Summarizing English News Articles: Leveraging T5 and Google Gemini 1.0 Pro

M Saipranav, Murari Sreekumar, Durairaj Thenmozhi, Shreyas Karthik and Rahul VS

Sri Sivasubramaniya Nadar College Of Engineering, Rajiv Gandhi Salai (OMR), Kalavakkam, 603 110, Tamil Nadu, India

Abstract

Text summarization presents a considerable challenge in the field of Natural Language Generation (NLG) due to its inherent context dependent nature. Despite growing research in this area, text summarization for Indian languages has received limited attention. The ILSUM 2024 shared task aims to address this gap by using machine learning approaches in order to generate meaningful fixed-length summaries, either extractive or abstractive, of articles in multiple Indian languages. Our team focused on the English dataset and employed machine learning approaches such as T5 and Gemini 1.0 Pro to generate the summary. In terms of BERT scores, the Gemini model outperformed T5, achieving a BERT F1-score of 0.8675 compared to T5's 0.8489. However, T5 excelled in ROUGE metrics, with a ROUGE-L score of 0.2644 versus Gemini's 0.233. Our models achieved an overall rank of 6, based on their highest ROUGE and BERT scores.

Keywords

Text Summarization, Machine Learning Algorithms, Natural Language Processing, Natural Language Generation, T5 Model, Google Gemini 1.0 Pro, Text Analytics

1. Introduction

One of the most vital applications in the field of Natural Language Generation (NLG), a subset of Natural Language Processing (NLP), is text summarization. It consists of summarizing expansive text-based information into highly relevant summaries. Traditionally, text summarization was a labour-intensive task, requiring human effort and contextual understanding to navigate the nuances of language. This complexity is heightened by ambiguity, which is characteristic of any language, along with challenges such as the presence of metaphors, idioms, and cultural references, making it difficult for humans to interpret and condense information easily.

Efficient and accurate summarization has been made possible with NLG technologies transforming the approach to summarizing texts. Through this, it has become possible to develop complex algorithms that can break down language structures, hence making it possible to automatically summarize complex information. Tokenization, semantic analysis, and machine learning techniques empower computers to derive meaning from text, overcoming the main limitations to manual summarization.

In recent years, advancements in deep learning and NLP have led to sophisticated models for text summarization. Architectures such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) have demonstrated impressive capabilities in understanding and generating natural language, improving the quality and relevance of automatically generated summaries. Furthermore, pre-trained models and transfer learning enable fine-tuning for summarization tasks with relatively small datasets, enhancing their performance.

This year's edition of Indian Language Text Summarization Task – ILSUM 2024 [1] [2] [3] [4] [5] [6] [7], organised by the Forum for Information Retrieval Evaluation, consisted of two tasks:

- Task 1: Language Summarization For Indian Languages
- Task 2: Detecting Factual Incorrectness in Machine Generated Cross Lingual Summaries

Forum for Information Retrieval Evaluation, December 12-15, 2024, India

saipranav2310324@ssn.edu.in (M. Saipranav); murari2310237@ssn.edu.in (M. Sreekumar); theni_d@ssn.edu.in (D. Thenmozhi); shreyas2310140@ssn.edu.in (S. Karthik); rahul2310239@ssn.edu.in (R. VS)

0000-0003-0681-6628 (D. Thenmozhi)

We participated in Task 1: Language Summarization for Indian Languages, aiming to generate a meaningful, fixed length summaries for English Language texts from various given news articles consisting of a headline-news article pair.

We used 2 pre-trained models such as the Transformer-based T5 (Text-to-Text Transfer Transformer) and Google Gemini 1.0 Pro, which make use of huge datasets to learn linguistic patterns that enhance their summary quality. Using these methods, we overcome many challenges related to summarization, successfully gaining the confidence to apply it to various languages.

2. Related Works

Numerous efforts have been dedicated to advancing text summarization techniques across a range of Indian languages. Significant efforts have been made by researchers around the world to summarize text by applying machine learning models such as T5, BART and Pegasus. In addition to these, various other extractive and abstractive methods have been used that consistently provide excellent accuracy in summarizing texts.

Research [8] delves into the text summarization of English and Hindi. For the summarization of English dataset, the text is first preprocessed after which T5-base and T5-small models were applied. For regional languages, the texts were first preprocessed, translated to English and after summarizing with T5-base, it was translated back to the original regional languages.

Another research [9] attempts to summarize news in English, Hindi and Gujarati by using the data built using article and headline pairs scraped from various external websites. For Hindi and Gujarati, multilingual models such as MT5, MBart and IndicBART variants were used and PEGASUS, BART, T5 and ProphetNet models were fine-tuned and used on English data

Dhaval Taunk et.al [10] performed text summarization using multilingual models like mBART, mT5 and IndicBART. After fine-tuning the models on the datasets, they had also performed data augmentation in two ways - by augmenting the 3X data to the actual dataset and another by appending 5X data to the actual dataset which gave a significant improvement in the results. HuggingFace API and PyTorch were also used to fine-tune the models for 5, 7, 10 epochs for different models.

Research [11] implemented deep learning based approaches for article summarization in Indian Languages. SoTA Pegasus model was fine-tuned and used as it worked best for English dataset and IndicBART model along with data augmentation was used for Hindi dataset. Apart from these, BRIO(Bringing Order to Abstractive Summarization), SentenceBERT and T5 were used for English and XL-Sum, mBART and Translation+Mapping+PEGASUS were used for Hindi and Gujarati.

An extractive approach for automated summarization of Indian languages was used in research [12]. This research initially involves reduction of a text document to generate a new form which conveys the key meaning of the information contained in the text. Then the splitting and vectorization, word and sentence vectorization techniques were applied on the dataset after which K-means Clustering was used. K-means further comple ments the efficiency of extractive techniques as it is quick and suitable for small as well as large samples. Fine tuning the parameters may prove to be very fruitful.

3. Dataset Description

The dataset for this task (Table:1) contains more than 13,000 news articles in each language, drawn from the leading newspapers of the country. The objective of this task is to generate meaningful fixed-length summaries, be it extractive or abstractive, for each article.

A unique feature of the dataset is that the Indian language datasets exhibit code-mixing in both the headlines as well as the articles, where English is mixed with the Indian language content.

The dataset consists of CSV files in languages such as Hindi, Gujarati, English, Tamil, Kannada, Telugu, and Bengali, where each language has 3 CSV files containing the training, testing, and validation data, respectively. The training and validation data contain columns for headings, articles, and summaries, providing a foundation for generating meaningful summaries for the given articles.

3.1. Task Description

Generate concise and meaningful fixed-length summaries for news articles in multiple Indian languages, considering the challenge of code-mixing and script mixing. The dataset includes articles and headline pairs from leading newspapers in English, Hindi, Gujarati, and Bengali,etc. We have performed this task by using the English dataset.

Task	Language	No. of Samples
Training	English	9376
Validation	English	1500
Testing	English	2500

Table 1
Distribution of tweet samples across training, development, and testing for each language

4. Approach

We fine tuned the machine learning models: Gemini 1.0 pro and T5- Base using the training dataset, evaluated the models on the dev dataset and submitted our runs by applying the ML models on the test dataset.

4.1. Data Preparation

Our first step was to clean the data given (Figure: 1) in order to improve the performance of the machine learning models:

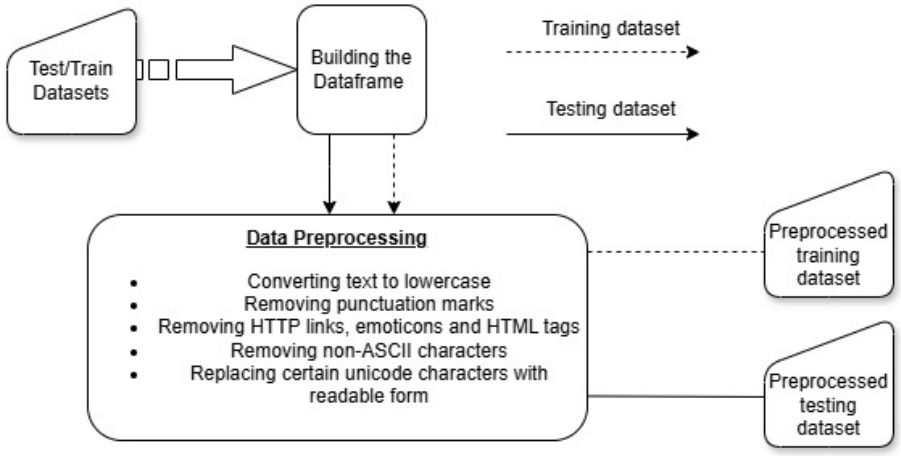

Figure 1: Data Preparation Process

1. Converting the text to lowercase: This ensures consistency in text data. By doing this the vocabulary size is reduced and it reduces the computational requirements.
2. Removing punctuation marks:They often point to external resources that are not relevant to the context of the text being analyzed.
3. Removing http links and emoticons:These do not contribute to the semantic meaning of the text.
4. Removing twitter mentions like `@username`
5. Removing HTML entities and tags : they often make the model find it difficult to understand the text which is present
6. Replaces specific Unicode characters (e.g., non-breaking spaces, dashes, quotes) with their appropriate readable forms.also removes non ASCII characters from the text.
   A sample of our results are shown in Table 2

Text	
Before Preprocessing	`Chandu Champion OTT Release: Here's How You Can Watch Kartik Aaryan's Film Online`
After Preprocessing	`chandu champion ott release heres how you can watch kartik aaryans film online`

Table 2
Preprocessing Results

4.2. Methodology

We have used 2 models to evaluate the dataset. They are:

4.2.1. Google Gemini 1.0 Pro:

Gemini 1.0 Pro [13] is a powerful large language model (LLM) designed for a variety of natural language tasks, including text summarization. It is a large language model (LLM) that is based on the Transformer architecture. This architecture is a type of neural network that is specifically designed for processing sequential data, such as text.

Gemini 1.0 Pro also uses a number of other techniques, such as pre-training on large datasets of text and fine-tuning on specific tasks. These techniques help to improve the model's accuracy and performance. Gemini 1.0 Pro generates text summarization by leveraging its deep understanding of language and context. It employs a combination of techniques, including:

Sequence-to-Sequence Modeling: The model treats summarization as a sequence-to-sequence task, where it takes a long input sequence (the original text) and generates a shorter output sequence (the summary). This approach allows the model to capture the overall context and structure of the original text.

Attention Mechanism: The attention mechanism enables the model to focus on the most relevant parts of the input text while generating the summary. By assigning weights to different words or phrases, the model can prioritize the most important information and exclude less relevant details.

Encoder-Decoder Architecture: The model consists of an encoder that processes the input text and a decoder that generates the summary. The encoder transforms the input text into a sequence of hidden representations, while the decoder uses these representations and the attention mechanism to generate the summary.

Pre-training on Large Datasets: Gemini 1.0 Pro is pre-trained on massive amounts of text data, allowing it to learn the nuances of language and develop a strong understanding of context. This pre-training helps the model generate more accurate and informative summaries.

Fine-tuning for Specific Tasks: For summarization tasks, the model can be further fine-tuned on specific datasets to improve its performance. This involves training the model on a large number of examples of input texts and their corresponding summaries, allowing it to learn the patterns and characteristics of effective summarization. The architecture diagram of Gemini 1.0 pro model is shown in Figure:2

4.2.2. T5 Text-to-Text Transfer Transformer

In this study, we have employed the T5 - Text to Text Transfer Transformer model developed by Google [14], for our text summarization task. This model features an encoder-decoder structure, specifically designed for the text-to-text approach, and is trained on a 750 GB dataset, the Colossal Clean Crawled Corpus (C4)[15]. The authors of the T5 paper (Raffel et al.)[14] achieved optimal performance by training the model for 1 million steps, utilizing a batch size of 2 to the power 11 sequences with a maximum length of 512 tokens, ensuring comprehensive learning across diverse tasks.

In T5, every task is converted to a text-text format, enabling the model to address any NLP task without requiring adjustments to the hyper parameters or loss functions. The model manages a variety

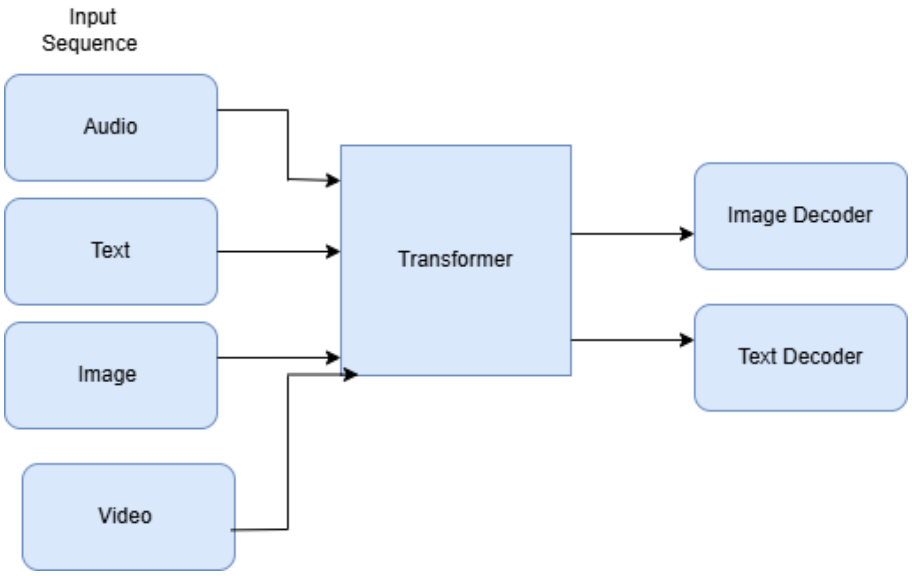

Figure 2: Architecture of gemini

of tasks by prepending a different prefix to the input corresponding to each task, e.g., for summarization: summarize {text} or for translation: translate {text} to {language} etc. It incorporates both supervised and self-supervised training methods, leveraging benchmarks like GLUE and SuperGLUE while also employing an approach to self-supervised training using corrupted tokens.

Additionally, T5 is "unified" in the sense that it can carry out multiple NLG tasks at once. Unlike other transformers like BERT and GPT2, it does not require distinct output layers for various tasks. The output can be a string of numbers, even for regression tasks.

Owing to the above reasons, the T5 model has performed better than several other contemporary SOTA architectures in different NLG tasks. The architecture of the T5 model is shown in Figure: 3.

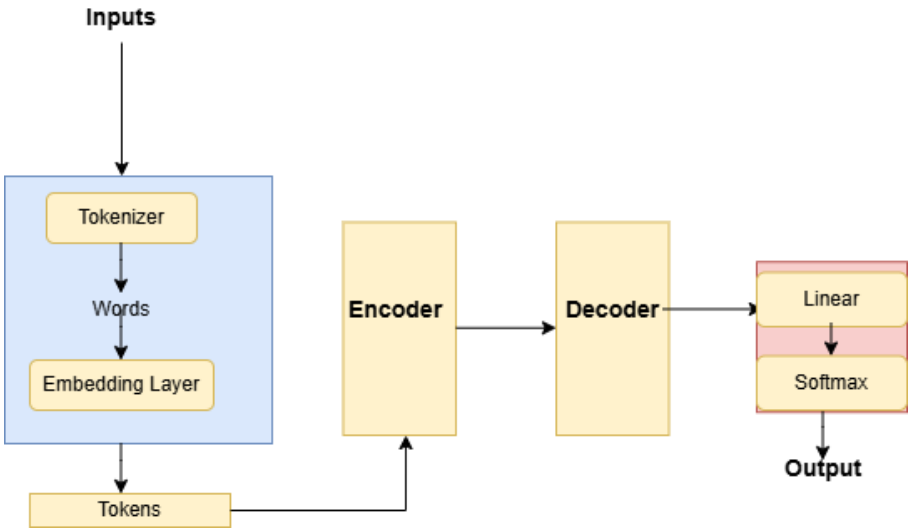

Figure 3: Architecture of T5

The T5 model's architecture features an encoder-decoder framework built from stacked identical layers. A position-wise feed-forward network and a multi-head self-attention mechanism make up each encoder layer's two primary parts. Layer normalization and residual connections are applied after each sub-layer. This structure is mirrored in the decoder, which also has a third sub-layer that applies multi-head attention to the encoder's outputs. The model performs exceptionally well on a

range of natural language processing tasks thanks to this design, which also makes information flow more efficient [14].

T5 For Text Summarisation: T5 comes in different sizes, such as *google-t5/t5-small, google-t5/t5-base, google-t5/t5-large, google-t5/t5-3b and google-t5/t5-11b.* We chose the *google-t5/t5-base* variant, which has 220 million parameters and effectively strikes a balance between model performance and computing efficiency. The model and tokenizer were initialized using HuggingFace's T5ForConditionalGeneration and T5Tokenizer, and the model was shifted to a CUDA-enabled GPU, to expedite processing time. The computational hardware used for this task is mentioned in Table 3

System Configuration:	Boston Supermicro 4U: Intel Skylake Processor. 128 GB DDR4 RAM, 6TB SATA HDD, 480GB SATA SSD, 4*Nvidia GeForce RTX 2080 11GB Graphics Cards
OS/Software	Ubuntu 18.04 LTS Server / Jupyter Notebook

Table 3
Hardware Configuration

Summarization was conducted in batches to optimize memory and computational throughput. Each input text was formatted as "summarize the following article, and convey the meaning briefly but exhaustively: {heading} {article}", allowing the model to focus on the heading to guide the summarization process.

Summaries were constrained to a maximum of 75 words and a minimum of 50 words, to maintain conciseness and prevent impacts on ROUGE precision and recall scores while evaluating against the ground truth summary.

To enhance the output quality, beam search (num_beams=15) was employed, evaluating multiple potential paths, while a length penalty of 1.3 discouraged verbosity, resulting in more fluent summaries. Additionally, repeated phrases were minimized using a No Repeat N-gram Size set to 5, improving readability and coherence.

5. Results and Performance Analysis

To evaluate the effectiveness of our models, we used the suite of BERT and ROGUE scores against our T5 and Gemini 1.0 Pro runs.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a widely-used set of metrics for evaluating automatic text summarization and machine-generated content against reference summaries. The most common types of ROUGE include ROUGE-N, ROUGE-L, and ROUGE-W. ROUGE-N measures the n-gram overlap, where n can be 1 for unigrams or 2 for bigrams, quantifying how well individual words or short phrases match. ROUGE-L assesses the longest common subsequence, providing insights into the syntactic structure alignment, while ROUGE-W weighs consecutive matches higher, favoring coherent text flow. BERTScore, on the other hand, leverages contextual embeddings from BERT (Bidirectional Encoder Representations from Transformers) to capture semantic similarity by comparing embeddings between reference and candidate sentences. This approach is beneficial as it goes beyond surface-level word matching, taking into account the context and meaning, which can be critical for complex text evaluation.

In terms of BERT scores, the Gemini model outperformed T5, achieving a BERT F1-score of 0.8675 compared to T5's 0.8489. However, T5 excelled in ROUGE metrics, with a ROUGE-L score of 0.2644 versus Gemini's 0.233. Our models achieved an overall rank of 6, based on their highest ROUGE and BERT scores.

The below tables (4 and 5) summarize the results of our team IdlyVadaSambar as measured by FIRE for the aforementioned approaches against what other teams obtained in the same task.

Rank	Team Name	Language	Run Name	BERTScore-Precision	BERTScore-Recall	BERTScore-F1
1	Data Lovers	English	run1.csv	0.8706	0.8862	0.8781
2	INITIATORS	English	run3.csv	0.8701	0.8789	0.8742
3	INITIATORS	English	run1.csv	0.8673	0.8802	0.8735
4	INITIATORS	English	run2.csv	0.8659	0.8751	0.8702
5	Data Lovers	English	run2.csv	0.8538	0.8847	0.8687
6	IdliVadaSambar	English	run2.csv	0.8529	0.8829	0.8675
7	Curious Coders	English	run1.csv	0.8684	0.8664	0.8672
8	NITK_Surathkal	English	run1.csv	0.8591	0.8686	0.8636
9	Squad	English	run1.csv	0.8764	0.8447	0.8601
10	Iem inturns	English	run1.csv	0.8482	0.8708	0.8591
11	IdliVadaSambar	English	run1.csv	0.8284	0.8708	0.8489
12	SynopSizers	English	run1.csv	0.8195	0.8706	0.8439
13	SCaLAR	English	task_1_en	0.8002	0.8284	0.8136

Table 4
BERTScore Results for Different Teams

Rank	Team Name	Language	Run Name	Rouge-1	Rouge-2	Rouge-4	Rouge-L
1	Data Lovers	English	run1.csv	0.3644	0.2060	0.1467	0.3133
2	INITIATORS	English	run1.csv	0.3435	0.1879	0.1357	0.2964
3	INITIATORS	English	run3.csv	0.3388	0.1702	0.1036	0.2834
4	Data Lovers	English	run2.csv	0.3238	0.1627	0.0992	0.2806
5	INITIATOR	English	run2.csv	0.3217	0.1589	0.1011	0.2705
6	IdliVadaSambar	English	run1.csv	0.3102	0.1554	0.0856	0.2644
7	Iem inturns	English	run1.csv	0.3044	0.1448	0.0843	0.2646
8	NITK_Surathkal	English	run1.csv	0.3039	0.1293	0.0664	0.2468
9	Curious Coders	English	run1.csv	0.2989	0.1392	0.0927	0.2497
10	IdliVadaSambar	English	run2.csv	0.2825	0.0944	0.02	0.233
11	SynopSizers	English	run1.csv	0.2295	0.0894	0.0513	0.1895
12	Squad	English	run1.csv	0.2214	0.0579	0.0111	0.1761
13	SCaLAR	English	task_1_en	0.1391	0.0437	0.0123	0.1189

Table 5
ROUGE Score Results for Different Teams

6. Conclusion

In this paper, we explored English text summarization using advanced NLP/NLG models T5 and Google Gemini 1.0 Pro, and assessed their generated summaries' quality against evaluation metrics such as ROUGE and BERTScore. Our findings indicate that these evaluation techniques validate both the coherence and informativeness of summaries, highlighting the robustness of text summarization models when measured against established benchmarks.

Google Gemini 1.0 Pro gave us the highest BERT score with BertScore-Precision at 0.8529, BertScore-Recall at 0.8829 and BertScore-F1 at 0.8675. T5 model produced the best ROUGE scores with ROUGE-1 at 0.3102, ROUGE-2 at 0.1554, ROUGE-4 at 0.0856 and ROUGE-L at 0.2644.

In conclusion, advancing NLP through this summarization technique offers potential for multilingual support and fine-tuning on diverse datasets, addressing the needs of various industries. Streamlining text summarization into real-world applications could make it a valuable tool for managing the rapidly expanding volume of information. By scaling this technology, we can further NLP's capabilities, enabling more systems to process, understand, and condense information across languages and domains.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] V. Ilanchezhiyan, R. Darshan, E. M. Dhitshithaa, B. Bharathi, Text summarization for indian languages: Finetuned transformer model application., in: FIRE (Working Notes), 2023, pp. 766–774.

[9] A. Urlana, S. M. Bhatt, N. Surange, M. Shrivastava, Indian language summarization using pretrained sequence-to-sequence models, arXiv preprint arXiv:2303.14461 (2023).

[10] D. Taunk, V. Varma, Summarizing indian languages using multilingual transformers based models, arXiv preprint arXiv:2303.16657 (2023).

[11] R. Tangsali, A. Pingle, A. Vyawahare, I. Joshi, R. Joshi, Implementing deep learning-based approaches for article summarization in indian languages, arXiv preprint arXiv:2212.05702 (2022).

[12] K. Kumari, R. Kumari, An extractive approach for automated summarization of indian languages using clustering techniques., in: FIRE (Working Notes), 2022, pp. 418–423.

[13] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al., Gemini: a family of highly capable multimodal models, arXiv preprint arXiv:2312.11805 (2023).

[14] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[15] TensorFlow, C4 dataset - tensorflow datasets, 2024. URL: https://www.tensorflow.org/datasets/catalog/c4, accessed on: 26-10-2024.