Text Summarization for Indian Languages: Extractive summarization using graph based technique

Ansh Vora[1], Aastha Shah[1], Rinit Jain[1] and Sheetal Sonawane[1]

[1]*Pune Institute of Computer Technology, Pune, India*

Abstract

The ILSUM shared task focuses on text summarization for Indic Languages. Given the enormous amount of data being generated in the day to day life, extracting important information from large amount of documents is a very tedious and time consuming task. It has become essentially important to develop an Automatic Text Summarization System(ATS) that can successfully extract and retrieve concise information from the documents and generate an accurate summary. In this task, we experiment with unsupervised method of summarization by enriching the vector embeddings of TextRank Algorithm with the help of the integration of dynamic damping factor Latent Dirichlet Allocation (LDA). The accuracy and precision of the proposed model is measured using ROUGE metrics.

Keywords

Summarization, Extractive, Latent Dirichlet Allocation, Dynamic Damping Factor

1. Introduction

There is a gigantic amount of data being generated on the internet in recent times. The amount of textual content generated in the form of legal documents, scripts, news articles, etc., is very vast. Generating concise summaries of such vast textual sources manually is a very time-consuming and tedious task to do. Hence, this gives rise to various approaches that help in generating automatic summaries through ATS (Automatic Text Summarization) systems. These ATS systems help to extract important information from documents. The main aim of text summarization is to shorten the length of the source text while preserving all important sentences and the overall meaning of the source text. Text summarization is divided into two types, mainly single-document summarization and multi-document summarization as mentioned in Paper [1] and Paper [2].
There are three types of approaches that can be followed for generating ATS systems as mentioned in Paper [3]. These approaches Extractive Summarization, Abstractive Summarization and Hybrid Summarization. The extractive method aims to extract the most important sentences from the input document. Abstractive method on the other hand aims at generating sentences that differ from the original sentences in the input text and generate summary that coveys the entire meaning of the document. The above mentioned paper also explains the different methodologies that can be followed to model the ATS systems. Extractive ATS systems can be modelled by various methods like Statistical based, Concept based, Graph based, Semantic based, etc methods. Abstractive ATS systems are modelled by using Graph based, Tree based, Ontology based, etc methods including BERT summarization which outperforms TextRank summarization as seen in Paper [4].

Extractive summarizers are less complex and they preserve the fidelity of the original document. They are not prone to errors related to text generation and misinterpretation as they directly extract the highest-ranked sentences. Paper [5] gives a survey of extractive graph-based summarization and states how it outperforms other approaches. Paper [6] is a graph-based method for extractive text summarization that aligns with PageRank's ranking algorithm [7]. The advantage of the TextRank algorithm is that it is an unsupervised method and does not require any training to generate the summary. Along with being unsupervised, TextRank is also a language-independent method that

Forum for Information Retrieval Evaluation, December 12-15, 2024, India
anshvora5@gmail.com (A. Vora); aasthass17014@gmail.com (A. Shah); rinitjain9@gmail.com (R. Jain); ssonawane@pict.edu (S. Sonawane)

generates a summary based on the occurrences of words. Aligning with this graph-based method of text summarization, our proposed model extends the TextRank algorithm by integrating LDA topic modeling (Paper [8] and Paper [9]) and a dynamic damping factor as parameters to enhance the vector embeddings of the nodes of the graph and shows the results obtained by our model using various evaluation metrics like ROUGE-N, ROUGE-L, and ROUGE-S that evaluate the scores based on measures like recall, precision, and F1 score.

2. Literature Survey

Given the large corpus of data being generated daily, it has become increasingly important to develop automatic text summarization models that can produce precise and concise summaries. Paper [6] initially proposed a graph-based ranking algorithm, TextRank, which uses an unsupervised method for sentence extraction. This technique is an extension of Google's renowned PageRank algorithm Paper [7]. One key advantage of this state-of-the-art method is its language independence, making it easily adaptable to different domains, genres, and languages. Building on this, Paper [10] made further advancements by analyzing 75 articles extracted from the Medium site. In their study, they combined a similarity matrix generated through BM25+ with the original TextRank, normalizing the matrix by dividing it by the maximum similarity score in a given matrix. Their primary focus was to enhance the mean F-score for all ROUGE metrics compared to both the original TextRank algorithm and BM25+. Their results showed an improvement of 1.654% and 0.413%, respectively.

Paper [9] suggested that LDA is an unsupervised generative probabilistic model for corpus modeling. They introduced various LDA parameters such as Gibbs sampling, Expectation-Maximization (EM), and Variational Bayes inference (VB). Paper [11] evaluated the extractive summarization method and the efficacy of LDA using Latent Semantic Analysis (LSA) and ROUGE metrics to assess its accuracy and reliability. Paper [12] focuses on enhancing extractive text summarization by using topic-based sentence representation. Paper [13] used lexical centrality to determine sentence salience within a document.

Paper [14] proposed DeepChannel, an attention-based neural network that achieved robust results using only 1 % of the CNN/DailyMail dataset. Paper [15] and Paper [16] introduced CNN based approaches to achieve high accuracy against benchmark datasets. Paper [17] adressed the challenges of such black box deep learning models. Paper [18] used the Webis-EditorialSum-2020 dataset, containing 1,330 summaries for 266 articles. Their approach involved two models for generating extractive summaries. The first was TextRank, which employed two similarity functions: lexical overlap and common named entities. The second model used BERT-based embeddings and clustered segments using K-Means, with two variants: ExtSum-XLNet and ExtSum-DistilBERT Paper [19]. Paper [20] contributed to the field by employing the SumSurvey dataset, a long-document abstractive summarization dataset composed of scientific survey papers. Well-known evaluation metrics such as ROUGE and BERT Score were used as reference-based measures in extractive summarization. Paper [21] and Paper [22] focuses on the benchmark dataset creation and implementation for summarization in Hindi, Gujarati, and Indian English. Paper [23] and paper [24] highlights approaches using pre-trained models with extended dataset including language Bengali. These papers also discuss about the second task about detection of factual incorrectness. Paper [25] and Paper [26] introduces three Dravidian languages and also extends the misinformation detection subtask to a cross-lingual setup. Paper [27] proposes an approach using large language models (LLMs) datasets for misinformation detection by generating factually incorrect summaries of trusted news articles.

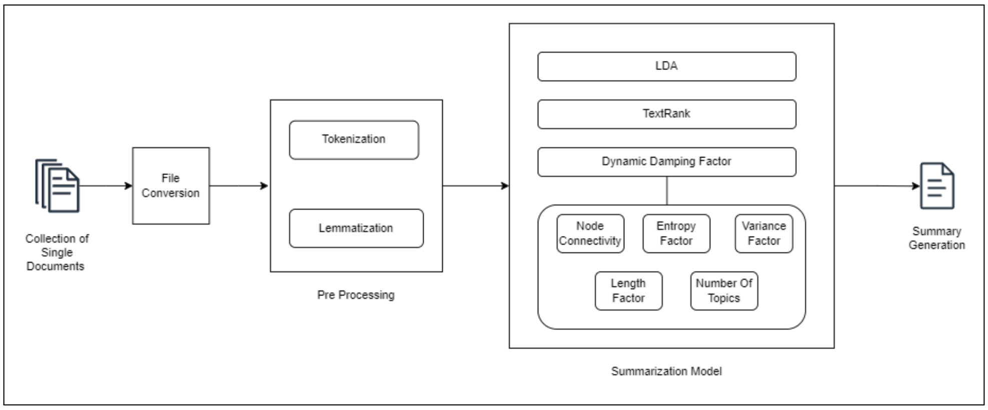


Figure 1: System Architecture Diagram

3. Methodology

3.1. Data Preprocessing

Data preprocessing is the first step in this summarization model and is used to make the documents relevant and targetable using the machine learning algorithms. We import the dataset of article-reference summary pairs and carry out the pre-processing phase on the text corpus of the article. The following steps are taken in the Data Preprocessing Phase as referred in Figure**1**:

1. **Sentence Tokenization** – The first step of processing involves sentence tokenization of the text corpus. The text corpus comprises of multiple sentences that form the article. LDA and TextRank work at a sentence level by creating summaries using the top ranked sentences.
2. **Word Tokenization** – Each of these individual sentences is then broken down into words. This increases the amount of analysis that can be carried out, such as using TextRank to calculate word overlaps and using LDA to create a document-term matrix where each topic is represented by a number of frequently co-occurring words. Thus each sentence is broken down into tokens and punctuations are removed while expanding the contractions.
3. **Removal of Stopwords** – Stopwords are simple English words used in the article for coherence and fluency. They do not contribute significantly to the importance and meaning of a sentence. Stopwords include words like ‘is’, ’in’, ’this’, and so on.

3.2. LDA for Topic Modelling

LDA is a generative probabilistic statistical model that aims to discover latent topics within a document. It assigns a set of topics to each sentence based on the words it contains, where each topic is identified by a group of words that occur frequently together. LDA is used to find the topical distribution of the document and identifies the topics or themes encapsulated by the text by looking for groups of words that frequently occur together across sentences and modelling them into a topic.

After preprocessing of the raw text is performed, LDA modelling is used to convert it into a numerical representation known as a document-term matrix that captures the frequency of words in every sentence. The columns of the document-term matrix correspond to unique words, and the rows correspond to sentences.

LDA Modelling calculates the word-topic distributions based on how likely a word is to belong to a topic given the other words in the sentence and determines how much each topic contributes to the overall content of a sentence as well as the importance of the sentence in the entire corpus of text.

The LDA model creates a Topic Matrix that maps sentences to the topics. Each row corresponds to a sentence, and each column corresponds to a topic. Thus, each sentence is mapped to various topics it encapsulates. The values in the matrix represent the probability that the sentence belongs to a particular topic.

Sentences that have higher topic scores are more likely to be about the key topics of the document and thus are more likely to be included in the final summary. More weight is given to sentences that are central to the main ideas or topics, improving the quality and coherence of the generated summary.

3.3. TextRank

TextRank is a graph-based algorithm used for extractive summarization of text. This sentence-based approach computes the similarity between sentences based on word overlap. In the TextRank algorithm, each sentence is represented as a node in a graph, and the edges between the nodes represent the similarity score between the sentences. A similarity matrix stores the similarity scores between all sentences and is used for summarization.

The similarity score between two sentences, A and B, is calculated as:

$$\text{Similarity}(A, B) = \frac{|A \cap B|}{\log(\text{len}(A) + 1) + \log(\text{len}(B) + 1)}$$

Once the similarity matrix is computed, the PageRank algorithm is applied to rank sentences according to their centrality, based on the assumption that the more overlaps a sentence has, the greater its importance. The algorithm updates the rank (importance) of each sentence based on its connections to other sentences.

The PageRank algorithm is expressed as:

$$PR(i) = \frac{1 - \alpha}{N} + \alpha \sum_{j} W_{ji} \cdot PR_t(j)$$

Where α is the damping factor, ideally set to 0.85.

After ranking the sentences, we selected the length of the summary to be generated in terms of the number of sentences (n), where the top-ranked n sentences are joined to form the required summary.

3.4. Damping Factor

In the PageRank formula, α is the damping factor used to control how the algorithm navigates through the connected nodes. With probability α, the rank of a sentence is determined by its similarity to other sentences (i.e., the structure of the graph), and with probability $1 - \alpha$, the algorithm allows random jumps to any other sentence, preventing rank accumulation from being too dependent on just a few influential nodes. With a damping factor of 0.85, the algorithm jumps to a connected node with 85% probability but, in 15% of cases, makes random jumps to avoid a thread of sentences monopolizing the summary.

A higher damping factor gives more weight to sentence connectivity, ensuring sentence similarity is emphasized. A lower damping factor places more emphasis on random jumps, making the system less reliant on sentence similarity. Hardcoding a value of 0.85 may lead to inefficient summarization in articles with lesser overlaps due to the diversified nature of content. In this model, we dynamically calculate the value for the damping factor using the following approach:

- **Node Connectivity** – A higher node connectivity factor means the sentences are very similar, positively contributing to the damping factor. Higher node connectivity indicates more coherence in the text corpus.

$$\text{Connectivity Factor} = \frac{\text{node connectivity factor}}{5}$$

- **Entropy Factor** – Entropy measures the distribution of topics throughout the document. High entropy indicates diverse topical spread and reduced similarity scores, suggesting a need to lower the damping factor.

$$\text{Entropy Adjust} = \frac{\text{entropy factor}}{5}$$

- **Variance Factor** – The variance of the integrated matrix represents values from the Topic Matrix formed by LDA modeling and the similarity score. A higher value denotes significant differences in sentence scores, indicating that some sentences are much more important than others, thus increasing the damping factor.

$$\text{Variance Factor} = \frac{\text{lda variance} + \text{textrank variance}}{2}$$

- **Length Factor** – This is based on the length of a document. If a document is too long, then more randomness in sentence selection is efficient. This factor has a lower weight and is inversely proportional to the damping factor.

$$\text{Length Factor} = \frac{1}{\text{length of document}}$$

- **Number of Topics** – This measures the diversity in the article. A higher number of topics typically correlates with reduced similarity scores due to topical spread. Thus, an increase in the number of topics should reduce the damping factor.

$$\text{Topic Factor} = \frac{50}{\text{Num topics}}$$

Thus, we calculate the dynamic damping factor as:

$$\textbf{Damping Factor} = 0.5 + 0.25 \times \text{Cf} - 0.35 \times \text{EA} - 0.2 \times \text{Lf} - 0.3 \times \text{Tf} + 0.15 \times \text{Vf}$$

Where:

- Cf is Connectivity factor
- EA is Entropy Adjust
- Lf is Length factor
- Tf is Topic factor
- Vf is Variance factor

Note: Weights for each factor can be calculated by iterating over from 0 to 1 with a 0.05 increment and choosing the best-performing weights. They may vary from dataset to dataset.

3.5. Integration Model

We then integrate the Topic Matrix and Similarity Matrix for our model. Depending on the structure and diversity of the document, we can decide the weights for the LDA Matrix and Similarity Matrix. The CNN-Daily Mail dataset works best with an LDA Matrix weight of 0.1 and a Similarity Weight of 0.9, while BBC Extractive summarization works well with an LDA weight of 0.8. These weights are computed as a function of the variance of the matrices or using brute-force techniques and trial-and-error methods. The Integrated Matrix will then be used for ranking the sentences and summarization. The formula for the Integrated Matrix is given by:

$$\textbf{Combined Score}(i) = \text{LDA Weight} \times \text{LDA Score}(i) + \text{TextRank Weight} \times \text{Similarity Score}(i) \quad (1)$$

Thus, the final Combined Score in the Integrated Matrix is a weighted sum for topical modeling and word overlap approaches of LDA and TextRank, respectively, for a more efficient and contextually relevant summary.

4. Evaluation

The summary generated is evaluated against the reference summary from the dataset. Various summary evaluation methods are available, such as the ROUGE Metric (ROUGE-N and ROUGE-L Scores) and ROUGE-S (Skip Bigram Scores). The evaluation is performed for multiple articles.

Table 1
Dataset Statistics for Train, Validation, and Test Sets

Metric	Train	Validation	Test
Average Length of Summary (words)	29.95	30.15	NA
Average Length of Article (words)	666.79	658.25	667.05
Longest Article (words)	11,433	5,883	5,193
Shortest Article (words)	0	10	5
Longest Summary (words)	123	128	NA
Shortest Summary (words)	1	6	NA
Average Number of Sentences per Article	26.59	26.43	26.63
Average Number of Sentences per Summary	1.25	1.26	NA
Average Length of Headline (words)	16.71	16.62	16.76
Longest Headline (words)	32	31	31
Shortest Headline (words)	4	7	6
Average Number of Sentences per Headline	1.05	1.04	1.06
Total Number of Entries	15000	1,500	2,500

The dataset available for the English language for ILSUM consists of 15,000 Article-Summary pairs for Training, 1500 pairs for Cross-Validation, and 2500 pairs for Test. The model proposed by us is a graph based extractive model based on the Textrank Algorithm. Textrank is a graph-based ranking algorithm which generates summary based on the structure of a single document and does not use observations to learn anything. Thus the model does not require Training Data and we can include data from Train and Cross Validation Datasets in Testing phase. The macro-statistics for the datasets are provided in **Table 1**.

Table 2
ROUGE Scores for Train Dataset

Articles	Metric	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-S
15000 Articles	**Recall**	0.5027	0.2246	0.3788	0.1907
	Precision	0.1684	0.0769	0.1273	0.0656
	F1	0.2428	0.1102	0.1834	0.0938

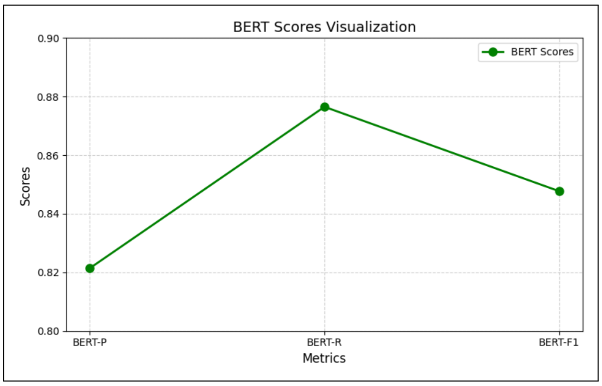
Table 3
ROUGE Scores for Cross Validation Dataset

Articles	Metric	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-S
1500 Articles	**Recall**	0.5191	0.2439	0.3968	0.2087
	Precision	0.1750	0.0847	0.1343	0.0736
	F1	0.2519	0.1210	0.1931	0.1047

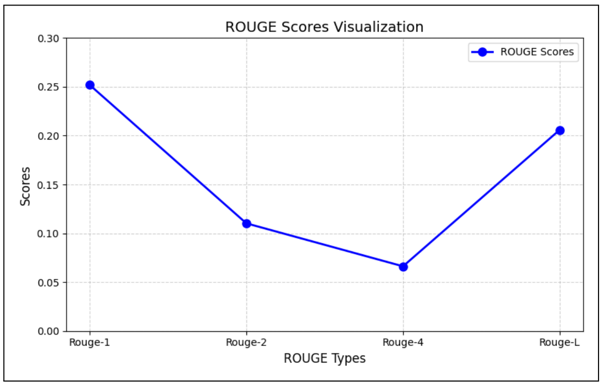
The results of this model on the train dataset, cross-validation dataset and test dataset are summarized in **Table 2**, **Table 3** and **Table 4** respectively. On further research into this model we found that this model is particularly suited for multi-sentence summaries and achieves high rouge scores for datasets with multi-sentence summaries (About 20% of the document's length) such as Webis Corpus [18] or BBC Summary Datasets. The ILSUM dataset has summaries that are roughly one to two sentence long and thus the model under-performs due to short nature of reference summaries.

Table 4
ROUGE and BERT Scores for Test Dataset

ROUGE-1	ROUGE-2	ROUGE-4	ROUGE-L	BERT-P	BERT-R	BERT-F1
0.2519	0.1102	0.0662	0.2059	0.8214	0.8765	0.8477



(a) BERT Scores



(b) ROUGE-N Scores

Figure 2a and Figure 2b visualizes the BERTScore and ROUGE metrics calculated on the test dataset respectively.

5. Acknowledgments

We would like to Thank FIRE 2024 Organising Team for conducting this shared task for Indian Language Summarization

6. Conclusion

In this thorough exploration of article summarization, unsupervised graph-based methodology has been used to generate an extractive summary on the provided English dataset. The effectiveness of the summarization model is calculated with the help ROUGE scores. The text summarization model exhibits moderate overlap with reference summaries, as indicated by ROUGE-1, ROUGE-2, ROUGE-4, and ROUGE-L scores of 0.2519, 0.1102, 0.0662, and 0.2059, respectively. Its strong semantic comprehension is reflected in high BERT-based precision (0.8214), recall (0.8765), and F1 score (0.8477), highlighting its effective contextual understanding. The successful results in English summarization provide a strong foundation to apply this methodology to other Indian languages. As this model is a graph based method, the semantic language barrier can be easily removed with the help of indic pre-processing libraries. Therefore, this can not only be used for English language but can also be used to generate summary for other Indian languages.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools for the research and while writing this paper.

References

[1] Z. Jalil, J. A. Nasir, M. Nasir, Extractive multi-document summarization: A review of progress in the last decade, IEEE Access 9 (2021) 130928–130946. doi:10.1109/ACCESS.2021.3112496.

[2] O. Tas, F. Kiyani, A survey automatic text summarization, PressAcademia Procedia 5 (2017) 205–213. doi:10.17261/Pressacademia.2017.591.
[3] W. S. El-Kassas, C. Salama, A. Rafea, H. Mohamed, Automatic text summarization: A comprehensive survey, Expert Systems with Applications 165 (2020) 113679. doi:10.1016/j.eswa.2020.113679.
[4] S. R. K. Harinatha, B. T. Tasara, N. N. Qomariyah, Evaluating extractive summarization techniques on news articles, in: 2021 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2021, pp. 88–94. doi:10.1109/ISITIA52817.2021.9502230.
[5] S. Sonawane, P. Kulkarni, C. Deshpande, B. Athawale, Extractive summarization using semigraph (essg), Evolving Systems 10 (2019). doi:10.1007/s12530-018-9246-8.
[6] R. Mihalcea, P. Tarau, TextRank: Bringing order into text, in: D. Lin, D. Wu (Eds.), Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 404–411. URL: https://aclanthology.org/W04-3252.
[7] S. Brin, L. Page, The anatomy of a large-scale hypertextual web search engine, Computer Networks 30 (1998) 107–117. URL: http://www-db.stanford.edu/~backrub/google.html.
[8] K. A. R. Issam, S. Patel, S. C. N, Topic modeling based extractive text summarization, arXiv preprint arXiv:2106.15313 (2021). URL: https://arxiv.org/abs/2106.15313.
[9] H. Jelodar, Y. Wang, C. Yuan, X. Feng, X. Jiang, Y. Li, L. Zhao, Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey, 2018. URL: https://arxiv.org/abs/1711.04305. arXiv:1711.04305.
[10] V. Gulati, D. Kumar, D. E. Popescu, J. D. Hemanth, Extractive article summarization using integrated textrank and bm25+ algorithm, Electronics 12 (2023). URL: https://www.mdpi.com/2079-9292/12/2/372. doi:10.3390/electronics12020372.
[11] D. F. O. Onah, E. L. L. Pang, M. El-Haj, A data-driven latent semantic analysis for automatic text summarization using lda topic modelling, 2023. URL: https://arxiv.org/abs/2207.14687. arXiv:2207.14687.
[12] N. Gialitsis, N. Pittaras, P. Stamatopoulos, A topic-based sentence representation for extractive text summarization, in: G. Giannakopoulos (Ed.), Proceedings of the Workshop MultiLing 2019: Summarization Across Languages, Genres and Sources, INCOMA Ltd., Varna, Bulgaria, 2019, pp. 26–34. URL: https://aclanthology.org/W19-8905. doi:10.26615/978-954-452-058-8_005.
[13] G. Erkan, D. R. Radev, Lexrank: Graph-based lexical centrality as salience in text summarization, Journal of Artificial Intelligence Research 22 (2004) 457–479. URL: http://dx.doi.org/10.1613/jair.1523. doi:10.1613/jair.1523.
[14] J. Shi, C. Liang, L. Hou, J. Li, Z. Liu, H. Zhang, Deepchannel: Salience estimation by contrastive learning for extractive document summarization, Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 6999–7006. doi:10.1609/aaai.v33i01.33016999.
[15] M. Niepert, M. Ahmed, K. Kutzkov, Learning convolutional neural networks for graphs, CoRR abs/1605.05273 (2016). URL: http://arxiv.org/abs/1605.05273. arXiv:1605.05273.
[16] H. Jin, T. ming Wang, X. Wan, Semsum: Semantic dependency guided neural abstractive summarization, in: AAAI Conference on Artificial Intelligence, 2020. URL: https://api.semanticscholar.org/CorpusID:214303164.
[17] I. Sukprapa, D. H. Nguyen, Text summarization based on argumentation techniques (2021).
[18] S. Syed, R. El Baff, J. Kiesel, K. Al Khatib, B. Stein, M. Potthast, News editorials: Towards summarizing long argumentative texts, in: D. Scott, N. Bel, C. Zong (Eds.), Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), 2020, pp. 5384–5396. URL: https://aclanthology.org/2020.coling-main.470. doi:10.18653/v1/2020.coling-main.470.
[19] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, 2020. URL: https://arxiv.org/abs/1906.08237. arXiv:1906.08237.
[20] R. Liu, M. Liu, M. Yu, H. Zhang, J. Jiang, G. Li, W. Huang, SumSurvey: An abstractive dataset of scientific survey papers for long document summarization, in: L.-W. Ku, A. Martins, V. Srikumar

(Eds.), Findings of the Association for Computational Linguistics ACL 2024, Association for Computational Linguistics, Bangkok, Thailand and virtual meeting, 2024, pp. 9632–9651. URL: https://aclanthology.org/2024.findings-acl.574. doi:10.18653/v1/2024.findings-acl.574.

[21] S. Satapara, B. Modha, S. Modha, P. Mehta, Findings of the first shared task on indian language summarization (ILSUM): approaches challenges and the path ahead, in: K. Ghosh, T. Mandl, P. Majumder, M. Mitra (Eds.), Working Notes of FIRE 2022 - Forum for Information Retrieval Evaluation, Kolkata, India, December 9-13, 2022, volume 3395 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 369–382. URL: https://ceur-ws.org/Vol-3395/T6-1.pdf.

[22] S. Satapara, B. Modha, S. Modha, P. Mehta, FIRE 2022 ILSUM track: Indian language summarization, in: D. Ganguly, S. Gangopadhyay, M. Mitra, P. Majumder (Eds.), Proceedings of the 14th Annual Meeting of the Forum for Information Retrieval Evaluation, FIRE 2022, Kolkata, India, December 9-13, 2022, ACM, 2022, pp. 8–11. URL: https://doi.org/10.1145/3574318.3574328. doi:10.1145/3574318.3574328.

[23] S. Satapara, P. Mehta, S. Modha, D. Ganguly, Key takeaways from the second shared task on indian language summarization (ILSUM 2023), in: K. Ghosh, T. Mandl, P. Majumder, M. Mitra (Eds.), Working Notes of FIRE 2023 - Forum for Information Retrieval Evaluation (FIRE-WN 2023), Goa, India, December 15-18, 2023, volume 3681 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023, pp. 724–733. URL: https://ceur-ws.org/Vol-3681/T8-1.pdf.

[24] S. Satapara, P. Mehta, S. Modha, D. Ganguly, Indian language summarization at FIRE 2023, in: D. Ganguly, S. Majumdar, B. Mitra, P. Gupta, S. Gangopadhyay, P. Majumder (Eds.), Proceedings of the 15th Annual Meeting of the Forum for Information Retrieval Evaluation, FIRE 2023, Panjim, India, December 15-18, 2023, ACM, 2023, pp. 27–29. URL: https://doi.org/10.1145/3632754.3634662. doi:10.1145/3632754.3634662.

[25] S. Satapara, P. Mehta, S. Modha, A. Hegde, S. HL, D. Ganguly, Overview of the third shared task on indian language summarization (ilsum 2024), in: K. Ghosh, T. Mandl, P. Majumder, D. Ganguly (Eds.), Working Notes of FIRE 2024 - Forum for Information Retrieval Evaluation, Gandhinagar, India. December 12-15, 2024, CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[26] S. Satapara, P. Mehta, S. Modha, A. Hegde, S. HL, D. Ganguly, Key insights from the third ilsum track at fire 2024, in: Proceedings of the 16th Annual Meeting of the Forum for Information Retrieval Evaluation, FIRE 2024, Gandhiinagar, India. December 12-15, 2024, ACM, 2024.

[27] S. Satapara, P. Mehta, D. Ganguly, S. Modha, Fighting fire with fire: Adversarial prompting to generate a misinformation detection dataset, CoRR abs/2401.04481 (2024). URL: https://doi.org/10.48550/arXiv.2401.04481. doi:10.48550/ARXIV.2401.04481. arXiv:2401.04481.