International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-9 Issue-1, May 2020

Text Summarization using Deep Learning

Saketh Mattupalli, Apurva Bhandari, B.J Praveena

Abstract*: In this century, Artificial Intelligence AI has gained lot of popularity because of the performance of the AI models with good accuracy scores. Natural Language Processing NLP which is a major subfield of AI deals with analysis of huge amounts of Natural Language data and processing it. Text Summarization is one of the major applications of NLP. The basic idea of Text Summarization is, when we have large news articles or reviews and we need a gist of news or reviews with in a short period of time then summarization will be useful. Text Summarization also finds its unique place in many applications like patent research, Help desk and customer support. There are numerous ways to build a Text Summarization Model but this paper will mainly focus on building a Text Summarization Model using seq2seq architecture and TensorFlow API.

Keywords*: Text Summarization, Artificial Intelligence, Natural Language Processing, Machine Learning.

I. INTRODUCTION

Modern Technology and with the rise of Internet today, we have huge amounts of language data like news articles reviews etc. Text Summarization helps us to summarize these language data so that we can understand the gist of the entire article very easily. Text Summarization is even difficult task for humans. If a person is dealing with the task of Text Summarization, he must have great command on the subject, topic and also the Natural Language. It is difficult task to train the computer or machine to perform the tasks which may involve Natural Language, because producing summary version of the original content while preserving the overall meaning is difficult for Machines or computers. However, with the recent advancements in Natural Language Processing NLP like seq2seq Architecture [1], Attention Mechanism, Word Embeddings [2] helps us in developing complex models which involve Natural Language. This paper mainly focuses on Abstractive Text Summarization model using TensorFlow, an open source Deep Learning framework developed by Google.

Manuscript received on May 02, 2020.
Revised Manuscript received on May 21, 2020.
Manuscript published on May 30, 2020.

* Correspondence Author

Saketh Mattupalli*, is currently pursuing B.E Degree program in Computer Science and Engineering in Matrusri Engineering College, Saidabad, Affiliated to Osmania University, Hyderabad, Telangana, India. His research work focuses on Natural Language Processing and Computer vision.

Apurva Bhandari, is currently pursuing B.E Degree program in Computer Science and Engineering in Matrusri Engineering College, Saidabad, affiliated to Osmania University, Hyderabad, Telangana, India Her research work focuses on Deep Learning and Machine Learning.

B.J Praveena, is currently working as a Asst.professor in Matrusri Engineering College, Saidabad, Affiliated to Osmania University, Hyderabad, Telangana, India. Her research work mainly focuses on cloud computing and Artificial Intelligence.

A. Text Summarization Approaches

Text Summarization can be broadly classified int to two types:

1.Extractive Summarization

2.Abstractive Summarization

Extractive Summarization relies on finding the important phrases, passages from the given original text and then creates the summarized version using those phrases as shown in the Figure 1. The summarized version is part of original text, we cannot find any new phrases in the summarized version. We can visualize it as the highlighter highlighting the important sentences of the text. Text Rank algorithm is most widely used for Extractive Summarization [3].

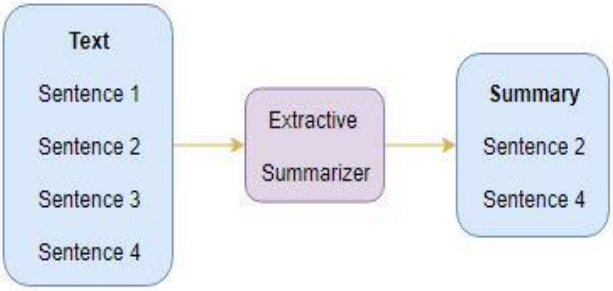

Fig. 1. Extractive Text Summarization

Abstractive Summarization [4] is an advanced approach which understands the context of the original text and then forms the summary. Unlike Extractive Summarization this produces new sentences in summarized version which were not present in original text. It is a difficult task but with the help of seq2seq model architecture Abstractive Summarization Model can be developed. This paper will be focusing on developing Abstractive Summarization Model.

B. Introduction to Seq2seq Architecture

Sequence-to-Sequence or "seq2seq" model is an end to end model generally made up of Recurrent Neural Networks RNN [5]. These are also called as encoder-decoder Models as shown in Figure 2.

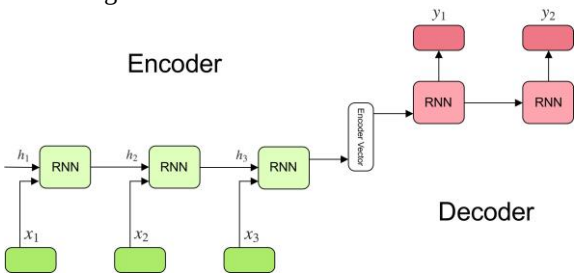

Fig. 2. Encoder Decoder Network

Encoder takes input as a sequence of words and encodes this input into corresponding hidden vectors called "context vectors".

Encoder will generally use an RNN cell to read the input tokens one at a time as shown in the figure 2. Decoder will decode the context vectors generated by Encoder and produces output as a sequence of words. Like Encoder, Decoder also uses RNN cell – Long Short Term Memory LSTM Network [6]. In Decoder an <EOS> End of Sentence token is appended at the end of input sentence. Both Encoder and Decoder networks are trained simultaneously so that both learn the same context vector representation. Seq2seq Model doesn't require inputs and outputs to be of equal length. Seq2seq architecture has revolutionized the way of Natural Language Processing with the help of Deep Neural Networks.

II. IMPLEMENTATION

The various phases in developing a Text Summarization Model are:

1. Gathering Data
2. Data Pre-processing
3. Selecting Word Embeddings
4. Exploratory Data Analysis
5. Building Encoding Layer
6. Building Decoding Layer
7. Building Seq2seq Model
8. Setting Hyperparameters
9. Optimization
10. Training the Model
11. Save the Model
12. Testing

These are the main building blocks for Text Summarization Model

A. Gathering Data

In order to train the seq2seq Model we need to use a huge dataset which consists of original text along with summarized versions. Preparing such data is a difficult task because a human has to write the summarized versions of original text. However, there is a Food reviews dataset from Amazon[7]. This paper demonstrates the usage of Amazon fine food reviews dataset which consists of summary as well as original text as shown in the figure 3.

reviews - DataFrame

Index	Summary	Text
0	Good Quality Dog Food	I have bought several of the Vitality canned dog food products and have found them all to be of good quali…
1	Not as Advertised	Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsalted. Not sure…
2	"Delight" says it all	This is a confection that has been around a few centuries. It is a light, pillowy citrus gelatin wi…
3	Cough Medicine	If you are looking for the secret ingredient in Robitussin I believe I have found it. I got this in…
4	Great taffy	Great taffy at a great price. There was a wide assortment of yummy taffy. Delivery was very quick.…
5	Nice Taffy	I got a wild hair for taffy and ordered this five pound bag. The taffy was all very enjoyable with man…
6	Great! Just as good as the exp…	This saltwater taffy had great flavors and was very soft and chewy. Each candy was individually wrapped…
7	Wonderful, tasty taffy	This taffy is so good. It is very soft and chewy. The flavors are amazing. I would definitely recomme…

Fig. 3. First few rows of Dataset

From this data seq2seq Model learns various texts and their summaries.

B. Data Pre-processing

The gathered data cannot be fed directly to the seq2seq Model. For preprocessing the data there are packages in Python like Natural Language Tool Kit NLTK, Regular Expressions(re.). Firstly, all the alphabetical characters are converted into lowercase. This task is achieved using the lower () method present in NLTK module. Now, the contractions have to be converted into their original long forms, for example, 'don't' has to be converted back in to 'do not'. This is a lengthy task because there is no python library available to perform contraction mapping, All the contractions have to be explicitly defined and then converted back into their original forms. After that, all the unwanted characters and symbols are removed. Stop words which does not have any significance are also removed. List of stop words are readily available in NLTK package. Removal of stop words help the model training faster.

C. Selecting Word Embeddings

Word Embeddings is a technique where individual words are represented as a real valued vector. Each word is represented as one vector and the vector values are learnt in a way that resembles Neural Network. Some of the best word embedding algorithms are word2vec, GloVe: Global Vectors for Word representation [8]. There is another pretrained word vector called Concept Net Number batch (CN) [9]. In this paper CN is used because of its better performance over GloVe as shown in the figure 4.

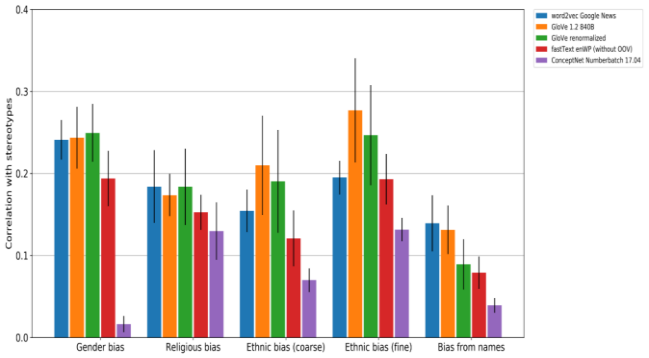

Fig. 4. Concept Net showing less demographic bias than other pre-trained word vectors.

CN is an ensemble of various word embeddings that includes GloVe also.

D.Exploratory Data Analysis

The size of the vocabulary in the dataset after pre-processing is 132880.A threshold value of 20 is chosen, words whose frequency is less than threshold value are discarded. Total number of words that are not present in ConceptNet but present in the dataset are 3840 which constitute to 2.7% of the vocabulary in the dataset. Now vocabulary of the dataset is limited to the words that are either in CN or the words which repeat more than the threshold value. Thus, for each word very good embeddings are selected and also seq2seq model can better understand the relation between words when they see the word many times. Now, total number of words selected is 60443 which is 46% of the original vocabulary. All the words in the dataset are converted in to integers such that each word will have a unique number, also called as word id assigned to them. If the word is not available in the selected vocabulary then replace it with UNK token. Some of the special tokens used in seq2seq model are shown in the below

table-1.

Table- I: Special tokens used in Data preparation

S.No	Token	Description
1	<UNK>	Unknown token, it is used to replace the words that are not present in our vocabulary.
2	<PAD>	Seq2seq model requires each batch of data to be of same length. If the input sentence length is less than sequence length then padding token can be used to fill the gap.
3	<EOS>	End of Sentence, this token is appended at end of each input
4	<GO>	Start token, it is the first token that is fed to the decoder indicating the start of sentence.

Finally, summaries or reviews which contain more than 1 UNK token are discarded. This ensures that the input dataset will contain meaningful data. word embeddings matrix is also created for the vocabulary, if the word is not present in the ConceptNet a random embedding is assigned to it. This embedding matrix is helpful later while building seq2seq model.

E. Building Encoding Layer

Encoding Layer consists of the bidirectional RNN with LSTM. Bidirectional RNNs will traverse the sequence in two directions. Here the LSTM cell reads one word at a time and also gets information about previous inputs and then updates the hidden state. The idea of bidirectional RNN is to use two independent RNNs working together. The input is fed to one RNN in sequential order and in reverse order for other RNN. After each time step outputs of both Neural Networks will be concatenated as shown in the figure 5.

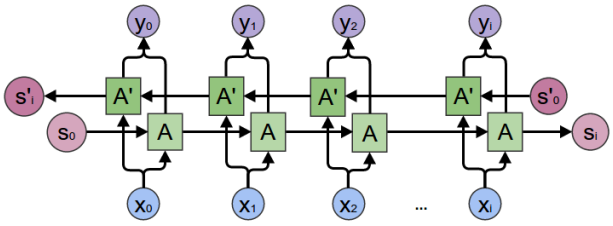

Fig. 5. Bidirectional RNN

F. Building Decoding Layer

The decoding layer consists of Decoding cell which is complex than encoding layer. Decoding cell used is 2-layer LSTM along with dropout, dropout is a regularization technique that is used to reduce the overfitting of the model thereby improving the performance of the model. All the layers of LSTM are processed one after another followed by a SoftMax function on the final layer and then generates the first output word. LSTM in the decoding cell uses the information from encoding layer as well as the information it has written before (as LSTMs can remember what they written previously) to generate the probability distribution for the next word and finally predict the next word.

G. Building seq2seq Model

Encoding and Decoding architecture is not sufficient to build a good seq2seq model. Attention Mechanism is used to improve the performance of seq2seq model. The drawback of encoding-decoding architecture without attention is, the decoder gets information only from the final hidden state of encoder, this information is not enough for the model to understand the context of the input because different parts of output may consider different parts of input to be important. With the help of Attention Mechanism, decoder can also access the intermediate hidden states of the encoder (along with the final states) at every decoding step. This enables decoder to understand the context of the input and decides what words are important at that point of time. Bhadanau attention Mechanism [10] is used in this seq2seq Model. Now the seq2seq model is built.

H. Setting Hyperparameters

The values of hyperparameters must be defined before training process begins. Hyperparameters describe the Neural Network structure (like RNN size) and also determine how the model is being trained. The following hyperparameters are defined for our Model:

- Epochs = 50
- Batch size = 64
- RNN size = 256
- Learning rate = 0.005
- Keep Probability = 0.75

The above defined values are the optimal hyperparameter values, but in some cases they may not be optimal. In such cases a set of hyperparameter values were chosen and tune those hyperparameters to get optimal values, this process is called Hyperparameter tuning [11].

I. Optimization

Once the loss function is calculated, the gradients are obtained which can be used by the optimizer to minimize the loss function. Adam optimization algorithm [12] was used in this model which is better than Stochastic Gradient Descent SGD [13]as shown in the figure 6.

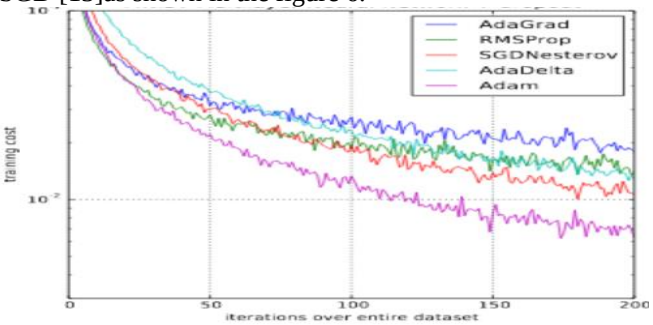

Fig.6. Graph showing performance of Adam optimizer.

Gradient clipping technique is used to prevent the common problems in RNN like vanishing gradient and Exploding gradient.

J. Training the Model

Training the network will take lot of time depending on the hardware. It is a better idea to train the models on Graphic Processing Unit GPU or Tensor Processing Unit TPU. Training over CPU will consume so much of time. This model was trained in google Colab which provides access to TPU. In training the model, the loss function is observed after every 20 batches and 3 update checks are made for each epoch, Model is training is stopped when there is no update in the consecutive 4 update checks.

K. Save the Model

Once the deep Learning model is trained and is ready for production deployment then the deep learning model has to be saved. When we need same model for another project, we can load directly the trained model. It is always better idea to save Deep Learning models after they got trained, as they may take lot of time for training. Deep Learning model can be saved in "HDF" format (High Definition file).

L. Testing

The model is performing very accurately. when the user gives input of length greater than 15 words then the model is accurately predicting the summary of length 2 to 13 words. The model has predicted summaries which contain the new vocabulary and phrases which are not present in the original input text. The predicted summaries are also abstractive. Thus, the model has learnt the context of the input sentence with the help of encoder, decoder and attention mechanism and produced very accurate summaries. The metric used to evaluate the performance of our model is ROUGE score which stands for Recall- Oriented Understudy for Gisting Evaluation. ROUGE score is calculated by comparing the model predicted summaries with original summaries which were written by humans. The model has showed a ROUGE-1 score of 0.46 which is pretty great.

III. RESULT AND DISCUSSION

While testing the model with custom inputs or user inputs the following process will take place, Firstly the input text is sent for pre-processing where the input text is converted into lower case and then contraction mapping is performed over it, stop words and insignificant words are also removed. Now for each unique word a word id is assigned. As we already saved the model, we have to load the model and give this input, then it produces the output sequence of word id which have to be mapped back to the word in this way seq2seq model produces the output. If the input sentence "the paradise biryani was tasty and authentic" is given, firstly it is sent for preprocessing, as there are no uppercase or special symbols, stop words are removed (the, was, and) and unique word ids are given to paradise, biryani, tasty and authentic. Now this data is given to encoder decoder architecture, where the encoder encodes this data into hidden context vector. Decoder starts decoding this context vector taking one input from the vector and the other input from it's previous output (decoding cell is LSTM it can remember it's previous output) , also considers the attention layer to understand the context of the input sentence at any point of time and produces the output sequence of word ids which were mapped back to the words with the help of id2word dictionary. Figure.7 shows the output produced by the model in jupyter notebook.

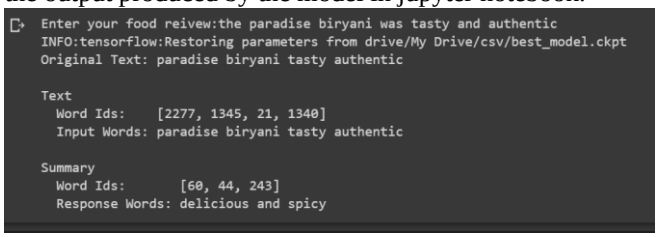

Fig.7. Implementation of Text Summarization Model when custom input is given.

As we can see from the figure 7, the response words in summary are "delicious and spicy" which were not present in the original input. The predictions were very accurate and abstractive. When the trained input data is given to model to test its performance the model produced accurate results as shown in the figure 8 below. The input shown in the figure below is "I have bought several of the Vitality canned dog food products and have found them all to be of good quality. The product looks more like a stew than a processed meat and it smells better. My Labrador is finicky and she appreciates this product better than most."

Fig.8.Comparing predicted summaries with original summaries.

From the above outputs we can see that the predicted summaries are almost equal to the original summaries. From the above outputs we observed that model is performing well in predicting summaries for food reviews. But when a news article or other input is given, model is not predicting very accurate summaries and the predicted summaries are extractive in nature. This is because model is trained on food reviews data and it can perform very well on food reviews.

IV. CONCLUSION

This paper has demonstrated the use of seq2seq modelling approach for Text Summarization. Advancements in Deep Learning, Natural Language Processing made the NLP models very powerful and efficient. There are also other ways of building text summarization model using, Pointer Generator network [14], Text Rank algorithm [15], but seq2seq model produced better results than the other models. Natural Language Processing is a major research area, with the technical advancements increasing exponentially in this area, NLP models will become much more powerful in the very near future. Need for automatic text summarization today is very high than ever before.

Text summarization helps in saving lot of time by enabling the users to understand the reviews or news in very short time. This paper started with definition of text summarization explaining various types of text summarization and finally built a model which can perform Abstractive Text summarization.

REFERENCES

1. Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, Chandan K. Reddy, Senior Member, IEEE, "Neural Abstractive Text Summarization with Sequence-to-Sequence Models: A Survey", https://arxiv.org/pdf/1812.02303.pdf
2. Amit Mandelbaum, Adi Shalev, Hebrew University of Jerusalm, (October 27, 2016). "Word Embeddings and Their Use in Sentence Classification Tasks" https://arxiv.org/pdf/1610.08229.pdf

3. Mehdi Allahyari,Seyedamin Pouriyeh,Mehdi Assef, Krys Kochut. , "Text Summarization Techniques: A Brief Survey", https://arxiv.org/pdf/1707.02268.pdf
4. N. Moratanch, Dr. S. Chitrakala, (2016) International Conference on Circuit, Power and Computing Technologies [ICCPCT], "A Survey on Abstractive Text Summarization", https://www.researchgate.net/publication/305912913_A_survey_on_abstractive_text_summarization
5. Zachary C. Lipton, John Berkowitz, "A Critical Review of RNN Neural Networks for Sequence Learning", https://arxiv.org/pdf/1506.00019.pdf
6. Sepp Hochreiter, Jurgen Schmidhuber, (1997) "LONG SHORT-TERM MEMORY", https://www.bioinf.jku.at/publications/older/2604.pdf
7. Retrived the dataset from Kaggle website: https://www.kaggle.com/snap/amazon-fine-food-reviews
8. Jeffrey Pennington, Richard Socher, Christopher D. Manning (2014) "GloVe: Global Vectors for Word Representation", https://www-nlp.stanford.edu/pubs/glove.pdf
9. Robyn Speer, Joshua Chin, Catherine Havasi (Dec 2016), "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge" arXiv:1612.03975
10. Andrea Galassi, Marco Lippi, Paolo Torroni (Feb 2019), "Attention, please! A Critical Review of Neural Attention Models in Natural Language Processing". arXiv preprint arXiv:1902.02181v1.
11. James Bergstra, Yoshua Bengio (2012), "Random Search for Hyper-Parameter Optimization", Journal of Machine Learning Research 13 (2012) 281-305.
12. Diederik P. Kingma, Jimmy Ba (Dec 2014). "Adam: A Method for Stochastic Optimization". Preprint arXiv:1412.6980
13. Sebastian Ruder (June 2017). "An overview of gradient descent optimization algorithms". Preprint arXiv:1609.04747v2
14. Abigail See, Peter J. Liu, Christopher D. Manning (Apr 2017), "Get To The Point: Summarization with Pointer-Generator Networks". Preprint arXiv:1704.04368
15. PRATEEK JOSHI, (NOVEMBER 1, 2018). "An Introduction to Text Summarization using the TextRank Algorithm (with Python implementation)". https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/

AUTHORS PROFILE

Saketh Mattupalli is currently pursuing B.E Degree program in Computer Science and Engineering in Matrusri Engineering College, Saidabad, Affiliated to Osmania University, Hyderabad, Telangana, India. His research work focuses on Natural Language Processing and Computer vision.

Apurva Bhandari is currently pursuing B.E Degree program in Computer Science and Engineering in Matrusri Engineering College, Saidabad, affiliated to Osmania University, Hyderabad, Telangana, India Her research work focuses on Deep Learning and Machine Learning.

B.J Praveena is currently working as a Asst.professor in Matrusri Engineering College, Saidabad, Affiliated to Osmania University, Hyderabad, Telangana, India. Her research work mainly focuses on cloud computing and Artificial Intelligence.