-squares objective. The model rests on a rather simple idea that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning which can be encoded as vector differences.

This project report will proceed as follows, the

rest of this chapter discusses the motivation for the project and contribution from our side towards this project. The Chapter 2 discusses the background and related works, the Chapter 3 discusses the underlying concepts governing the working of this project. The remaining chapters discusses the system architecture and discusses the current working sate of the work and its evaluation.

## 2 Concepts

### 2.1 Encoder Decoder Architecture

The encoder-decoder architecture(Sutskever et al., 2014; Luong et al., 2015) comprises of an encoder and a decoder components. Both these components/parts are recurrent neural networks which make use of neural networks to improve the performance of the sequence to sequence mapping of a translation process.

The input given to the encoder is the text of the document that we want to summarize and the text of the document is fed one word at a time. Each word is converted into the corresponding distributed representation (Bengio et al., 2015) by passing the word through an embedding layer, thus the name word embedding. The distributed representation refers to a form which makes the conversion of sequence to vectors easier with the help of sub-sampling of frequent words like 'a', 'the',etc.. and Softmax layer. Instead of a softmax layer, there is scope for in-cooperating a negative sampling which tries to reduce the probability of occurrence of two words close to each other, this process is a less complex and computationally easier work. The distributed representation thus produced by the embedded layer is then combined using a multi-layer neural network with the hidden layers generated after feeding in the previous word to the encoder, or all zeros for the first word in the text of the document. The input of the decoder is the hidden layers created after feeding in the final word of the input text of the document. End of sequence(eos) symbol is fed as input first to the decoder and then uses the embedding layer to create distributed representation of the result of the encoder. The decoder then creates each word of the headline ending with the end-of-sequence symbol by using a softmax layer and the attention mechanism. After the generation of each word, this word is fed as input to the encoder-decoder network and creates the next word and so-on.

The loss function we use is the log loss function

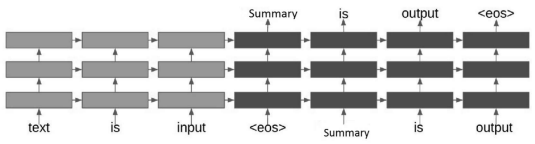


Figure 1: Encoder-decoder neural network architecture

$$-logP(y1,\ldots,yT'|x1,\ldots,xT) = -\sum_{t=1}^{T'} logP(yt|y1,\ldots,yt-1 \quad (1)$$

Here x and y represent input and output words respectively.

To stand unique in terms of efficiency compared to the prior models, the model discussed here include 'teacher forcing'(Goodfellow et al., 2016) during the training phase. Teacher forcing is an optimization procedure done during the training phase that helps in training the recurrent neural network faster and more efficiently. This is done by utilizing the output from a previous time step as the input for the current layer. Instead of generating a new word at each run and feeding it in as input for generating the next word, an expected word from the actual headline is fed to the network. However, during the testing, this practice is not followed, and the previously generated word from the previous layer is introduced when generating the next word. This gives rise to a break between training and testing. To overcome this break, during training, the network is fed randomly with a generated word, instead of the expected word as in the model discussed earlier (Bengio et al., 2015). Specifically, we do this 10% of the time, as also done in (Chan et al., 2015).

### 2.2 Attention

The attention mechanism is a mechanism that helps the network to remember/focus certain parts of the source document better, such as names, numbers, facts, etc... The attention mechanism is used after each word is output in the decoder. For each output word, the weighing mechanism calculates the weight for each input word that determines how much attention/focus should be paid to that input word. The total weights adds up to one and are used to calculate the weighted average of the last layers created after processing each input word. This weighted average, referred to as con-

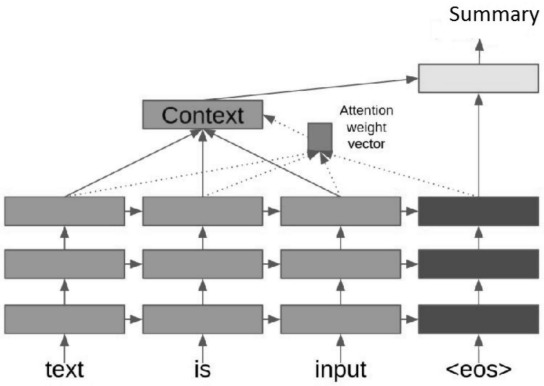


Figure 2: Complex Attention

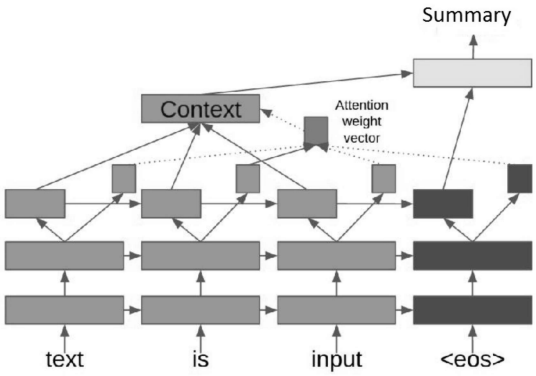


Figure 3: Simple Attention

text, is then entered into the softmax layer along with another hidden layer from the current decoding step. There are two different attention mechanism we looked into : The first attention mechanism, which we refer to as complex attention. This mechanism is shown in Fig: 2. The attention weight for the input word at position t, computed when outputting the t' th word is:

$$a_{y_{t'}}(t) = \frac{exp(h_{x_t}^T h_{y\_t'})}{\sum \frac{T}{t} exp(h_{x_{t-}}^T h_{y_{t'}})} \quad (2)$$

where $h^T_{xt}$ represents the last hidden layer generated after processing the t-th input word, and $h_{y_{t'}}$ represents the last hidden layer from the current step of decoding. The main highlight of this mechanism is that the same hidden units are used for computing the attention weight as for computing the context of the source. The second attention mechanism, which is referred to as simple attention, is a slight variation of the complex mechanism that makes it easier to analyze how the neural network learns to compute the attention weights. This mechanism is shown in Fig: 3. Here, the hidden units of the last layer generated after processing each of the input words are split into 2 sets: one set of size 50 used for computing the attention weight, and the other of size 550 used for computing the context. Analogously, the hidden units of the last layer from the current step of decoding are split into 2 sets: one set of size 50 used for computing the attention weight, and the other of size 550 fed into the softmax layer. Aside from these changes the formula for computing the attention weights, given the corresponding hidden units, and the formula for computing the context is kept the same.

## 3 System Architecture

The system comprises of two recurrent neural networks . The first one is an encoder network which takes in an input sequence and creates an encoded representation of it. The second one is a decoder network which takes in the earlier encoded representation as to its input and it will generate an output sequence decoding it. The model comprises of three stacked LSTM networks. The weight of the first layer is initialized with pre-trained GloVe embeddings which are computationally feasible. The embedding layer is meant to turn input into fixed-size vectors. The cross-entropy losses are minimized using rms-prop (Zaremba et al., 2014). The decoder network has the same LSTM architecture of the encoder network. The model is initialized with same pre-trained GloVe embedding weights. The input is the vector representation generated after feeding in the last word of the input text. It will generate its own representation using its embedding layer. The next step is to convert this representation into a word. The decoder will generate a word as its output and that same word will be fed in as input when generating the next word until it produces a summary. An attention mechanism is used when outputting each word in the decoder. For each output word, it computes a weight over each of the input words that determines how much attention should be given to that input word. All the weight sum up to 1 and are used to compute a weighted average of the last hidden layers generated after processing each of the inputted words. The weighted average is fed into the softmax layer along with the last hidden layer from the current step of the decoder(Lopyrev, 2015).

## 4 Evaluation

BLEU metric was used for the evaluation of generated summary. BLEU uses precision for measurement. It computes the number of overlapping n gram. Uni-gram based BLEU is used for evaluation. The actual summary generated is compared with target summary. The word-word occurrences is computed using BLEU score implemented in nltk package. BLEU is a metric used widely for machine based translation evaluation. The summaries are tokenized and fed to the function. The average of BLEU score is computed over hundred text sample. The max score obtained is one using this metric which resembles an exact match between generated and actual summaries. The score reduces when there is no word match between actual and generated summaries. For an abstractive summarization process the metrics as some shortcomings, An abstractive summary tends to replace word used in text to some alternative for enhanced understanding, so a word co-occurrences metric doesnt actually model an exact scenario of use-case. Human based evaluation are more advisable than system based metrics. The performance of BLEU metric also depends on the quality of actual summary. This intrinsic evaluation doesnt account for acceptability or further usage of summary.

## 5 Conclusion

The project describes the development of a text summarizer. Apart from the conventional text summarizers which summarize based on the word count/importance, the model we discussed in this report summarizes a textual document in an abstractive way. The abstractive manner is similar to the semantic representation of the human perceives while summarizing a text. As the number of works done in this field is minimal the development of the work faced quite a number of difficulties. The major difficulty was the absence of a standard dataset and depreciation of certain packages by the tensorflow.

The model starts with pre-processing of the textual document, for that purpose we make use of GloVe which is an unsupervised learning algorithm which takes the word-to-word co-occurrence count statistics from a corpus to generate a vector representation for words. This representation has the potential for encoding some form of meaning for the representation. The Seq2seq encoder-decoder model simulates the working of the human brain and the attention model helps to store knowledge which tries to mimic the memory capacity of the human brain. The possible future works to be done as a part of the project are:

- To create a dataset for analyzing context in textual data.
- To formulate an appropriate data structure for context representation.

## References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1171–1179, Cambridge, MA, USA. MIT Press.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. 2015. Listen, attend and spell. *CoRR*, abs/1508.01211.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Konstantin Lopyrev. 2015. Generating news headlines with recurrent neural networks. *CoRR*, abs/1512.01712.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *CoRR*, abs/1409.2329.