IJCSE International Journal of Computer Sciences and Engineering Open Access
Survey Paper Vol.6, Special Issue.11, Dec 2018 E-ISSN: 2347-2693

Indian Language Text Summarization Using Recurrent Neural Networks

Anjali A V[1*], N. Ramasubramanian[2], A. Santhanavijayan[3]

[1]Scholar Engineering, NIT, Trichy, India
[2]Dept. of Computer Science NIT, Trichy, India
[3]Dept of CS NIT, Trichy, India

Available online at: www.ijcseonline.org

Abstract— Text summarization is the process of creating short and accurate summary of a given text document. The paper is proposing an abstractive method of text summarization for text in Indian languages. The proposed algorithm uses an encoder-decoder recurrent neural networks with attention mechanism. The results observed were significantly better compared to the performance of already existing Indian language summarizer**.**

Keywords— Abstractive Summarization, LSTM, Recurrent Neural Network (RNN)

I. INTRODUCTION

Text summarization is the process of creating short and accurate summary of a given text document. Text summarization can be classified into two-abstractive and extractive. Extractive text summarization forms the summary by selecting subsets of words or sentences from the text document. Abstractive text summarization is the process of producing a short summary which highlights the salient ideas of a huge text document. The abstractive method is found to produce more relevant summary compared to extractive methods.

Deep-learning based models matches an input sequence into corresponding output sequence. This type of model is called sequence-to-sequence models which have been found useful in several domains like machine translation. This process can be thought of as mapping of input sequence of words present in source document to another set of sequence of words called summary.

Although machine translation and summarization sounds alike ,both are very different concepts. In text summarisation, the summary is very brief and is not dependent on the length of the text to be summarized. Also, a major difficulty in summarization is for the orginal document to be optimally compressed in a lossy manner and keeping sure that the main topics in the original document remains unchanged, whereas in Machine Translation, the translation should be lossless.

II.EXISTING WORK

Numerous automatic text summarization systems are available in all languages especially in English and other foreign languages. The Indian languages, summarization models are very few. Some of the text summarizer for Indian languages is discussed below.

A.Solutions Existing for Tamil

Sankar K et .al proposed a solution based on scoring of sentences for summarizing texts inTamil language. An average Rouge score was found to be 0.4723.

B.Solutions Existing for Kanada

Jayashree.R et .al used a method which produces extractive summaries of documents in the Kannada language. The proposed algorithm uses GSS (Galavotti, Sebastiani, Simi) coefficients and IDF (Inverse Document Frequency) methods all along with TF (Term Frequency).It gave average precision of– 0.76,– 0.8– 0.7.

C.Solutions Existing for Bengali

Kamal Sarkar proposed a technique for Bengali text summarization by making use of sentence extraction technique. The result showed an average unigram based score of 0.4122.

D.Solutions Existing for Punjabi

Visual Gupta et .al proposed an extractive automatic text summarization system for the Punjabi language. They made use of sentence feature calculations.The average accuracy of the system was found to be 81%.

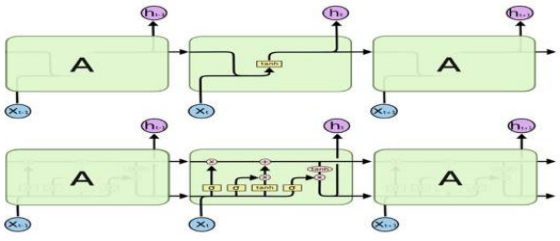

Fig.1

E.Solutions Existing for Hindi

K. Vimal Kumar et .al proposed an extractive approach to Hindi text summarization. The system is built on an algorithm by scoring the sentences based on co-occurrence of the radix of thematic words. The average accuracy was found to be 80%.

III. PROPOSED MODEL

I am proposing a model that is language independent and is working based on recurrent neural networks. The model is made of a sequence to sequence model with attention mechanism.

A. Recurrent Neural Network

A recurrent neural network (RNN) is a class of artificial neural network in which the links between units form a directed graph. Thus it to shows dynamic behavior for a given sequence. RNNs make use of their memory for processing a sequential form of input. Recurrent Neural Network is used when model requires the context to be able to provide the output based on particular input. RNN stores everything and thus are good at recollecting previous data. But in RNN, all the inputs are dependent on one another. The RNN will not forget all the relations it previously encountered while training itself. For accomplishing this, the RNN forms many layers of interconnected system with loops in them, which allows it to remember the patterns.

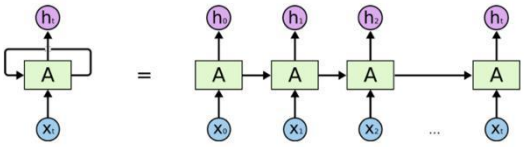

Figure 2:Unrolled RNN

B. Long Short Term Memory Cell

LSTM networks are a particular type of recurrent neural network (RNN) which is a neural network architecture typically used for modeling sequential data. The ability of LSTM cells to retain information for long periods of time gives it the added advantage over traditional neural networks.This allows for important information learned early in the sequence to have a greater impact on model decisions made at the end of the sequence.

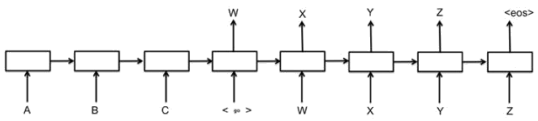

Figure 3:LSTM

A represents a full RNN cell which takes the current input word of the sequence, xi, and produces an output to the current hidden state, hi, which then passes this to the next RNN cell for the respective input sequence. While the traditional RNN cell is having a single internal layer acting on the current state and input, the LSTM cell has three.

First there is a forget gate which is responsible for controlling the information which is maintained from the previous state.

Secondly we have an update gate,the responsibility of this gate is to update the state depending on the current input.It is passed into a tanh activation layer inside of which an element-wise multiplication between these two results will be performed. Followed by this is an addition with the output and the current state.

Thirdly there is an output gate which is responsible for controling which information that goes to the next state.The words are initially given as input into an embedding lookup. Word embedding is a representation of words in the vector space. We are training the model to learn the embeddings during the course of training. These embeddings are forwarded as input into our LSTM layer.

C. Sequence To Sequence Model

A typical sequence to sequence model has two parts– an encoder and a decoder. Both the parts are two different neural network models combined into
mensional representation of it. This representation is then forwarded to a decoder network which generates a sequence of its own that represents the output.

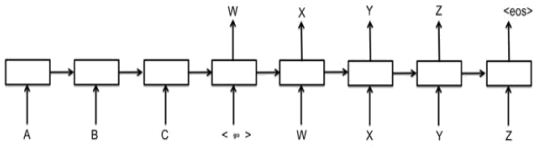

Figure 4:Sequence to Sequence model

A cell of recurrent neural network is depicted as a box in figure 3, commonly known as the LSTM cell. In the vanilla model as shown in figure 3, each input is to be encoded into a fixed-size state vector. For allowing the decoder to have better access to the input, an attention mechanism was introduced.

A multi-layer sequence-to-sequence recurrent neural network with LSTM cells and attention mechanism in the decoder is depicted in figure 4.

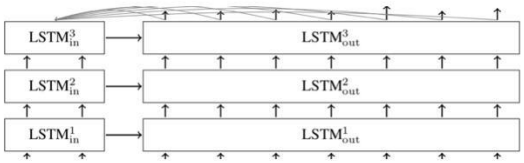

Figure 5:Sequence to Sequence Model with Attention Mechanism

In sequence-to-sequence models, the output produced by decoder at time k is fed back as the input of the decoder at time k+1. At time of testing, when decoding a sequence, this is the process followed for the sequence to be constructed. During training, on the other hand, the decoder is provided with the correct input at every time-step, even if the decoder has made a mistake earlier.

E.Bucketting and Padding

The model makes use of bucketing, which is required for handling sentences of different length effciently. During the process of summarizing there are input sentences of different lengths I, and output sentences of different lengths O.The input text to be summarized is passed as inputs for encoder, and the summary comes as output from decoder, for every pair (I, O+1) of lengths of an input text and its corresponding summary creates a sequence to sequence model. The input is padded for every sentence with a special PAD symbol.

IV. IMPLEMENTATION AND RESULT

The input text to be summarized is given to the system for summarizing.The accuracy of mAchine summarized text is evaluvated against the human written summary of the same text.

A.Experimental Setup

A 40MB of hindi articles is given as input for training. The input data is divided into 20% of it for testing and rest 80% for training.The model has 3 LSTM layers and each has 512 units.

B.Calculation Of Accuracy

BLEUScore is used as a measure to find the accuracy. BLEU, or the Bilingual Evaluation Understudy, is a score for comparing a translation of text to one or more reference translations .To calculate the BLEU score for 1-gram matches, we give a weight of 1 for 1-gram and 0 for others(1, 0, 0, 0).The product of bleuscore value with 100 gives the percentage value.The following graph is plotted based on the accuracy got on different sizes of training data.

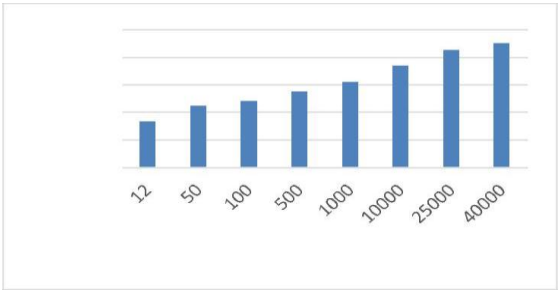

Figure 6: Graph Plotted for Accuracy

C.Result Analysis

It was observed that as the training data size increased the accuracy of summary generated also improved significantly. The model proposed gave an accuracy for 40MB of input file size.

V. CONCLUSION

The proposed architecture addresses the issue of conventional method of the neural network with encoder-decoder model which uses fixed-length context vector was found problematic when faced with long inputs sentences. The proposed model extends the vanilla encoder–decoder model by introducing a model search for a given input computed by an encoder, when producing each target word.Thus freeing the model from the overhead of having to encode a whole input sentence into a fixed length vector, and enables the model to focus only on information required for generating the next target word.

The model was tested for Hindi and Marathi language .It was observed that the model proposed outperformed any of the other existing Indian language summarizers in terms of space utilization and accuracy. One of the challenges left to handle is the bucketing of input can be made dynamic which can improve the performance further.

REFERENCES

[1] Y. Bengio, P. Simard, and P. Frasconi. *Learning long-term dependencies with gradient descent is difficult*.IEEE Transactions on Neural Networks, 5(2):157–166, 2017.

[2] K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio. *Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Arxiv* preprint arXiv:1406.1078,2017.

[3] M. Johnson et al., "*Google's multilingual neural machine translation system: Enabling zero-shot translation," Transactions of the Association for Computational Linguistics*, vol. 5, no. 20, pp. 339–351, 2016.

[4] J. Lee, K. Cho, and T. Hofmann, *"Fully character-level neural machine translation without explicit segmentation,"* Transactions of the Association for Computational Linguistics, vol. 5, pp. 365–378, 2017.

