Abstractive Text Summarization using Machine Learning

Aditya Dingare
Department of Computer Science
California State University, Fullerton
Fullerton, USA
adityadingare@csu.fullerton.edu

Doina Bein
Department of Computer Science
California State University, Fullerton
Fullerton, USA
dbein@fullerton.edu

Wolfgang Bein
Department of Computer Science
University of Nevada, Las Vegas
Las Vegas, USA
wolfgang.bein@unlv.edu

Abhishek Verma
Department of Computer Science
California State University, Northridge
Northridge, USA
abhishek.verma@csun.edu

***Abstract*— Text summarization creates a brief and succinct summary of the original text. The summarized text highlights the main text's most interesting points without omitting crucial details. There is a plethora of applications on the market that include news summaries, such as Inshort and Blinklist which not only save time but also effort. The method of manually summarizing a text can be time-consuming. Fortunately, using algorithms, the mechanism can be automated. We apply three text summarization algorithms on the Amazon Product Review dataset from Kaggle [23]: extractive text summarization using NLTK, extractive text summarization using TextRank, and abstractive text summarization using Seq-to-Seq.**

Keywords—Machine learning, extractive text summarization, abstractive text summarization.

I. Introduction

There are various forms of summaries: single document, multi document, informative summary, and query focused summary. The type of input provided to an algorithm determines these types, so for a multi-document summary, multiple documents are used. The input for query based is focused on a specific query outcome. There are two output-based primary methods for summarizing the text: abstractive text summarization and extractive text summarization.

In extractive text summarization, the summarized text is part of the original text as the algorithm extracts the most relevant words and sentences from the original text. For example, in Fig. 1 the output text is consisting of all the words from the original input text only.

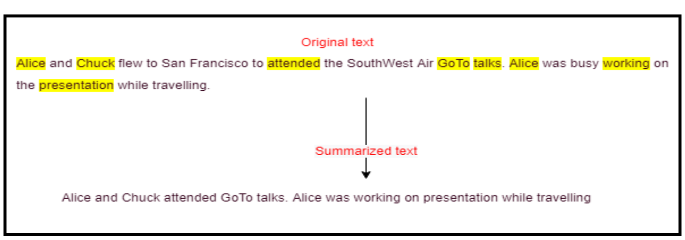

Figure 1. Extractive text summarization

Abstractive text summarization is opposite to extractive text summarization [3] as it returns the summary of the text that may consists of new word and sentence that are not part of the original text. For example, in Fig. 2, the output of the abstractive summarization consists of the words that are not part of the original text. Hybrid text summarization uses both abstractive and extractive text summarization techniques together. (S. Selvarani, 2014)

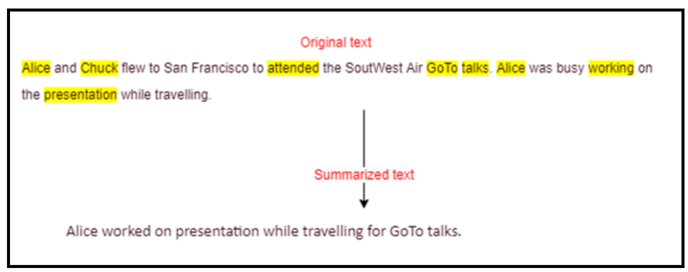

Figure 2. Abstractive text summarization

While abstractive text summarization produces more substantive summarized text than extractive text summarization, it is more difficult to implement. Most of the research focuses on extractive text summarization's implementation and limitations.

We apply three text summarization algorithms on the Amazon Product Review dataset from Kaggle [23]: extractive text summarization using NLTK, extractive text summarization using TextRank, and abstractive text summarization using Seq-to-Seq. We compare their performances over various product reviews.

The paper is organized as follows. In Section II we present the background and related work, followed by the description of the algorithms used in Section III. Simulations are results are presented in Section IV. Concluding remarks and future work are given in Section V.

II. Background and Related Work

Term frequency, latent frequency, and graphical extractive algorithms are the three primary types of extractive algorithms. The sentence that has a similar appearance to the document word has a high score in terms of frequency. The sentences are sorted first in latent variable, and the sentence with the closest representation of the latent variable is chosen [4]. A similarity matrix is constructed in the graphical process, and the TextRank algorithm is executed based on it.

The extractive text summarization can be divided into three main categories (see Fig. 3): term frequency, latent variable, and graphical. Term frequency and sum basic algorithm are similar: commonly occurring sentences are added together [6]. Latent variable algorithm works like as discussed above. In embedding page rank algorithm, the embedding page vector is calculated and used during the algorithm execution [9].

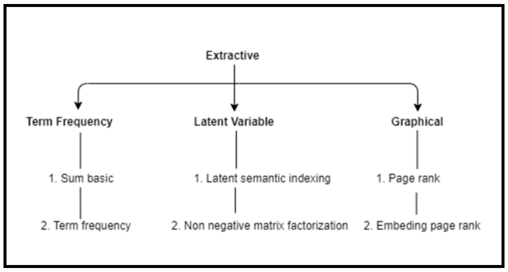

Figure 3. Algorithms to implement extractive text summarization

Abstractive text summarization can be divided into two categories (see Fig. 4): semantics based, and structure based. The implementation discussed in this project is from the semantics graph-based technique.

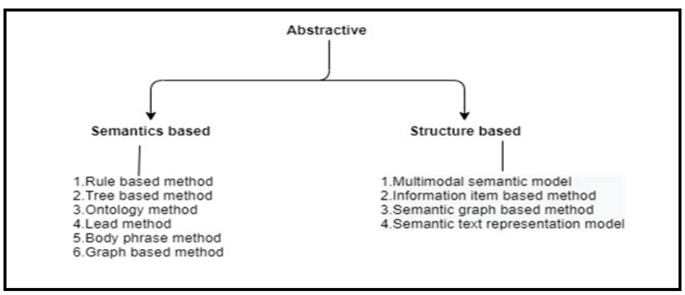

Figure 4. Algorithms to implement abstractive text summarization

We use Text Rank algorithm developed by (F. Hai-jian, 2011). The TextRank algorithm is like PageRank PageRank algorithm that was designed and developed by Google, but instead of web pages, it uses sentences. The similarity between two sentences is the likelihood of a web page switch. This score of similarity is stored in a square matrix [11]. The standard steps in the TextRank algorithm are to load input data and construct vectors for sentences using GloVe word embeddings. The next stage is text preprocessing, which involves cleaning the data and removing common terms such as am, an, the, for, and so on. We then construct a vector representation of sentences and a similarity matrix and next we apply TextRank algorithm.

The abstractive text summarization technique using Sequence-to-Sequence modeling (Seq2Seq model) is used to summarize the text. The standard implementation involves usage of encoder and decoder as referenced. The encoder and decoder are configured into two phases training and inference phase. The encoder reads input data and extracts the contextual information present in the input sequence using the Long Short-Term Memory model (LSTM). The decoder, on the other hand, uses the encoder's output as an input and is equipped to predict the next word in the series [17]. The input data used by TextRank algorithm is a single document and does not support the use of RNNs and LSTM (Callahan, 2018). It is difficult for encoder to memorize the huge data size as the fixed length vector is used to store the input data. In addition, the encoder uses a unidirectional LSTM. The context cannot be captured in both directions using a unidirectional LSTM. The bidirectional LSTM, combined with global attention for the previous problem, can be used to address the LSTM issue [1][2].

III. Implemented Text Summarization Algorithms

The main steps in implementation are data gathering, data cleaning, and algorithm implementation. We implement three algorithms and compare their results.

For data gathering we used Amazon Product Review dataset from Kaggle [23] that has approximately 568,455 rows and 10 columns, almost 300 MB in size. Out of 10 columns, the ProductReviewText column has detailed description of the product available on the Amazon and is mainly used by the extractive text summarization for the TextRank algorithm.

Data cleaning involves contraction mapping for handling the words with short forms like "ain't", "don't" as "do not" etc., and changing the input data into either lower or upper case, remove parenthesis, eliminate stops words (e.g. “is”, “and”,” are”), punctuations, and special characters like @, # etc. These two steps are common to both abstractive and extractive algorithms [16].

The abstractive text summarization uses two columns mainly ProductReviewHeader and ProductReviewText. The column ProductReviewHeader is nothing but the header line of that particular review. This column is either one or two lines of short headline of the review.

1. Extractive text summarization using NLTK

We used the Natural Language Toolkit (NLTK) library for statistical language processing which include tokenization, calculating frequency of words, and calculating weighted frequency of words. Term frequency can be used to identify the keywords. Extraction of keywords in reviews enable customers in determining whether a product review is necessary and whether or not to continue reading it. Following the calculation of word frequency, a weighted frequency of each sentence in the input data column, ProductReviewText, is calculated. We calculated the frequency and weighted frequency of each word that is present in the review text. The weighted frequency can be calculated in other way by dividing the words frequency by frequency of the word that is mostly occurred [22]. The next step is to add weighted frequencies and sort the sum of weighted frequencies in descending order. The sentence with the maximum sum of weighted frequency is extracted as the summarized text. Based on the weighted frequency, the summary of the original text is returned.

2. Extractive text summarization using TextRank

The TextRank algorithm depends on PageRank algorithm. The probability among two words in the sentences is calculated. For the TextRank algorithm, the input reviews data will be subdivided into text units such as keywords, key phrases and the graph model is built. In this implementation, we used an undirected weighted graph; each node represents the sentence in the review text and the edges represents the relationship between them calculated using the formula [21]:

$$WS\ (V_{i)} = (1\text{- } d) + d^{*}\ \Sigma_{\ Vj \in In(Vi\ )}\ eji\ /\ \ \Sigma_{\ Vj \in Out\ (Vi\ )\ eJk}\ WS_{(Vij)}$$

Each sentence is treated as a node in the text. There is an undirected right edge between the nodes corresponding to the two sentences if two sentences are identical. The following formula can be used to check sentence similarity [21].

$$Similarity\ (S_{i,}\ S_{j}) = |\{W_{k}|\ W_{k\ \in}\ S_{i}\ \&\ W_{k\ \in}\ S_{j}\ \}|\ /\ \log(|S_{i}\ |) + \log(|S_{j}\ |)$$

where Si and Sj are two sentences of our product review where as Wk represents word in the sentence.

The steps are:

1. Split the given text review T into complete sentences
2. Clean the data by deleting stop words, nouns, and verbs from the input data for each sentence
3. Build a candidate keyword graph G = (V, E), where V is a node collection of sentences, and then draw an edge between any two points if and only if these two sentences are linked.
4. Calculate the weight repetitively using the formula mentioned below.
5. The node weights are sorted in reverse order to obtain the most relevant T terms as candidate keywords.
6. The most significant T words are extracted from #5, marked in the original text, and then combined into a keyword if adjacent phrases are created. [19]

The product reviews in column ProductReviewText is divided into small chunks of sentences only if it contains long sentences.

3. Abstractive text summarization using Seq-to-Seq

The encoder and decoder are needed for extractive text summarization using the Seq-to-Seq model. In this implementation, the Long Short-Term Memory (LSTM) is used as encoders and decoders to catch the phrase dependencies in a sentence's words. To implement encode and decoder," Recurrent Neural Networks i.e. RNN" can also be used. The encoders and decoders are designed further in two stages, namely training and inference. The encoder receives the input data as input and extracts the contextual information present in the data. The timestamp is also important factor here. So, for each time stamp, each word from the product review sentence is given to the encoder which retrieves the contextual information from the product review. This is the part of the training phase. Before feeding the target sequence into the decoder, their start and end are inserted [22].

The encoder training phase is single direction (see Fig. 5).

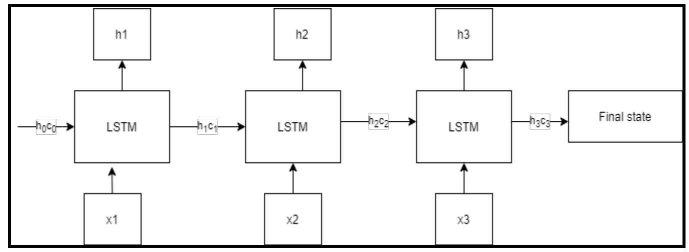

Figure 5. Training encoder

The encoder receives the input word by word at each time interval and each steps of the LSTM passes the output to the next cell in the LSTM.

The decoder receives the hidden state (h_i) and cell state (c_i) from the previous steps as data. The decoder training is single direction (see Fig. 6) This decoder cell also takes input from the encoder and process each word.

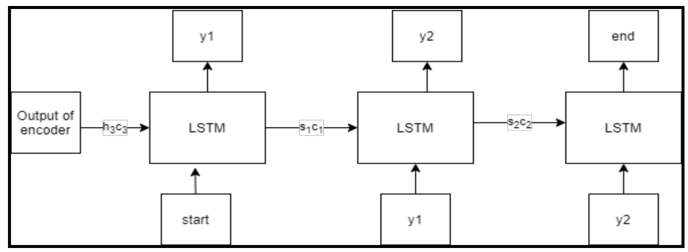

Figure 6. Training decoder [3]

There are total of three cells of each encoder and decoder.

To calculate the past as well as future context for each product review sequence we used a bi-directional LSTM.

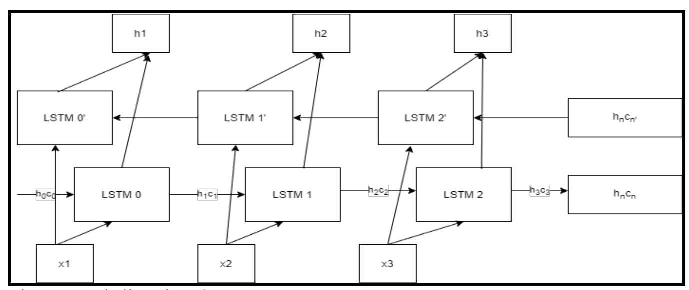

Figure 7. Bi-directional LSTM

The bidirectional LSTM in which the original cells as well as it's transpose are used. The single-directional LSTM has the disadvantage of being unable to predict future background information and learning all the input sequence sentences. As referenced to the figure 16, in bi-directional LSTM the output of the last cell is given back to additional LSTM. LSTM0', LSTM1', and LSTM2' are all the transpose of original LSTM0, LSTM1, and LSTM2 [22].

IV. Simulations and Results

For both the abstractive and extractive text summarization we considered reviews that have at least 250 words. The output of the extractive text summarization using NLTK applied to a sample product review and its German language translation of the summary (cross language summary) is shown in Fig. 8.

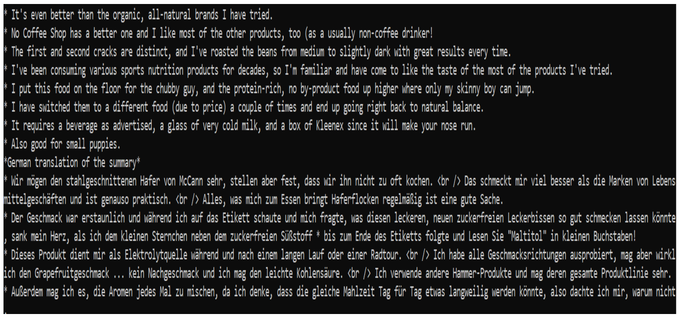

Figure 8. NLTK Output

The output by the TextRank algorithm of the sample product review is shown in Fig. 9. There are approximately more than 5 reviews being summarized and their associated German translations.

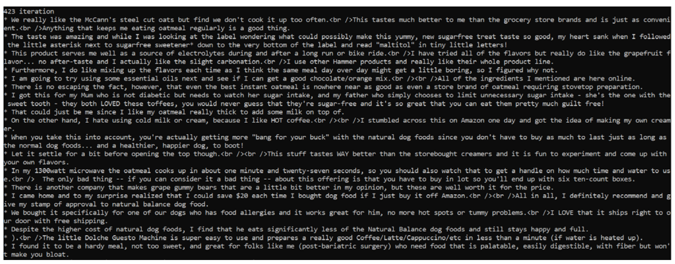

Figure 9. TextRank Output

Fig. 10 shows the output of the Seq-to-Seq implementation with original text as well as summarized text.

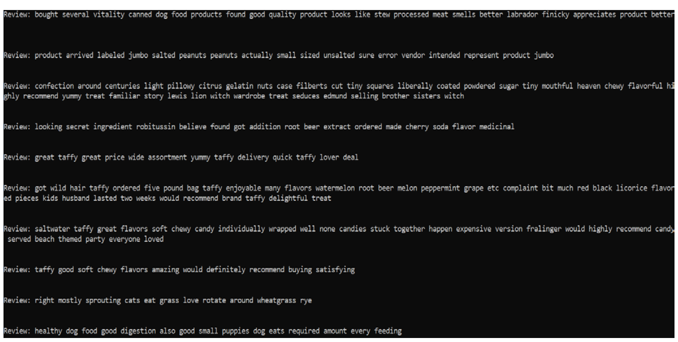

Figure 10. Seq-to-Seq output

The following limitations have been noted. The extractive text summarization implemented using TextRank algorithm does not return proper output for the duplicate words and sentences. The algorithm is modified to perform the multi document text summarization. It means the algorithm checks for the .CSV files in input directory and picks up all the file while processing. However, both the TextRank and Seq-to-Seq algorithms do not remember the input data from one source input file while processing another input file. In short, even though the multi document processing is supported, the inter dependency of input is not taken into consideration. Another drawback of our implementation is that the models we developed are unable to generate new product feedback that could be used in conjunction with summarizing the subsequent input data. The main disadvantage of using Seq-to-Seq for abstractive text summarization is that sentences are not ranked like text summarization. This will cause us to skip over text that appears frequently in the input.

V. Conclusion and Future Work

In this paper we apply three text summarization algorithms on the Amazon Product Review dataset from Kaggle [23]: extractive text summarization using NLTK, extractive text summarization using TextRank, and abstractive text summarization using Seq-to-Seq. There are advantages and disadvantages to using these algorithms for product reviews summarization.

As future work, we note that the TextRank algorithm we used for extractive text summarization does not endorse Recurrent Neural Networks (RNN). The RNN is the most common algorithm for dealing with a continuous stream of data. Internal memory in the RNN aids in remembering the input and makes it ideal for deep learning problems. We could use RN to improve the current algorithm processing because the RNN remembers the import part of the input and uses it as a guide in subsequent runs. The RNN's performance summarized text looks more like text summarized by a person. However, the RNN has its own disadvantage - It fails for complex model. Its output is also poor if the input text includes duplicate words and sentences. For both the abstractive and extractive text summarization, using certain user parameters, the output of the summarized text may be further refined.

Acknowledgment

This research was sponsored by the NATO Science for Peace and Security Program under grant SPS MYP G5700.

References

[1] Sequence to Sequence Learning with Neural Networks arXiv:1409.3215v3 [cs.CL] 14 Dec 2014

[2] Get To The Point: Summarization with Pointer-Generator Networks. https://arxiv.org/abs/1704.04368

[3] https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summarization-using-deep-learning-python/

[4] https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/

[5] https://www.analyticsvidhya.com/blog/2020/12/understanding-text-classification-in-nlp-with-movie-review-example-example/

[6] Abstractive Text Summarization Using Transformers | by Rohan Jagtap | The Startup | Medium. https://medium.com/swlh/abstractive-text-summarization-using-transformers-3e774cc42453

[7] A Gentle Introduction to Text Summarization in Machine Learning. https://blog.floydhub.com/gentle-introduction-to-text-summarization-in-machine-learning/

[8] 8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L. and Polosukhin, I., 2021. Attention Is All You Need. [online] arXiv.org. Available at: https://arxiv.org/abs/1706.03762. [Accessed 17 May 2021].

[9] D. Suleiman and A. Awajan, "Deep Learning Based Abstractive Text Summarization: Approaches, Datasets, Evaluation Measures, and Challenges", 2021

[10] Pranab Ghosh. "Six Unsupervised Extractive Text Summarization Techniques Side by Side". [Online]. Available:

https://pkghosh.wordpress.com/2019/06/27/six-unsupervised-extractive-text-summarization-techniques-side-by-side/. [Accessed: 17- May-2021].

[11] N. Patel and N. Mangaokar, "Abstractive vs Extractive Text Summarization (Output based approach) - A Comparative Study," 2020 IEEE International Conference for Innovation in Technology (INOCON), 2020, pp. 1-6, doi: 10.1109/INOCON50539.2020.9298416.

[12] J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), 2019, pp. 1-3, doi: 10.1109/IconDSC.2019.8817040.

[13] C. Lakshmi Devasena and M. Hemalatha, "Automatic Text categorization and summarization using rule reduction," IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012), 2012, pp. 594-598.

[14] R. Mishra, V. K. Panchal and P. Kumar, "Extractive Text Summarization - An effective approach to extract information from Text," 2019 International Conference on contemporary Computing and Informatics (IC3I), 2019, pp. 252-255, doi: 10.1109/IC3I46837.2019.9055636.

[15] Meena S M, Ramkumar M P, Emil Selvan G SR, "Text Summarization Using Text Frequency Ranking Sentence Prediction," 2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP), 2020, pp. 1-5, doi: 10.1109/ICCCSP49186.2020.9315203.

[16] S. R. Manalu, "Stop words in review summarization using TextRank," 2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2017, pp. 846-849, doi: 10.1109/ECTICon.2017.8096371.

[17] A. Rahman, F. M. Rafiq, R. Saha, R. Rafian and H. Arif, "Bengali Text Summarization using TextRank, Fuzzy C-Means and Aggregate Scoring methods," 2019 IEEE Region 10 Symposium (TENSYMP), 2019, pp. 331-336, doi: 10.1109/TENSYMP46218.2019.8971039.

[18] X. You, "Automatic Summarization and Keyword Extraction from Web Page or Text File," 2019 IEEE 2nd International Conference on Computer and Communication Engineering Technology (CCET), 2019, pp. 154-158, doi: 10.1109/CCET48361.2019.8989315.

[19] M. R. Ramadhan, S. N. Endah and A. B. J. Mantau, "Implementation of Textrank Algorithm in Product Review Summarization," 2020 4th International Conference on Informatics and Computational Sciences (ICICoS), 2020, pp. 1-5, doi: 10.1109/ICICoS51170.2020.9299005.

[20] Y. Chen and Q. Song, "News Text Summarization Method based on BART-TextRank Model," 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), 2021, pp. 2005-2010, doi: 10.1109/IAEAC50856.2021.9390683.

[21] T. Behere, A. Vaidya, A. Birhade, K. Shinde, P. Deshpande and S. Jahirabadkar, "Text Summarization and Classification of Conversation Data between Service Chatbot and Customer," 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 2020, pp. 833-838, doi: 10.1109/WorldS450073.2020.9210289.

[22] W. Jiang, Y. Zou, T. Zhao, Q. Zhang and Y. Ma, "A Hierarchical Bidirectional LSTM Sequence Model for Extractive Text Summarization in Electric Power Systems," 2020 13th International Symposium on Computational Intelligence and Design (ISCID), 2020, pp. 290-294, doi: 10.1109/ISCID51228.2020.00071.

[23] Consumer Reviews of Amazon Products. https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products