Abstractive Summarization of large articles in Hindi Language using IndicBART

Chaitanya Subhedar[1,*,†], Sheetal Sonawane[3]

[1]SCTR's Pune Institute of Computer Technology, Pune, India

Abstract

Natural Language Processing (NLP) is rapidly evolving, with significant advancements in tasks such as translation, summarization, and language understanding across multiple languages. The field has made remarkable progress in enabling machines to comprehend and generate human language, bridging communication gaps. However, summarization for Indian languages, especially low-resource ones like Hindi, remains a challenging problem due to limited availability of annotated datasets and linguistic diversity. This paper presents an approach using the IndicBART model, a pre-trained language model designed for Indian languages, to generate coherent and meaningful summaries for Hindi text, addressing the unique challenges associated with Indian language summarization. This approach also discusses the possibility of dividing the articles into smaller chunks to generate sub-summaries when compute resources are not available, and a smaller model is required to be used.

Keywords

Natural Language Processing, Text Summarization, IndicBART

1. Introduction

Natural Language Processing (NLP) has seen rapid advancements over the past decade, enabling machines to perform tasks such as translation, summarization, and sentiment analysis across diverse languages. These developments have significantly improved the ability to bridge language barriers and facilitate multi-language communication. However, challenges remain, especially in the context of low-resource languages, where the availability of high-quality annotated data is limited, hindering the effectiveness of various NLP models.

In the case of Indian languages, summarization poses a unique challenge due to the vast linguistic diversity and complex syntactic structures. While some progress has been made for widely spoken languages like Hindi, there is still a need for more refined approaches that can accurately capture the nuances and produce coherent summaries. This paper addresses these challenges by focusing on Hindi language summarization using pre-trained language models.

One of the initial approaches explored for Hindi summarization was the use of the IndicBERT model, a pre-trained language model specifically designed for Indian languages. IndicBERT has shown promising results in various language understanding tasks; however, it encountered significant challenges in tokenizing Hindi text for summarization. Specifically, the tokenizer often stripped away vowel markers and other diacritics, which are crucial for meaning in Hindi. This resulted in distorted tokenized representations that compromised the quality of generated summaries. The inability to retain essential linguistic components made IndicBERT unsuitable for the task, as it failed to adequately capture the semantic nuances of the original text.

Another approach tested was the mT5 model, a multilingual variant of the T5 (Text-to-Text Transfer Transformer) model that supports over 100 languages, including Hindi. The mT5-small model was chosen due to computational resource constraints. However, it struggled to learn effectively during the training phase, yielding low ROUGE scores that did not show significant improvement over successive epochs. While larger versions of mT5, such as mT5-base or mT5-large, could potentially offer better performance, upgrading to these models was not feasible due to the limited availability of GPU memory.

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

chaitanyasubhedar@gmail.com (C. Subhedar); sssonawane@pict.edu (S. Sonawane)

Consequently, mT5-small was found to be inadequate for producing high-quality summaries, particularly for a complex language like Hindi.

IndicBART emerged as a more suitable model for Hindi summarization compared to both IndicBERT and mT5-small. Unlike IndicBERT, IndicBART's tokenizer handled the intricacies of the Hindi language much more effectively, preserving vowel markers and diacritics, which are essential for capturing the meaning of words accurately. This resulted in better tokenized representations that retained the linguistic richness of the original text. Additionally, while both IndicBART and mT5-small are based on encoder-decoder architectures, IndicBART is specifically fine-tuned for Indian languages and has been pre-trained on a diverse corpus of Indian language texts. This likely gave it an advantage over mT5-small, which, although multilingual, may not have had the same level of exposure to Indian language data. Consequently, IndicBART demonstrated better learning capabilities and produced higher quality summaries, making it a more effective choice for the task.

We further discuss the approach of dividing large input text (news articles in this case) into smaller chunks so that fit into the maximum input size of IndicBART (1024 tokens).

2. Methodology and Algorithm

IndicBART is conceptually based on the mBART25/50 model family, utilizing a sequence-to-sequence architecture with both an encoder and a decoder. The model features six layers each for the encoder and decoder, with a hidden size of 1024 and a feed-forward filter size of 4096. It employs 16 attention heads to effectively capture contextual information across the input sequence. With a total of 244 million parameters,

Following are the details of our approach -

The model IndicBART allows an input size of a maximum of 1024 tokens. The average size of the input in the dataset (Headline + Article) was 474 words before the tokenizer was applied, while the maximum was 5089 words. IndicBART uses a sentence-piece tokenizer, which breaks down words into sub-words. Essentially, this means that the length of the input tensor comprising of the tokens that the model generates for each input article, will be even greater than the number of words. Thus, we cannot pass the whole input string directly into the This meant that if the Tokenizer for IndicBART was applied directly on the input row of each dataset, the tokenizer would simply truncate all the tokens past the 1024 count. This would result in severe information loss.

To counter this, the following approach was used. Initially we combined the headline and article into a single column called "Combined" so as to give the article more context in a single data sample. Then we executed an approach to form chunks of each sample in the "Combined" column. We then trained the model on these chunks, created the summary for each chunk, and then recombine the generated summary for each chunk belonging to the same article and evaluated against the target summaries.

Following is the step-by-step flow of the algorithm -

1. Create a column called Combined that combines each Headline with it's corresponding Article.
2. Initialize chunks = []
3. Initialize current chunk = []
4. Initialize current chunk length = 0
5. For each data sample in the Combined Column
    a) Split the whole text into sentences.
    b) Initial
    c) For each sentence
        i. Tokenize the sentence using the IndicBART Tokenizer.
        ii. If (length of tokenizedsentence) + currentchunklength > maxtokens then
            A. currentchunk.clear(), currentchunk.push(tokenizedsentence) // Create a new chunk
            B. currentchunklength = tokenizedsentence.length()
        iii. Else

	id	Summary	Combined
0	hindi_2024_train_0	इंटरनेट और खासकर सोशल मीडिया पर जानकारियों की ...	सोशल मीडिया पर नादान रहने में भलाई...: उंगलियो...
1	hindi_2024_train_1	Uterine Fibroid: यूटरस (Utrus) के अंदर बनने वा...	हंसती खेलती जिंदगी पर फायब्रॉयड्स का वार, कहीं...
2	hindi_2024_train_2	Statins Reduce Dementia Risk: कोलेस्ट्रॉल कंट्...	कोलेस्ट्रॉल की इस दवा से होगा ब्रेन की खतरनाक ...
3	hindi_2024_train_3	यह टावर शहर के मध्य पुराने नगर पालिका कार्याल...	ये है भारत का सबसे ऊंचा घंटाघर, 8 मील दूर तक स...
4	hindi_2024_train_4	निदा खान आला हजरत खानदान की बहू हैं. उन्हें शा...	निदा खान को दिए गए तीन तलाक को बरेली कोर्ट ने ...

Figure 1: Before Chunking

	id	Summary	Chunks
0	hindi_2024_train_0	इंटरनेट और खासकर सोशल मीडिया पर जानकारियों की ...	सोशल मीडिया पर नादान रहने में भलाई...: उंगलियो...
1	hindi_2024_train_0	इंटरनेट और खासकर सोशल मीडिया पर जानकारियों की ...	वो भीड़ जो उस व्यक्ति को मारना अपना 'धर्म' समझ...
2	hindi_2024_train_1	Uterine Fibroid: यूटरस (Utrus) के अंदर बनने वा...	हंसती खेलती जिंदगी पर फायब्रॉयड्स का वार, कहीं...
3	hindi_2024_train_1	Uterine Fibroid: यूटरस (Utrus) के अंदर बनने वा...	इसके अलावा यह भी अनुमान लगाया जाता है कि महिला...
4	hindi_2024_train_1	Uterine Fibroid: यूटरस (Utrus) के अंदर बनने वा...	वहीं, पेल्विक एरिया में तेज दर्द और हैवी ब्लीड...

Figure 2: After Chunking

A. currentchunk.push(tokenizedsentence)
B. currentchunklength += tokenizedsentence.length()

We have implemented the above using the pandas apply function in python, and we return an array of chunks for each data in the "Combined" column. We create an additional column named "Chunks" to store the returned array of Chunks. We now use the explode function provided by pandas to make sure each chunk has a common id. In this case the same id as the original data sample in the combined row.

Following this, we used the Trainer API provided by hugging face to train the IndicBART model. We trained the model for 10 epochs, with a weight decay of 0.01, and a batch size of 4. The model generated summary for each chunk, which we further recombined with the help of the id that we preserved in the dataset post chunking. An individual sub-summary is generated for each chunk and all the sub-summaries associated with a chunk having common ids are recombined to generate the final summary, which is what is used for rouge score evaluation.

3. Experimentation Details

3.1. Dataset

The dataset given for this task contained pairs of articles and headlines from various newspapers. A target summary has also been given against which validation is to be performed. The training, validation and test datasets consisted of 10427, 1500 and 3000 rows respectively.

The histograms in Figure 3 show the token count (post-tokenization) of the training dataset for the Article (including headline) and the Summary.

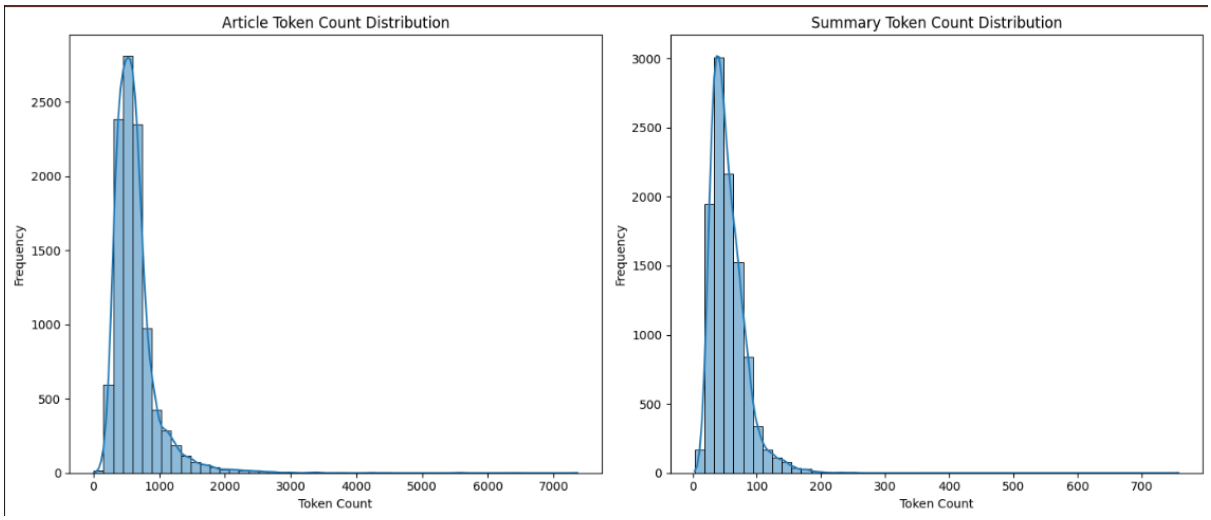

Figure 3: Token Counts for Article and Summary

Table 1
Rouge scores of Indic BART on validation set

Epoch	Rouge 1	Rouge 2	Rouge L	Rouge L Sum
1	24.190600	17.483600	23.869000	23.836200
2	24.644900	18.038200	24.288400	24.296900
3	24.569500	17.944300	24.182100	24.200600
4	24.859600	18.206600	24.464000	24.463400
5	24.424100	18.021200	24.019500	24.048800
6	25.298400	18.339500	24.906300	24.919000
7	25.100300	18.341100	24.672400	24.694700
8	25.202000	18.306900	24.797500	24.818400
9	25.210600	18.330100	24.817400	24.840900
10	25.254200	18.325700	24.869300	24.901600

4. Results

In the results, we observed a clear performance distinction between IndicBART and mT5-small when evaluated on the Hindi summarization dataset. IndicBART (Validation Set Results shown in Table 1) demonstrated consistently lower training and validation loss values across multiple epochs, indicating improved convergence. The ROUGE scores for IndicBART showed a steady increase, ultimately reaching 25.25 for ROUGE-1, 18.32 for ROUGE-2, 24.87 for ROUGE-L, and 24.91 for ROUGE-Lsum by the 10th epoch. In contrast, mT5-small (Validation Set Results shown in Table 2) yielded lower ROUGE scores, with ROUGE-1 peaking at 11.78, ROUGE-2 at 5.10, ROUGE-L at 11.60, and ROUGE-Lsum at 11.53 after five epochs. These metrics underscore IndicBART's ability to generate summaries with higher relevance and coherence for this task, outperforming mT5-small, which struggled to capture sufficient contextual information in its summaries, likely due to its smaller pre-trained vocabulary and fewer layers. IndicBART's performance on the Test Set is documented in Table 3 and Table 4.

Table 2
Rouge scores of mT5 on validation set

Epoch	Rouge 1	Rouge 2	Rouge L	Rouge L Sum
1	7.199400	2.875800	7.052600	7.058600
2	9.560700	2.783800	9.411800	9.428500
3	9.756800	3.457000	9.670700	9.701200
4	10.031700	3.622100	9.891900	9.905100
5	11.781600	5.096500	11.603400	11.533300

Table 3
Rouge scores of IndicBART on test set

Rouge 1	Rouge 2	Rouge-4	Rouge-L
0.3421	0.1713	0.102	0.312

Table 4
BERT scores of IndicBART on test set

BertScore-Precision	BertScore-Recall	BertScore-F1
0.7204	0.7449	0.7318

5. Conclusion and Future Work

In this paper we explored fine tuning the IndicBART model for Summarization by applying some additional data preprocessing techniques like chunking to cover the large amount of text data that overshot the input size of the model.

In the future, we would like to explore the how to transform the data further so as to establish more coherent relation between the chunked summaries. We also would explore using larger models like the mT5 family by acquiring stronger compute environments.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT and Grammarly in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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

[6] 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.

[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] N. Moratanch, S. Chitrakala, A survey on abstractive text summarization, in: 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), 2016, pp. 1–7. doi:10.1109/ICCPCT.2016.7530193.

[9] R. Dabre, H. Shrotriya, A. Kunchukuttan, R. Puduppully, M. M. Khapra, P. Kumar, Indicbart: A pre-trained model for natural language generation of indic languages, in: Findings of the Association for Computational Linguistics, 2022.

[10] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Huggingface's transformers: State-of-the-art natural language processing, 2019. URL: https://arxiv.org/abs/1910.03771. doi:10.48550/ ARXIV.1910.03771.

[11] S. Sonawane, P. Kulkarni, C. Deshpande, B. Athawale, Extractive summarization using semigraph (essg), Evolving Systems 10 (2019). doi:10.1007/s12530-018-9246-8.

[12] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. URL: https://arxiv.org/abs/1810.04805. doi:10.48550/ARXIV.1810.04805