Hindi Document Extractive Summarization: Neural Method on a New Data Set

Kunal Tawatia, Nishant Jain, Suman Kundu
Department of Computer Science and Engineering
Indian Institute of Technology Jodhpur, Jodhpur - 342037, India
Email: https://sumankundu.info

***Abstract*—Extractive summarization is one of the vital tasks in text analysis and natural language processing. Although Hindi is one of the world's highly speaking languages and produces thousands of online documents daily, most existing text summarization works focus on the English language. A Neural network-based summarizer is popular for abstractive summarization but has not been explored for extractive one except in a few recent studies. The present work uses a neural extractive summarizing model to develop a Hindi language extractive summarizer. The main contribution of the paper is two-fold. First, we generated a new Hindi-based text summarization data set from a popular Hindi news channel AajTak. The code to generate the data set is available at https://tinyurl.com/sonaa-hindi-text. Then we use this data set to train a Neural Extractive Summarization model. The model also learns the word embeddings while learning itself. The ROUGE-2-F1 and ROUGE-1-F1 results on test data show promising output with a score of 20.02 and 39.81, respectively.**

I. Introduction

Extractive document summarization is one of the important research problems in text analysis and natural language understanding. Extractive summarization is to extract out sentences from a document. It is similar to the highlight important lines in a document. It has been in research for a long time [1], [2], [3], [4], [5], [6] and most of the approaches can be categorized under the head of Semantic Analysis-based Approaches [4], [7], [6], Graph-based Approaches [5], [3], [8], [9], [10], Meta-heuristic based approach [11], [12], [13], [14]. Abstractive document summarization [15], on the other hand, generates a summary by paraphrasing the important information of the document. Although neural network or deep learning-based algorithms have been extensively used for abstractive summarization, recent research shows its application in successfully extracting of summary out of single document [6], [16], [17].

English documents are substantially explored in the literature of text summarization. There are a few works on other European languages [18] as well. No study of neural network-based text extraction was conducted for Indian languages such as Hindi. However, online documents generated per day for Hindi is on a huge scale, considering hundreds of Hindi newspapers and news channels produce thousands of online documents. It provides a multitude of text-based applications and research opportunities. A comprehensive comparative analysis of different extractive text summarization techniques on Hindi alongside English text was presented in [19]. The study concluded that the neural network-based methods perform poorly on Hindi documents. However, the data set used therein was minimal, about 500 documents, which for a deep learning model is not convincing.

In this work, we generated a Hindi document dataset with the full article and editorial summarization by scrapping one of a popular Hindi news channel AajTak. We then used this data set to train a *Neural Network* based text extractor model. Our model is motivated by the document summarization solution proposed in [17] wherein modifications are made in order to use the model for the Hindi language. In addition to that we initialized the word embeddings using a Gaussian distribution with the Xavier scheme and allowed our model to optimize them to make the model a true end-to-end solution. We compared our results in terms of ROUGE-2-F1 and ROUGE-1-F1 [20] scores with the baseline algorithms and got a score of 20.01 and 39.81, respectively. Our methodology achieved more than 16% improvement from the best baseline algorithm *LEAD-3* algorithm. The contribution can be summarized as follows:

- We generated a Hindi Text Summarization data set of 100,000 documents from one popular Hindi news channel. The code to generate a similar data set will be published for future research.
- A neural extractive summarization methodology has been developed to obtained extractive summaries from Hindi documents using the aforementioned data set. To our best knowledge, this is one of the first attempts to develop a Hindi Neural Summarizer.

The rest of the paper is organized as follows. Motivation and literature review on Hindi text summarization is presented in Section II, followed by the description of the proposed Neural Summarization methodology for Hindi extractive summary in Section III. Section IV reports the results of the experiments, and finally Section V concludes the article.

II. Motivation and Related Work

Hindi is one of India's highly speaking languages, and the third-highest [21] speaking language in the world. It provides ample opportunity for text and speech-based research. However, the literature on Hindi text summarization is much less as compared to English. Recent research attempt for Hindi text summarization are majorly Meta-heuristic based [22], [23], [24] except a few. For example, [25] proposed a graph-based algorithm for Hindi text summarization, whereas sentence scoring based on the occurrence of radix terms and thematic words has been proposed in [26]. There has not been any

independent attempt to use the Neural Network based model for Hindi text analysis. It motivates us to work on this research study.

In the same line of thought, a comprehensive review work was conducted recently in [19], where the authors systematically reviewed the performance of several different English text Summarization algorithms for the Hindi data set. The attempt to use Neural Network based algorithm in this research work was limited. The paper mentions that 500 documents were used in the research, but it is not clear from the paper whether these 500 were tested on a model trained with English or the training was conducted over Hindi files. The number of documents used for training was also not specified. On the other hand, the work concludes that "the neural networks-based techniques have not exhibited a good performance in comparison to the graph-based and meta-heuristic-based techniques for Hindi". It also motivates us to see how neural network-based algorithm performs to extract from Hindi documents if the model is trained with Hindi documents.

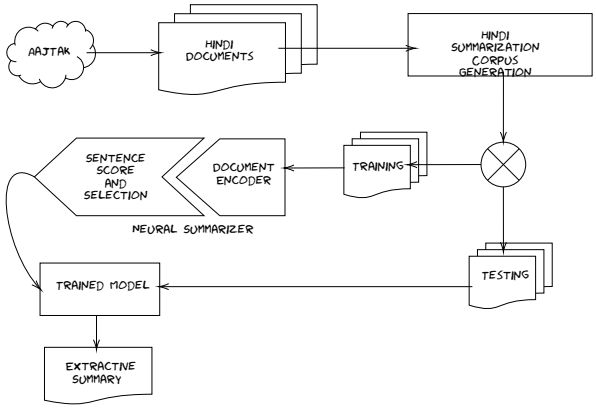

Fig. 1: Block Diagram of NeuSumHD

III. NeuSumHD: Neural Extractive Summarization of Hindi Document

This Section provides detailed descriptions of the methodology used for the proposed NeuSumHD for generating text summary. Figure 1 shows a block diagram of the overall methodology used in this research work. We first generated a Hindi Summarization Corpus of 100,000 documents from AajTak web articles. The data is available at https://doi.org/10.6084/m9.figshare.13712881.v1 for download. The data set is then divided into training and testing. The neural summarization model is divided into two parts viz. (i) document encoding and (ii) sentence selection as described in the following Sections.

A. Hindi Extractive Summarization Corpus Generation

We generated a Hindi corpus of 100,000 news articles as there is no useful data set is available for Hindi extractive summarization. The article's body and highlights are scraped from the AajTak website. AajTak is one of the popular Hindi news channels having a very active web portal of Hindi news


न बहुमत-न वापसी, 19 साल में कोई दल नहीं बदल पाया झारखंड का इतिहास

झारखंड विधानसभा चुनाव में जेएमएम-कांग्रेस-आरजेडी गठबंधन ने भले ही बहुमत हासिल कर लिया हो, लेकिन इस बार भी कोई पार्टी अपने दम पर बहुमत हासिल करने में नाकाम रही. यही नहीं, झारखंड में 19 साल के राजनीतिक इतिहास में किसी भी सत्ताधारी पार्टी सत्ता में वापसी नहीं करने का रिकॉर्ड भी कायम रहा. इस बार भी रघुवर दास के नेतृत्व वाली बीजेपी सरकार की सत्ता से विदाई हो गई.

- **झारखंड में किसी भी एकल पार्टी को बहुमत नहीं मिला**
- **झारखंड में सत्ताधारी पार्टी की सत्ता में वापसी नहीं**

झारखंड की राजनीति इतनी कॉम्प्लेक्स है कि इस राज्य के गठन को 19 साल हुए और अब तक 10 सीएम भी बन चुके हैं. झारखंड में अब तक रघुवर दास पहले मुख्यमंत्री हैं, जिन्होंने भले ही पांच साल का कार्यकाल पूरा किया. लेकिन वह अपनी सरकार की दूसरी बार सत्ता में वापसी नहीं करा सके.

Fig. 2: A sample article used in AajTak dataset which has a *description* in grey text which is followed by bold bullet points, *Highlights*.

articles. For reference to the reader, Figure 2 shows examples of news articles from AajTak. The AajTak article comes with curated editorial summarization. The AajTak articles contain bullet highlights as well as a paragraph of editorial summary. We have taken three sentences as a reference summary. The challenges were to find the 3-sentences as not all article contains three bullet highlights. In such cases, we took the first n-sentences from the summary paragraph to compensate for the remaining. One of the other challenges with this curated summarization is that the editors' summary highlights are not exact sentences from the article but a novel abstract summary. We use the Algorithm 1 to generate the extractive reference summaries for the training and analysis purpose. The process is described next.

1) Data Prepocessing and Extractive Reference Summaries: The raw data scraped is split into words, and then these words are processed into tokens. This process of tokenization involves classifying entities into punctuation and actual words. There are several challenges as different editors may have different ways of representing the same things; for example, there are several tokens just for the quotation that are used differently by various editors. The data scrapped is in UTF-8 encoding, due to Hindi characters and different punctuation and end of sentence tokens are in Unicode. Documents are accordingly tokenized. As already mentioned, the editors' summary highlights are not exact sentences from the article but a novel abstract summary. We label the target summary as a combination of a subset of sentences appearing in the document, which maximizes the reference abstract summary. The extent of overlap is judged by the ROUGE-2-F1 metrics.

Unlike [17], where a paragraph of the original article is considered as a sentence, we split the data further into "sentences". We did this to optimize the output as it is expected to have human-readable sentences. It would then have one disadvantage that the selected sentence can be from anywhere in a particular paragraph, and it might not make sense individually without the context of the paragraph. Nevertheless, this is a common disadvantage generally faced in extractive text summarization techniques. We break paragraphs into sentences at the occurrence of several end tokens and keep quotations as

it is for maximum context coverage. Since document lengths have drastically increased, there is an explosion in the number of combinations for any given length, and the greedy approach used in [17] may take much time just to label the data. We did not maximize the ROUGE-2-F1 score for all possible combinations to handle this issue; rather, we fixed the number of sentences to a threshold θ for our summary and proposed a greedy hill-climbing algorithm for labeling as in Algorithm 1. The algorithm starts with an empty set as a summary. It selects a sentence that maximizes the ROUGE-2-F1 with the editorial summary. This partial summary is then incremented with appending sentences that maximize the ROUGE-2-F1 until we get the θ sentences as a labeled extractive summary.

Algorithm 1 Extractive Summary Labeling

Input: doc, S $\{doc$: collection of sentences, S: editorial summary$\}$
Output: S_l $\{S_l$: labeled summary$\}$
Initialize: $S_l = \emptyset$
while $|S_l| < \theta$ **do**
  $s = \arg\max_s$ ROUGE-2-F1$(S_l \cup \{s\}, S)$ $\forall s \in doc$
  $S_l = S_l \cup \{s\}$
end while

Final data set properties are listed in the Table I. The statistics are shown in Figure 3 for reference.

TABLE I: Properties of Generated Hindi Document Corpus

Properties	Value
No. of Documents	100,000
Mean Document Length	15.65 sentences
Mean Sentence Length	19.64 words
Mean ROUGE-2-F1 of labeled summary	35.73

B. Model

Extractive summarization algorithms are mainly reduced into two subroutines, which can be understood to be: sentence scoring and sentence selection. Sentence scoring intends to represent the importance of each sentence appearing in the document, while sentence selection corresponds to the method employed to sequentially pick those sentences such that the generated summary has the maximum coverage while being minimally redundant.

Our scoring and selection model is divided into two sections: document encoding and sentence selection. The evaluation metrics used in the model are the ROUGE-2-F1 metric. The ROUGE-2-F1 score gives the extent of overlap between predicted and reference summary. The model is briefly described in the following Section.

1) Document Encoding: A document constitutes several sentences, and each sentence is formed out of words. Thus this model employs a hierarchical document encoder, as shown in Figure 4, to represent the sentences in the input document. This encoder is intrinsically a combination of a sentence-level encoder and a document-level encoder.

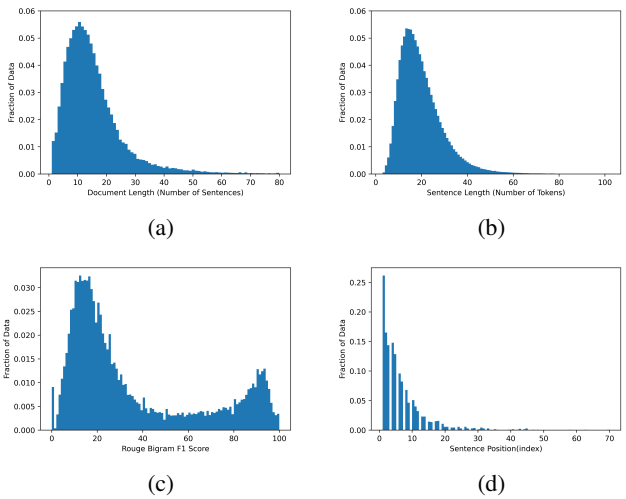

Fig. 3: (a) Distribution curve of document lengths (in sentences). (b) Distribution curve of sentence lengths (in words). (c) The ROUGE-2-F1 score distribution of the labeled target summary. (d) Distribution curve of the positions of sentences selected in the target summary.

The sentence-level encoder employs a bidirectional GRU (BiGRU) [27] which reads the word embeddings of a sentence in both directions (i.e., *left* $\rightarrow$ *right*, and *right* $\rightarrow$ *left*) and then its sentence-level representation is constructed by concatenating the last hidden layer from both the directions. Here the GRU recurrent unit is defined as:

$$z_i = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (1)$$
$$r_t = \sigma(W_r x_r + U_r h_{t-1} + b_r) \quad (2)$$
$$\hat{h}_t = \phi_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \quad (3)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \quad (4)$$

where W_z, W_r, U_z, U_r are the parameter matrices and b_z, b_r are the biases; x_t is the input word embeddings corresponds to the sentences in the document. h_t represents the hidden states and the initial states of the BiGRU is set to zero vector in the experiments.

The document-level encoder again employs a BiGRU, which reads the sentence-level representation of each sentence sequentially in both the direction and then forms document-level representation by concatenating both the forward hidden layer and backward hidden layer after input of every sentence. These document-level representations are considered actual sentence representations and then fed to the scorer and selector network.

2) Sentence Scoring and Selection: This Section of the model has the responsibility to remember the partial output summary and then scoring the remaining sentences according to their importance and redundancy with the existing summary. The model employs another GRU, shown in Figure 5, for this task that remembers the selected sentences in summary,

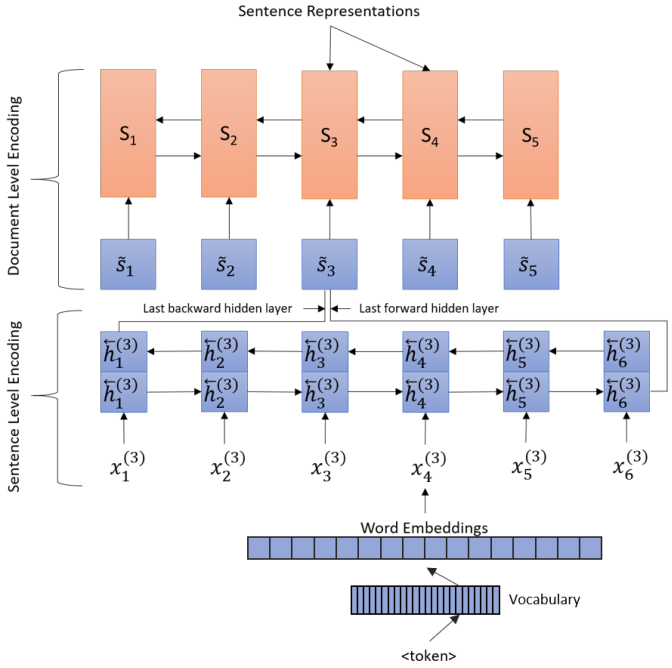

Fig. 4: Document Encoder

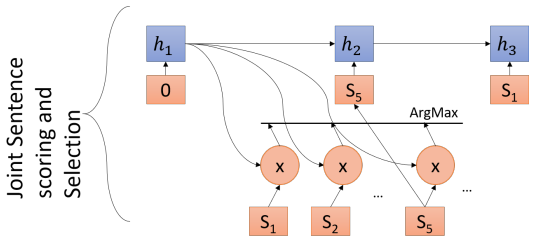

Fig. 5: Sentence Scoring and Selection

and then uses a two-layer Multi-Layer Perceptron (MLP), which takes the current hidden layer of GRU as one input and a sentence representation as another and gives the score of that sentence as output. The sentence representation with the maximum score is selected to be the part of the summary, and its sentence representation is then fed back to the GRU as the next input. This process continues until we have a summary of the desired number of sentences.

3) Evaluation metrics and Optimizer: The model optimizes Kullback-Leibler (KL) divergence of the predicted score distribution P and the labeled training data Q at every time step t. The sentence scores generated by the previous Section are normalized into a probability distribution P with softmax function, while labeled training data Q is the normalized ROUGE-2-F1 gain achieved by each sentence when coupled with the partial output summary. Thus we train our model to minimize the loss function:

$$J = D_{KL}(P \parallel Q) = -\sum_{s} P_t(s) \log(\frac{Q_t(s)}{P_t(s)}) \qquad (5)$$

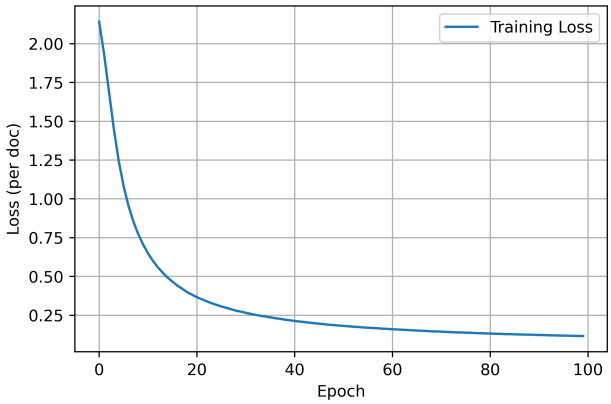

Fig. 6: Training loss curve

IV. Experiments and Results

The experiment has been conducted with the prepared AajTak dataset on Google Co-lab using PyTorch. All the experiments were run on Tesla P100 with CUDA version 10.1. Our AajTak dataset has 188816 unique words. We then stem [28] these words and constructed our vocabulary. We slash our vocabulary at a size of 100,000 words since we get 99.64% coverage over the dataset. Besides, the words ignored have a frequency of 2 or less and insignificant for training the model. We converted the ignored words into `<unk>` token. The dimensions of the parameters used in the program are shown in Table II. We kept the batch size 64 documents and ran 100 epochs for our experiment.

TABLE II: Parameters of the Experiments

Parameters	Value
Word embeddings size	50
Sentence-level encoder size	256
Dropout at sentence-level	0.3
Document-level encoder size	256
Dropout at document-level	0.2
Sentence extractor size	256

Initialization of parameters, the configuration of optimizing algorithm, gradient clipping, and all other model configurations are also kept to be the PyTorch's default values. We initialized the word embeddings using a Gaussian distribution with the Xavier scheme and allowed our model to optimize them. A GPU compatible code is used with several batch processing techniques to speed up the training process. The loss curve of the training is reported in Figure 6.

The data set is divided into 90%, 5%, and 5% chunks for training, validation, and testing. Length of the summary is kept to 3 in our experiments and we compare it with the LEAD-3 and other baseline methods.

A. Baseline Algorithms

We compare our results with the following baseline algorithms.

- *Random*: In this algorithm, uniformly randomly selected sentences are considered as an extractive summary. We selected 3 random sentences from the article as we considered summary length to be 3 for the proposed method in our experiment.
- *TextRank*: TextRank [3] is a graph-based text summarization algorithm. This is an unsupervised algorithm for extracting text summaries.
- *LEAD2*: In this method, the first two sentences of the document is considered a summary.
- *LEAD3*: Similar to LEAD2, in this method, 3 sentences are taken as summary. We particularly included this because in our experiment we generated 3 sentence summaries.

For reference, we also mentioned the scores with the *Oracle* data, which is the score of overlap with the reference Summary with the editorial summary. One can consider this as a theoretical maximum possible to achieve in our data set.

TABLE III: Comparative results of the proposed methods with other baseline methods

Models	ROUGE-2-F1	ROUGE-1-F1
Oracle	35.76	53.22
Random	9.67	31.19
TextRank	12.26	31.38
LEAD2	15.70	36.76
LEAD3	17.16	38.51
NeuSumHD	20.02	39.81

B. Results

The comparative results of the experiment are shown in Table III. The proposed NeuSumHD achieved a score of 20.02 ROUGE-2-F1 and 39.81 ROUGH-1-F1 scores on the Hindi *AajTak* dataset. The result is clearly better than that of the strongest baseline algorithm we experimented with. In terms of absolute value, the score is 2.86 and 1.30 greater than *LEAD-3* in terms of ROUGE-2-F1 and ROUGE-1-F1, respectively. That is, this improves about 16.67% and 3.37% over *LEAD-3*. A graphical representation of the comparative results are shown in Figure 7.

Besides, we analyzed the position of the output sentences. Figure 8 shows the distribution of the sentence position in the document. The x-axis shows the index of the sentence in the document, and the y-axis shows the fraction of the test data appear in the corresponding positions. As expected, the lower indexed sentences are in a higher number in the results.

The percentage overlapped with the end result and the expected result is shown in Table IV. For over 32% of the documents, the predicted results match exactly the expected/labeled summary, and for 51.5% of cases, the overlap is more than or equal to 50%.

V. Conclusion

The paper presented a neural network-based text summarization for Hindi documents. We call it NeuSumHD. NeuSumHD is found to outperform TextRank and LEAD-3 baseline algorithms. The paper also provides a mechanism to generate Hindi Datasets from Hindi news channel AajTak. The paper generated a dataset of size 100,000 documents based on the proposed methodology.

Although our methodology is seen to outperform the baseline algorithm and Graph-based TextRank algorithm for AajTak dataset, different other datasets can be explored as a future work. In addition, it would be interesting to see how the methodology can be modified to perform multidocument summarization of Hindi texts.

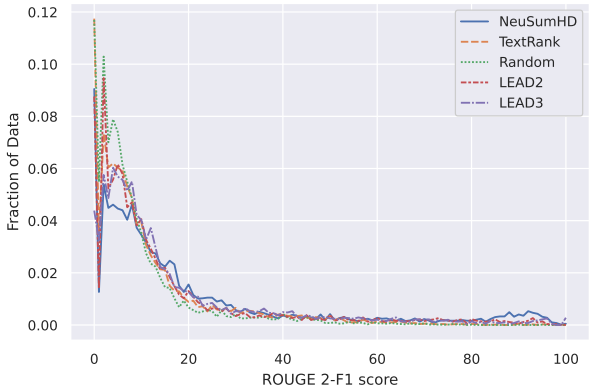

Fig. 7: Distribution curve of ROUGE-2-F1 scores achieved by various models

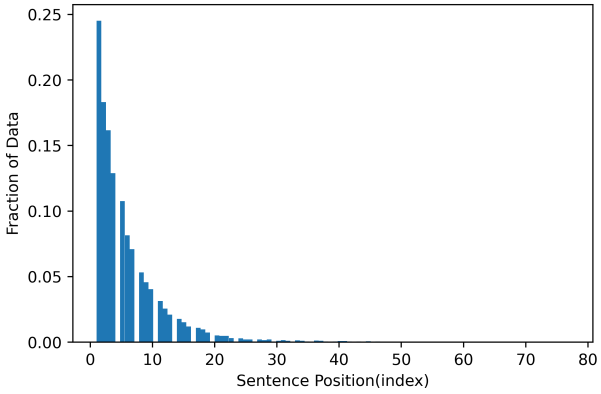

Fig. 8: Distribution curve of positions of selected sentences by NeuSumHD

TABLE IV: Percentage of overlap between expected and predicted extractive summary.

Overlap	Fraction of data
0%	38.22%
33%	9.15%
50%	12.41%
66%	6.40%
100%	32.69%

REFERENCES

[1] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159–165, apr 1958.

[2] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the ACM (JACM)*, vol. 16, no. 2, pp. 264–285, 1969.

[3] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, 2004, pp. 404—-411.

[4] J. Steinberger and K. Jezek, "Using Latent Semantic Analysis in Bioinformatics," in *Proc. of the International Conference on Information System Implementation and Modeling (ISIM'04)*, no. June, 2004, pp. 93–100.

[5] G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *Journal of Artificial Intelligence Research*, vol. 22, pp. 457–479, 2004.

[6] J. Cheng and M. Lapata, "Neural summarization by extracting sentences and words," in *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, vol. 1. Association for Computational Linguistics (ACL), 2016, pp. 484–494.

[7] H. Kobayashi, M. Noguchi, and T. Yatsuka, "Summarization based on embedding distributions," in *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (ACL), 2015, pp. 1984–1989.

[8] K. Sankar and L. Sobha, "An approach to text summarization," in *Proceedings of CLIAWS3, Third International Cross Lingual Information Access Workshop*. Boulder, Colorado: Association for Computational Linguistics, 2009, pp. 53–60.

[9] X. Wan, "Towards a Unified Approach to Simultaneous Single-Document and Multi-Document Summarizations," in *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China: Coling 2010 Organizing Committee, 2010, pp. 1137–1145.

[10] D. Parveen, H. M. Ramsl, and M. Strube, "Topical coherence for graph-based extractive summarization," in *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (ACL), 2015, pp. 1949–1954.

[11] E. Shareghi and L. S. Hassanabadi, "Text summarization with harmony search algorithm-based sentence extraction," in *5th International Conference on Soft Computing as Transdisciplinary Science and Technology, CSTST '08 - Proceedings*. New York, New York, USA: ACM Press, 2008, pp. 226–231.

[12] R. M. Alguliev, R. M. Aliguliyev, and N. R. Isazade, "Multiple documents summarization based on evolutionary optimization algorithm," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1675–1689, apr 2013.

[13] R. Rautray and R. C. Balabantaray, "Cat swarm optimization based evolutionary framework for multi document summarization," *Physica A: Statistical Mechanics and its Applications*, vol. 477, pp. 174–186, jul 2017.

[14] P. Verma and H. Om, "A variable dimension optimization approach for text summarization," in *Advances in Intelligent Systems and Computing*, vol. 741. Springer Verlag, 2019, pp. 687–696.

[15] S. Gupta and S. K. Gupta, "Abstractive summarization: An overview of the state of the art," pp. 49–65, may 2019.

[16] R. Nallapati, F. Zhai, and B. Zhou, "SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents," in *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, San Francisco, California, 2017, pp. 3075–3081.

[17] Q. Zhou, N. Yang, F. Wei, S. Huang, M. Zhou, and T. Zhao, "Neural document summarization by jointly learning to score and select sentences," in *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, vol. 1, 2018, pp. 654–663.

[18] M. Suppa and J. Adamec, "A Summarization Dataset of Slovak News Articles," in *Proc. of The 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 2020, pp. 6725—-6730.

[19] P. Verma, S. Pal, and H. Om, "A comparative analysis on Hindi and English extractive text summarization," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 18, no. 3, p. 39, 2019.

[20] C.-Y. Lin and E. Hovy, "Automatic Evaluation of Summaries Using N-Gram Co-Occurrence Statistics," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*. Edmonton, Canada: Association for Computational Linguistics, 2003, pp. 71–78.

[21] Wikipedia, "List of languages by total number of speakers - Wikipedia," 2020. [Online]. Available: https://en.wikipedia.org/wiki/List_of_languages_by_total_number_of_speakers

[22] A. N. Gulati and S. D. Sawarkar, "A novel technique for multidocument Hindi text summarization," in *2017 International Conference on Nascent Technologies in Engineering, ICNTE 2017 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., jun 2017.

[23] V. Dalal and L. Malik, "Semantic graph based automatic text summarization for hindi documents using particle swarm optimization," in *Smart Innovation, Systems and Technologies*, vol. 84. Springer Science and Business Media Deutschland GmbH, 2018, pp. 284–289.

[24] P. B. Bafna and J. R. Saini, "Hindi Multi-document Word Cloud based Summarization through Unsupervised Learning," in *International Conference on Emerging Trends in Engineering and Technology, ICETET*, vol. 2019-Novem. IEEE Computer Society, nov 2019.

[25] K. V. Kumar, D. Yadav, and A. Sharma, "Graph based technique for hindi text summarization," in *Advances in Intelligent Systems and Computing*, vol. 339. Springer Verlag, 2015, pp. 301–310.

[26] K. Vimal Kumar and D. Yadav, "An improvised extractive approach to hindi text summarization," in *Advances in Intelligent Systems and Computing*, vol. 339. Springer Verlag, 2015, pp. 291–300.

[27] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014, pp. 1724–1734.

[28] L. Gomes, "Hindi Stemmer," 2010. [Online]. Available: https://research.variancia.com/hindi_stemmer/