International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

Text Summarization using Deep Learning

Kasimahanthi Divya[1], Kambala Sneha[2], Baisetti Sowmya[3], G Sankara Rao[4]

[1,2,3,4]*Department of Computer Science and Engineering,*
Gayathri Vidya Parishad College of Engineering for Women
(Affiliated by Jawaharlal Nehru Technological University, Kakinada),
Visakhapatnam, Andhra Pradesh, India

Abstract - *Text summarization is a process of extracting the context of a large document and summarize it into a smaller paragraph or a few sentences. Text summarization plays a vital role in saving time in our day to day life. It is also used in many bigger project implementations of classification of documents or in search engines. This paper presents a method of achieving text summaries accurately using deep learning methods.*

Key Words: Text Summarization, Deep Learning, Long Short Term Memory, Natural Language Processing, Abstractive summary, Extractive summary

1. INTRODUCTION

Text summarization is a process of producing brief and concise summary by capturing the vital information and the comprehensive meaning. Text summarization is achieved by natural language processing techniques by using algorithms like page rank algorithms etc. While these algorithms fulfil the objective of text summarization, they cannot generate new sentences which are not in the document like humans. They can also have grammatical errors. This is where Deep Learning comes to our rescue. The use of deep learning builds an efficient and fast model for text summarization. The use of deep learning methods helps us generate summaries which can be formed with new phrases and sentences and also which are grammatically correct.

Text Summarization is broadly classified into two types:

1. Abstractive Summarization

2. Extractive Summarization

1.1 Extractive Text Summarization

The extractive text summarization involves pulling key phrases from the source document and combining them to make a summary. We identify important words or phrases from the text and extract only those for the summary.

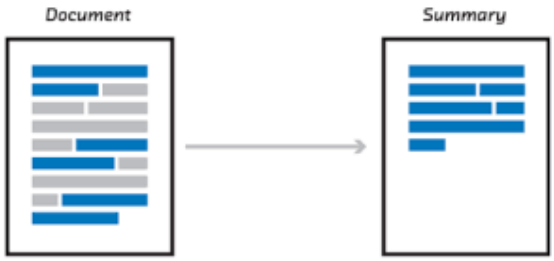

Fig -1: Extractive Text Summarization

1.2 Abstractive Text Summarization

The abstractive text summarization can create new phrases and sentences that relay the most useful information from the original text. The sentences generated through this method may not be present in the original document.

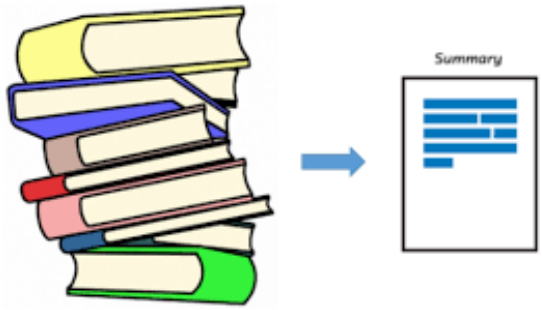

Fig -2: Abstractive Text Summarization

2. IMPLEMENTATION

We implement the abstractive method using the deep learning technique called Long Short Term Memory (LSTM) which is a type of Recurrent Neural Network Algorithms. The data used for this project is CNN_dailymail dataset.

2.1 Data

The data used is the CNN_dailymail dataset. It has two features: article and highlights. The article includes the document that is to be summarized. It is the news article. Highlights are the headlines of the corresponding news which are used as summaries.

2.2 Method

The model used is the abstractive method which is implemented using deep learning techniques.

2.3 Algorithm

The algorithm used is the LSTM or Long Short Term Memory model which is a type of Recurrent Neural Network model.

2.4 Model

The model used is sequence to sequence model. Sequence-to-sequence learning is a training model that can convert sequences of one input domain into the sequences of another output domain. It is generally used when the input and output of a model can be of variable lengths.

3. DATA PREPROCESSING

Performing basic pre-processing steps is very important before we get to the model building part. Using messy and uncleaned text data is a potentially disastrous move. So, we will drop all the unwanted symbols, characters, etc. from the text that do not affect the objective of our problem.

We will perform the below preprocessing tasks for our data:

- Convert everything to lowercase
- Remove (‘s)
- Remove any text inside the parenthesis ( )
- Eliminate punctuations and special characters
- Remove stopwords
- Remove short words

4. METHODS

4.1 CleanData()

It is used to clean the data by using preprocessing steps mentioned earlier.

4.2 BuildDataset()

It is used to build train and test data sets

4.3 BuildDict()

It is used to build dictionary where keys are words and values are random and unique numbers. It also builds another dictionary with keys as unique numbers and values as words. These are used in tokenization of words so that the input to the model is a set of numbers rather than words so as to make the computation easier using vectors.

4.4 Tokenize()

It is used to tokenize the data and send it to the model. Tokenizing data is important as the networks need numerical data to work on rather than raw data with characters.

5. ALGORITHM

In the current days, we are trying to create algorithms which can help us replicate the human brain and achieve its functionalities. This has been achieved by the neural networks. Neural Networks are the set of algorithms that an recognize patterns in the data. They closely resemble the human brain and have the capability to create models that can work or function like a human brain. Recurrent Neural Network (RNN) are a type of neural networks. They are feedforward neural networks which have an internal memory. In a traditional neural network, the input and the output sequences are independent of the each other. But in order to predict a sequence or a sentence, we need to know the previous words to predict the next word. Hence, we need internal memory. RNN helps us store the previous memory with the help of hidden states which remembers information about previous sequences.

The RNN is named so as it recurrently performs the same function on all input of data and the hidden layers. This reduces the complexity of having to store various parameters for each of the layers in the network thus saving the memory. The output of the current input depends on the past outputs too. After the output is produced, it is sent back to the same network so that it can be stored and used for the processing of next output in the same sequence. In order to generate an output in RNN, we consider the current input and the output that was stored from the previous input.

RNNs work perfectly when it comes to short contexts. But when we want to create a summary of a complete article, we need to capture the context behind the complete input sequence and not just the output of the previous input. Hence, we need a network that can capture the complete context like a human brain. Unfortunately, simple RNN fails to capture the context or the long term relation of the data that is it cannot remember or recall data in the input that occurred long before and hence cannot make an effective prediction. RNN can remember data or context only for a short term. This is called vanishing gradient problem. This issue can be resolved by a slightly different version of RNN - The Long Short Term Memory Networks

Long Short-Term Memory (LSTM) networks are a better version of RNN. They can remember the past data easily by resolving the vanishing gradient problem. LSTM uses back propagation to train the model. LSTM is well-suited for predictions and classifications of data sequences of unknown durations. They can also be used in language translation and text summarization methods.

6. MODEL

The model used here is sequence to sequence model. Sequence-to-sequence learning is a training model that can convert sequences of one input domain into the sequences of another output domain. It is generally used when the input and output of a model can be of variable lengths. It is a method of encoder-decoder based machine translation that maps an input of sequence to an output of sequence with a tag and attention value. The idea is to use two LSTMs that will work together with a special token and try to predict the next state sequence from previous sequence.

6.1 Encoder-Decoder architecture:

The Sequence to Sequence model uses a method of encoder-decoder based machine translation. Encoder-Decoder architecture is used in predicting sequences when the length of output and input data may vary. The input sequence is read entirely by the encoder and a fixed length internal representation is generated. The internal representation captures the entire context of the input data sequence. The decoder network uses this internal representation to predict the output words until the end of the sequence token is reached.

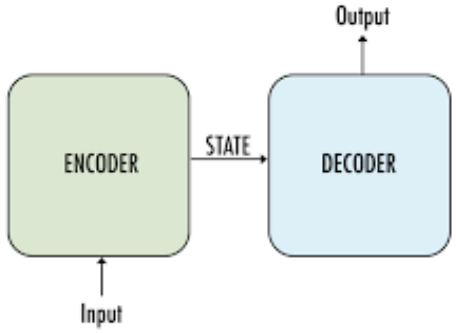

Fig -3: Encoder Decoder

6.2 Encoder

An encoder is an LSTM network which reads the entire input sequence. At each time step, one word from the input sequence is read by the encoder. It then processes the input at each time step and captures the context and the key information related to the input sequence. It takes each word of input(x) and generates the hidden state output (h) and the cell state which is an internal state(c). The hidden state(h_i) and cell state(c_i) of the last time step are the internal representation of the complete input sequence which will be use to initialize the decoder.

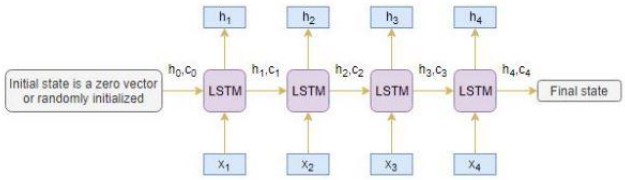

Fig -4: Encoder

6.3 Decoder

The decoder is also an LSTM network. It reads the entire internal representation generated by the encoder one word at a time step. It then predicts the same sequence offset by one time step. The decoder is trained to predict the next word in the output sequence given the previous word based on the contextual memory stored by the LSTM architecture. Two special tokens <start> and <end> are added at the beginning and at the end of the target sequence before feeding it to the decoder. We start predicting the target sequence by passing one word at a time. The first word of output of the decoder is always <start> token. The end of the output sequence is represented by <end> token.

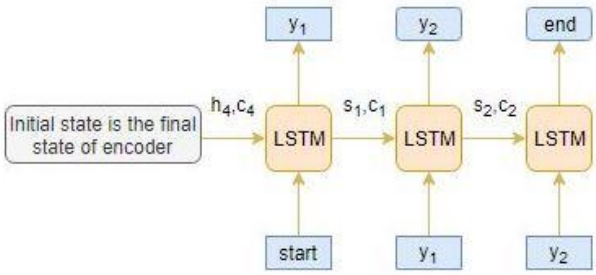

Fig -5: Decoder

The above architecture of the model is built using the TensorFlow library which used to build layers in neural networks. The final architecture of the model will be as shown below

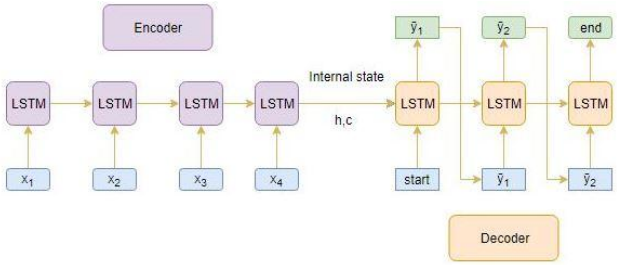

Fig -6: LSTM Seq2Seq model architecture

6.4 Attention layer

A Sequence to Sequence model with an attention mechanism consists of encoder, decoder and an attention layer. Attention mechanism is used to secure individual parts of the input which are more important at that particular time. It can be implemented by taking inputs from each time steps and giving weightage to time steps. The weightage depends on the contextual importance of that particular time step. It helps pay attention to the most relevant parts of the input data sequence so that the decoder can optimally generate the next word in the output sequence.

7. VALIDATION

In this text summarization model, we use ROUGE as the validation metric. ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is used for evaluating automatic text summarizations and machine translations. It consists of a set of metrics which are used for evaluation. It compares the result automatically produced using the model by the machine against human produced results. In the text summarization model, the human produced summaries which are used for reference are reference summaries and the summary generated using the model are called system summaries. The model is evaluated based on the extent to which the system summaries are similar to the reference summaries. This is captured by recall.

Here, Recall means how much the system summary is able to capture or recover the essence of reference summary. If we are just considering the individual words, it can be computed as:

(Number_of_overlapping_words)÷

(Total_words_in_reference_summary)

ROUGE-1 refers to overlap of unigrams between the system summary and reference summary.

ROUGE-2 refers to the overlap of bigrams between the system and reference summaries.

ROUGE-N measures unigram, bigram, trigram and higher order n-gram overlap

Given below are the statements while running ROUGE scores on unigrams.

```
a.injury=(nsubj)=>leaves
<BasicElement: injury-[nsubj]->leave>
a.kwan=(dobj)=>leaves
<BasicElement: kwan-[dobj]->leave>
b.olympic=(amod)=>hopes
a.leaders=(nsubj)=>slam
<BasicElement: leaders-[nsubj]->slam>
b.philippines=(compound)=>laws
a.laws=(dobj)=>slam
<BasicElement: laws-[dobj]->slam>
a.attacks=(nsubj)=>tough
<BasicElement: attacks-[nsubj]->tough>
a.law=(dobj)=>tough
<BasicElement: law-[dobj]->tough>
a.sales=(pobj)=>gm
<BasicElement: sales-[pobj]->gm>
a.sales=(nsubj)=>fall
<BasicElement: sales-[nsubj]->fall>
a.percent=(dobj)=>fall
<BasicElement: percent-[dobj]->fall>
b.celebrate=(acl)=>libyans
a.victory=(dobj)=>celebrate
<BasicElement: victory-[dobj]->celebrate>
a.thousands=(nsubj)=>celebrate
<BasicElement: thousands-[nsubj]->celebrate>
```

Fig -7: ROUGE unigram values

8. CONCLUSION

The increasing growth of the Internet has made a huge amount of information available. It is difficult for humans to summarize large amounts of text. Thus, there is an immense need for automatic summarization tools in this age of information overload.

The International Data Corporation (IDC) projects that the total amount of digital data circulating annually around the world would sprout from 4.4 zettabytes in 2013 to hit 180 zettabytes in 2025. That's a huge amount of data circulating in the digital world. There is a dire need of algorithms which can be used to automatically shorten the amount of data with accurate summaries that capture the essence of the intended messages. Furthermore, applying text summarization reduces reading time and accelerates the process of researching for information playing a major role in current era of rapid development and digitalisation.

Humans are generally quite good at this task as we have the capacity to understand the meaning of a text document and extract salient features to summarize the documents using our own words. However, automatic methods for text summarization are crucial in today's world where there is an over-abundance of data and lack of manpower as well as time to interpret the data.

REFERENCES

[1] U. Hahn, I. Mani, "of Automatic Researchers are investigating summarization tools and methods that", IEEE Computer33. 11, pp. 29-36, November 2000.

[2] E. Lloret, M. Palomar, "Text summarization in progress: a literature review" in Springer, Springer, pp. 1-41, 2012

[3] K. Spärck Jones, "Automatic summarizing: The state of the art", Information Processing & Management, vol. 43, pp. 1449-1481, nov 2007

[4] A. Khan, N. Salim, "A review on abstractive summarization methods", Journal of Theoretical and Applied Information Technology, vol. 59, no. 1, pp. 64-72, 2014

[5] E. Baralis, L. Cagliero, A. Fiori, P. Garza, "MWI-Sum: A Multilingual Summarizer Based on Frequent Weighted Itemsets", *ACM* Transactions on Information Systems, vol. 34, no. 1, pp. 1-35, 2015

[6] M. Kamal, "Text Summarization in Bioinformatics", 15th International Conference on Computer and Information Technology, pp. 592-597, 2012

BIOGRAPHIES

Kasimahanthi Divya is currently pursuing her B.Tech degree in Department of Computer Science & Engg at Gayatri Vidhya Parishad College of Engineering For Women, Near Kommadi Junction, Madhurawada, Visakhapatnam. She worked on projects based on Deep Learning, Machine Learning, blockchain, Web Development and cloud computing. She secured job offers at Amazon, TCS and Infosys. She is working at AWS as intern.

Kambala Sneha is currently pursuing her B.Tech degree in Department of Computer Science & Engg at Gayatri Vidhya Parishad College of Engineering For Women, Near Kommadi Junction, Madhurawada, Visakhapatnam. She worked on Web Development

Baisetti Sowmya is currently pursuing her B.Tech degree in Department of Computer Science & Engg at Gayatri Vidhya Parishad College of Engineering For Women, Near Kommadi Junction, Madhurawada, Visakhapatnam. She worked on machine learning.

G. Sankara Rao is currently working as an Assistant Professor in Department of Computer Science & Engg at Gayatri Vidhya Parishad College of Engineering For Women, Near Kommadi Junction, Madhurawada, Visakhapatnam. He has more than 12 years of experience in Teaching field. His resarch interests includes Deep learning, Wireless Sensor Networks,Networking and Security. He completed M.Tech from Andhra University, and pursuing Ph.D at JNTUK University.