

# Text Summarization Using Neural Networks

Asmiriti Kumari<sup>1</sup> Rupali Mittal<sup>2</sup> Anant Kaulage<sup>3</sup> Geetika Chuphal<sup>4</sup> Simran Sharma<sup>5</sup>

<sup>1,2,3,4,5</sup>Department of Computer Engineering

<sup>1,2,3,4,5</sup>Army Institute of Technology, Pune, India

**Abstract**— There are various news/articles which cannot be read completely in the hush of our daily schedules. Thus, summarization comes into picture. This paper focuses on summarizing a text using neural networks which creates a summary containing the important key points of the text/article. This summarization will be done using neural networks (word2vec model). It will focus only on English articles. The input given will be in .txt format. Thus it will make a lot easier to get a quick summary of the long articles and derive the conclusion about what is there in the articles and whether they are relevant for a user according to their interest.

**Key words:** Word2vec, Neural Network, Abstractive, Extractive, LSTM

## I. INTRODUCTION

As the amount of information on the web is increasing rapidly day by day in different format such as text, video, images. It has become difficult for individuals to find relevant information of the interest. When user queries for information on the internet he gets thousands of result documents which may not necessarily be relevant to his concern. To find appropriate information, a user needs to go through the complete documents which results in information overload problem which leads to wastage of time and efforts. To deal with this situation of dilemma, automatic text summarization plays a vital role [6]. Automatic summarization compresses a source document into meaningful content which reflects main thought in the document without altering information. Thus it helps user to grab the main notion within short time span. If the user gets effective summary it helps to understand document at a glance without checking it completely, so time and efforts could be saved. Text summarization process undergoes in three steps analysis, transformation and synthesis. Analysis step analyzes source text and select attributes. Transformation step transforms the result of analysis and finally representation of summary is done in synthesis step.

In an abstract summary, the summarized text is an interpretation of an original text. The process of producing involves rewriting the original text in a shorter version by replacing wordy concept with shorter ones[9].

## II. RELATED WORK

### A. Types of Summarization

A large document is entered into the computer and recapitulated content is returned, which is a non-redundant extract from the original passage. Automatic text summarization process model can be divided into three steps. First is the preprocessing of source text, second is interpretation of source text representation and source representation transformation to summary text representation with an algorithm and in the final step, summary text generation from summary representation [10].

Feature extraction for Wikipedia articles is done using ten different feature scores which is fed to the neural network and the neural network returns single value signifying the importance of the sentence in the summary[8].

There are two distinct types of features: non-structured features (paragraph location, offset in paragraph, number of bonus words, number of title words, etc.) and structured features (rhetorical relations between units such as cause, antithesis, condition, contrast, etc.) [2]

#### 1) Extractive Method:

Extraction is mainly concerned with judging the importance, or indicative power, of each sentence in a given document [1]. Extractive text summarization involves the selection of phrases and sentences from the source document to generate the new summary. Techniques involve ranking the relevance of phrases in order to choose only those most relevant to the meaning of the source. Extractive summarization is basically just picking up the words from the text as it is which are important and putting them in the summary. No interpretation of the text is done in this process. We also anticipate that shod sentences are unlikely to be included in summaries[3].

There are four major challenges for extractive text summarization as follows: identification of the most important pieces of information from the document, removal of irrelevant information, minimizing details, and assembling of the extracted relevant information into a compact coherent report[5].

#### 2) Abstractive Method:

Abstractive text summarization involves generating entirely new phrases and sentences to capture the meaning of the source document. This approach is commonly used by humans for getting the summary but it proves to be a challenging approach. Classical methods operate by selecting and compressing content from the source document. Abstractive summarization techniques tend to copy the process of 'paraphrasing' from a text rather than simply summarizing it. The abstractive method is more difficult and complex as compared to extractive. It copies the way human gets the summaries.

### B. Techniques of Summarization

#### 1) Bag of words:

This model is a simplified representation which is used by natural language processing and information retrieval (IR). A text which can be a sentence or a document is represented by bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. In this approach, words are tokenized which are used for each observation and frequency of each token is found.

#### 2) TF-IDF:

Tf-idf refers term frequency-inverse document frequency, and the tf-idf weight is a weight often used in information retrieval and text mining. TF-IDF weight is a statistical measure which is used to evaluate the importance of a word in a document in a collection or corpus. The importance shows proportional behaviour to the number of times a word

appears in the document but is offset by the frequency of the word in the corpus. Search engines use variations of the tf-idf weighting schemas a central tool in scoring and ranking a document's relevance given a user query. One of the simplest ranking function computation is performed by adding the every query term of tf-idf. Tf-idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification.

Typically, the tf-idf weight consists of two terms: the first computes the normalized Term Frequency (TF), aka. the number of times a occurrence of word in a document is divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF) is calculated by taking the the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

### 3) BOW & TFID Problems:

Semantic information of the text document is not stored. More focus is on uncommon words which gives undesired output. There is a chance of overfitting the model (Overfitting is the scenario when your model works properly for your dataset but fails when applied to other dataset).

## III. SYSTEM DESIGN

### A. System Architecture:

The first module in our system is user validation. Authorizing the user for giving the inputs and getting the summary through username and password. A user need to be registered before for successful validation and further proceedings else he/she needs to register first and then generate summary. The second module is uploading the input file which is in .txt format. The user may add a .txt file or paste some content in the window for taking input. Further processing is done on this source of data.

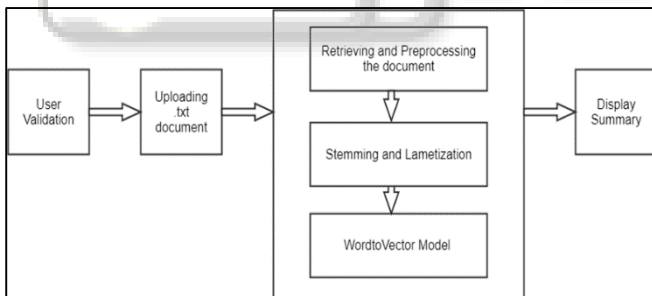


Fig. 1: System Architecture

The main module of the whole system is the processing part in which firstly pre-processing of the document is done. Preprocessing involves removing the spaces, converting all capital letters to small letters, eliminating the references and numbers etc. It brings text in a standardised format. After that stemming and lemmatization is done which involves extracting the base word from the words given in the document. This reduces the text content without making any impact on the essence of document. Finally the model used to implement the system is word2vec model.

In the last module the summary is displayed to the user.

## IV. PROPOSED SYSTEM

### A. Word2vec - The Solution

Word vectors represent a significant step towards advancing our ability to analyse relationships across words, sentences and documents. Word vectors are simply vectors of numbers that constitute the meaning of a word.

Word vectors represent words as multidimensional floating point numbers where semantically similar words are mapped to immediate points in geometric space. Put differently, words that are used in a similar context will be mapped to a approximate vector space. The charm of representing words as vectors is that they contribute themselves to mathematical operators. For example, we can add and subtract vectors—the example here is showing the power of word vectors,

### B. King—Man + Woman = Queen

In other words, we can subtract word vector of men from the word vector for king (i.e. maleness), add another word vector women or meaning (femaleness), and show that this new word vector (king—man + woman) maps most closely to the word vector for queen.

The numbers in the word vector represent the word's dispense weight across dimensions. Simply it can be said that each dimension shows a meaning and the word's numerical weight on that dimension captures the closeness of its association with and to that meaning. Thus, the semantics of the word are lodged across the dimensions of the vector [6].

The vectors which are used to represent words are called *neural word embeddings*, and representations are strange. Description of one thing gives other, with the fact that those two things are radically different. Word2vec “vectorizes” about words, and by doing so it makes natural language computer-readable – performing strong mathematical operations can be started on words to detect their similarities. So a neural word embedding represents a word with numbers.

Training of words in word2vec is done against other words that neighbor them in the input corpus. Which is performed in one of two ways, either using context to predict a target word (a method known as continuous bag of words, or CBOV), or predicting a target context using word,

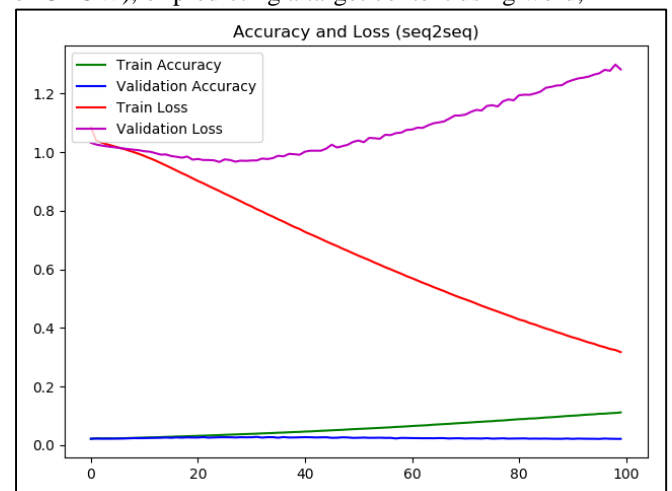


Fig. 2: Seq2Seq model Accuracy and Loss

Which is called skip-gram. We use the latter method because it produces more accurate results on large datasets [4].

If in case the feature vector assigned to a word cannot be used to accurately predict that word's context, the components of the vector are adjusted. In the corpus each word's context is the *teacher* sending error signals back to adjust the feature vector. The words vector judged similar through their context are nudged together closely by adjusting the numbers in the vector.

## V. ENCODER-DECODER ARCHITECTURE

The architecture of Encoder-Decoder is a way of organizing recurrent neural networks for sequence prediction problems that have a variable number of inputs, outputs, or both inputs and outputs.

The encoder-decoder architecture consists of two components: an encoder and a decoder.

- 1) Encoder: The encoder scans the entire input sequence and encodes it into an internal representation, often a fixed-length vector called the context vector.
- 2) Decoder: The decoder reads the encoded input sequence from the encoder and generates the output sequence [14].

Both the encoder and the decoder submodels are trained jointly, meaning at the same time.

For generating each step in the output the entire encoded input is used as context. This works fine but the fixed-length encoding of the input limits the length of output sequences that can be generated.

For exhibiting form of the encoded input sequence and allow the decoder to learn where to pay attention to the encoded input when generating each step of the output sequence an extension of the Encoder-Decoder architecture is used.

This extension of the architecture is called attention.

### A. Text Summarization Encoders

Where the complexity of the model resides is encoder as it is responsible for capturing the meaning of the source document.

Number of encoders can be used, although more commonly bidirectional recurrent neural networks, such as LSTMs, are used.

### B. Text Summarization Decoders

The decoder should be responsible in generating each word in the output sequence given two sources of information:

- 1) Context Vector: The encoder is responsible for providing the encoded representation of the source document.
- 2) Generated Sequence: The summary of word or sequence of words already generated.

As in the simple Encoder-Decoder architecture the context vector may be a fixed-length encoding, or may be a more expressive form filtered via an attention mechanism

Little preparation is provided to the generated sequence, with, such as distributed representation of each generated word via a word embedding.

One by one words are generated, this requires that the model be run until some maximum number of summary words are generated or a special end-of-sequence token is reached. The process must be started by imparting the model

with a special start-of-sequence token in order to generate the first word.

### C. Long Short Term Memory

Long Short Term Memory (LSTM) networks are defined as a recurrent neural network that can be used with STS neural networks. Long Short Term Memory (LSTM) are similar to Gated Recurrent units (GRU) but have an extra memory state buffer and an extra gate which gives them more parameters and hence a longer training time.

- a) The encoder is made up of:
  - 1) Input Layer: Takes the English sentence and pass it to the embedding layer [12].
  - 2) Embedding Layer is responsible of taking the English sentence and convert each word to fixed size vector
  - 3) First LSTM Layer: every time step this layer takes a vector that represents a word and pass its output to the next layer, we used CuDNN LSTM layer instead of LSTM because it's much much faster [12].
  - 4) Second LSTM Layer: Its working is same as the previous layer, but instead of passing its output, it passes its states to the first LSTM layer of the decoder [12].
- b) The decoder is made up of:
  - 1) Input Layer: Takes the French sentence and pass it to the embedding layer [12].
  - 2) Embedding Layer: Takes the French sentence as input and convert each word to fixed size vector.
  - 3) First LSTM Layer: Every time step, it accepts a vector that represents a word as input and pass its output to the next layer, but here in the decoder, we initialize the state of this layer to be the last state of the last LSTM layer from the decoder [12].
  - 4) Second LSTM Layer: Processing the output from the previous layer and passes its output to a dense layer [12].
  - 5) Dense Layer (Output Layer): Takes the output from the previous layer and outputs a one hot vector representing the target French word [12].

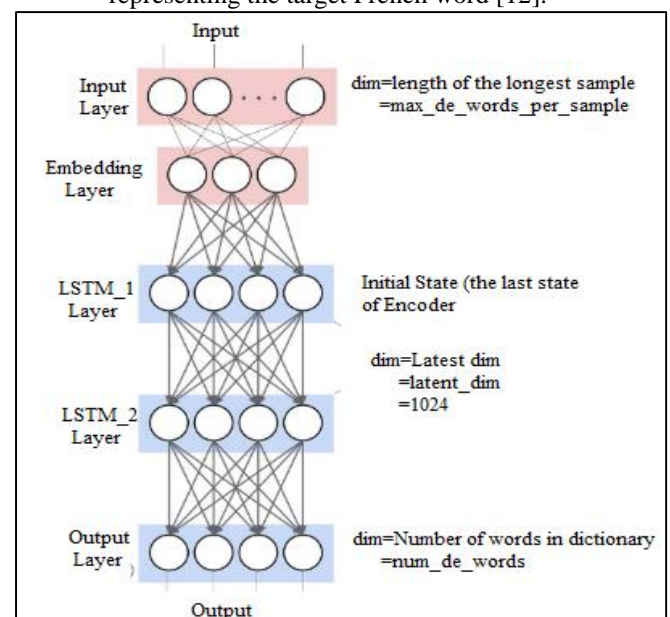


Fig. 3: The Decoder

## VI. MATHEMATICAL MODEL

Suppose input vector is X with vocabulary  $x_1$  to  $x_k$  and  $y$  is the output vector. So, to calculate the error E, we need to define the error function or loss function, and we will use the L2 loss function.

Loss functions want to achieve their minimized value while learning from a dataset and the L2 function also wants to minimize the squared differences between the estimated and existing target values[7].

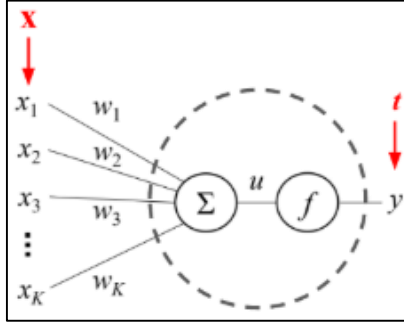


Fig. 4: A Neuron

### A. Single neuron at the time of training

So, when we are trying to calculate an L2 loss function of a single neuron, we will use the following equation

$$E = (1/2)(t - y)^2 \quad (1.1)$$

Here,  $t$  is the target vector value,  $y$  is the estimated vector value or predicted vector value, and  $E$  is the error function. We have defined our L2 error function.

To minimize this function value for accurately predicting target value we will take partial derivative of the L2 function equation with respect to  $y$ . The procedure of deriving derivation and then by using this derivation trying to minimize error values, is called gradient descent.

$$\delta E / \delta y = y - t \quad (1.2)$$

We know that output  $y$  is dependent on  $f(u)$  and  $f(u)$  is dependent on input weight values  $w_i$ . So we need to apply chain rules and generate the error function value. If we use partial derivative chain rules, then we will get the following equation, which will be useful in word2vec.

Result of the partial derivative chain rule

$$\delta E / \delta u = \delta E / \delta y \cdot \delta y / \delta u$$

$$= (y - t) y (1 - y)$$

$$\delta E / \delta w_i = \delta E / \delta y \cdot \delta y / \delta u \cdot \delta u / \delta w_i$$

$$= (y - t) \cdot y (1 - y) \cdot x_i$$

After calculating the L2 loss function as per the value of the error, the neural network input weight will be updated and this kind of iteration will continue until we achieve the minimum error value or error rate.

Mathematical calculation performed by neural networks

$$E = (1/2) \sum_{j=1}^M (y_j - t_j)^2$$

$$\delta E / \delta y_j = y_j - t_j$$

$$\delta E / \delta u_j^i = \delta E / \delta y_j \cdot \delta y_j / \delta u_j^i$$

$$\delta E / \delta w_{ij}^i = \delta E / \delta u_j^i \cdot \delta u_j^i / \delta w_{ij}^i$$

$$\delta E / \delta h_i = \sum_{j=1}^M \delta E / \delta u_j^i \cdot \delta u_j^i / \delta h_i$$

$$\delta E / \delta u_i = \delta E / \delta h_i \cdot \delta h_i / \delta u_i$$

start with the output index, backpropagate the error to the hidden layer, and update the weight. In the fifth equation, you can see that we need to calculate the error function for the output layer so that backpropagate the error and we can update the weight of the input layer[7].

### B. Mathematics behind the word2vec model

The word2vec neural network is using a one-hot encoded word vector as input and then it passes this vector value to the next layer, which is the hidden layer, and this is nothing but the weighted sum values that feed into the hidden layer as input. The last output layer generates the vector value, but to make sense of the output, we will convert the vector into probability format, and with the help of softmax techniques, we will also convert the output word vector into probability format [7].

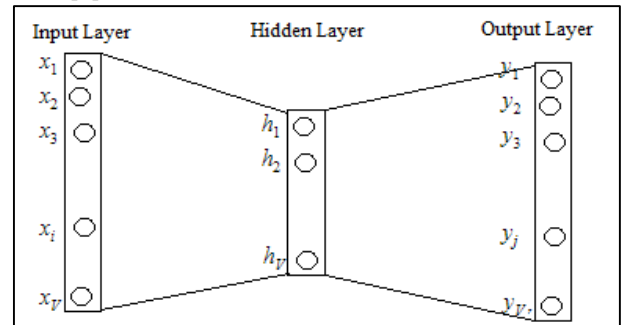


Fig. 5: Layers

$$h = x^t W = v_{w1}$$

$$u_j = v_{wj}^T \cdot h$$

$$p(w_j | w_1) = y_j = \exp(u_j) / \sum_{j'=1}^v \exp(u_{j'})$$

In the first equation, we can see the weighted sum of the input word vector and weights and get  $h$  in the second equation. We multiply  $h$  and the weighted sum of word vectors of the hidden layer  $v_{wj}$ . Here, weight and index has been changed. This multiplication is  $u_j$ . Here,  $u_j$  is the activation function. We will then generate probability by using the value of  $u_j$ . So, the final equation is the softmax function. Let's simplify the equation by replacing  $u_j$  in the third equation with the input and output word vector format, then you will get the final equation.

$$p(w_j | w_1) = \exp(v_{w0}^T v_{w1}) / \sum_{j'=1}^v \exp(v_{wj'}^T v_{w1})$$

Final probability equation of the word2vec model

This time, our output is a softmax function, so for updating weight using backpropagation, we need to define the loss function. So, here, we will define the loss function in the form of the softmax function, so we will use minus log probability of the softmax function and then we will perform gradient descent.

## VII. CONCLUSION

The performance of the text summarization process depends dominantly on the style of the human reader. The selection of features as well as the selection of summary sentences from the training paragraphs by the human reader play an important role in the performance of the network. The

network if trained according to the style of the human reader and to which sentences the human reader deems to be important in a paragraph. Infact this approach provides advantage to greater extent.

#### VIII. FUTURE WORK

Filling up forms, given text containing the necessary details. Creating a bio-data, from textual details of the person. It is very important to build a multilingual summarization system and this research could be a stepping stone towards achieving that goal provided there is availability of online lexical databases in other languages.

#### REFERENCES

- [1] Khosrow Kaikhah, Automatic Text Summarization with Neural Networks, June 2004.
- [2] W. T. Chuang and J. Ymg "Extracting sentence segments for text summarization: a machine learning approach", Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, pp.152-159, 2000.
- [3] J. Kupiec, J. Pedenon and F. Cheb "A Trainable Document Summarizer", Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, pp. 68-73, 1997.
- [4] <https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2vec/>.
- [5] J. L. Neto, A. A. Freitas, C. A. A. Kaestner, "Automatic Text Summarization using a Machine Learning Approach," in Proc. Brazilian Symposium on Artificial Intelligence (SBIA 2002), Lecture Notes in Computer Science, Vol. 2507. Springer, 2002, pp. 205-215.
- [6] [machinelearningmastery.com/gentle-introduction-text-summarization](https://machinelearningmastery.com/gentle-introduction-text-summarization/).
- [7] [https://subscription.packtpub.com/book/big\\_data\\_and\\_business\\_intelligence/9781787121423/6/ch06lv11sec48/understanding-the-logic-of-the-word2vec-model?fbclid=IwAR3bR\\_5W1ZJIjyUstB3m8nKX7P\\_EY5N\\_W6cAmFneksvTP3akMwz2J2t9prg](https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781787121423/6/ch06lv11sec48/understanding-the-logic-of-the-word2vec-model?fbclid=IwAR3bR_5W1ZJIjyUstB3m8nKX7P_EY5N_W6cAmFneksvTP3akMwz2J2t9prg).
- [8] Dharmendra Hingu, Deep Shah, Sandeep S Udmale, "Automatic Text Summarization of Wikipedia Articles", IEEE, 2015.
- [9] Waleed al-sanie, "Towards an infrastructure for Arabic text summarization using theoretical structure theory", Master Thesis, Department of computer science. King Saud university, Riyadh, Kingdom of Saudi Arabia, 2005.
- [10] S. Akter, A.S. Asa, Md. P. Uddin, "An extractive text summarization technique for Bengali document(s) using K-means clustering algorithm," in Proc. IEEE International conference on Imaging, Vision & Pattern Recognition, 2017.
- [11] S. Gholamrezazadeh, M. Amini, B. Gholamzadeh, "A Comprehensive survey on Text Summarization Systems", IEEE.
- [12] <https://towardsdatascience.com/nlp-sequence-to-sequence-networks-part-2-seq2seq-model-encoderdecoder-model-6c22e29fd7e1>.
- [13] [http://www.jackdermody.net/brightwire/article/Sequence\\_to\\_Sequence\\_with\\_LSTM](http://www.jackdermody.net/brightwire/article/Sequence_to_Sequence_with_LSTM).
- [14] <https://machinelearningmastery.com/encoder-decoder-models-text-summarization-keras/>.