Abstractive Text Summarization for Hindi Language using IndicBART

Arjit Agarwal[1,†], Soham Naik[1,†] and Sheetal Sonawane[1]

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

Abstract

Text summarization is an important application of natural language processing (NLP) especially in this era where there is an abundance of information on the internet. In such a scenario, it will be easier to navigate useful information quickly if a clear and concise summary of articles (or other text sources) can be generated. It is time consuming to give this task to humans because it involves scanning thousands of words and documents. But by using the advancements in natural language processing, models can be constructed for text summarization, that generate summaries in an adept and concise manner. There is a big scope for implementing these advanced natural language processing techniques for a low-resource language like Hindi because of its popularity and the fact that relatively less research work is done in this field. This paper is a part of the ILSUM shared task whose main focus is to generate abstractive text summaries using textual data in Hindi language. The accuracy of the generated summaries are checked using the ROUGE evaluation metric. We have achieved a ROUGE-1 Fscore of 0.544 on the testing dataset by using the IndicBART model for training.

Keywords

Text summarization, Abstractive, Natural language processing, Bart, Rouge

1. Introduction

Today, with the influx of huge amounts of textual data from numerous sources worldwide, there is a need to have robust mechanisms for scanning through them and represent the large amount of information in short and concise statements for better understanding of humans. That being said, the first thing which comes to mind is text summarization, in which models are used to generate a summary of the given text corpus.

Historically, there have been many advancements in natural language processing regarding text summarization. All the models that were involved were trained on textual data in English language. Text summarization can either be extractive or abstractive. In extractive text summarization, the final summary contains sentences from the article itself whereas in abstractive summarization, the models generates the summary after processing the input.

In abstractive text summarization, the model has to predict new words and terms which are different from the actual article. In addition to this, the model also has to generate the

Forum for Information Retrieval Evaluation, December 9-13, 2022, India

[†]These authors contributed equally.

arjitagarwal123@gmail.com (A. Agarwal); nsoham01@gmail.com (S. Naik); sssonawane@pict.edu (S. Sonawane)
https://arjitagarwal.tech/ (A. Agarwal); https://www.linkedin.com/in/sn-07/ (S. Naik);
https://www.linkedin.com/in/dr-sheetal-sonawane-6b94b01b (S. Sonawane)

words in the summary in a particular order so that they make sense to the reader and not sound incoherent. It can be argued that the summary generated by abstractive method is similar to human generated summaries. The different types of architectures used in abstractive summarization is given by Moratanch et al. in [1].

Hindi is an Indo-Aryan language spoken chiefly in the northern, central, eastern and western belt of India and is the fourth most widely spoken in the world after Mandarin, Spanish and English in linguistic terms. Most of the work done in text summarization has been focused on the English language and there is significantly less research work done in other languages like Hindi which has around 615 million speakers all over the world.

Text summarization of low resource languages has the following main challenges:

- lack of an extensive dataset
- language understanding and processing
- different structures and grammar rules

This proves to be a major hindrance and a cause for discouragement to those who want to delve further into this. There is a need for more similar work as done by Parida et al. [2] in other low resource languages. Having an extensive dataset also means that better performance can be achieved in tasks involving deep learning techniques. These challenges present an opportunity to perform research in these languages.

In this paper, we have used the Hindi language dataset (training and validation) provided by ILSUM [3, 4]. We have used the IndicBART [5] model for generating text summary on the given dataset. IndicBART is a multilingual sequence-to-sequence pre-trained model which supports 11 Indian languages. The ROUGE metric [6] is used to evaluate the accuracy of the generated summaries. The ROUGE-1,2 and 4 scores are considered for evaluating generated summary. This model comes under the package of Hugging Face [7] transformer models which are pretrained and can be used for transfer learning.

2. Related Work

Text summarization has found its applications in a variety of real-world scenarios and so there is a great amount of scholarly research done on it. Many approaches have been put forward and primitive models have been refined according to the advancements in NLP. The initial approaches were heavily focused on extractive text summarization. There are three main tasks used in extractive summarization as mentioned in [8]

- Capturing key aspects and storing them
- Scoring them according to importance
- Selecting the appropriate sentences for summary

Initial models were rule based models. Then came the production rule based models. After that researchers used Term Frequency (TF) and Inverse Document Frequency (IDF). It converts the textual sentences to vector space and each term is given a corresponding weight according to the frequency with which it occurs [9, 10, 11]. Another important method for extractive summary generation was proposed by Suanmali et al. [12] where they used fuzzy logic method

to generate text summary. Sonawane et al. [13] proposed the use of semigraph in extractive text summarization.

Significant developments finally started happening in the abstractive text summarization space after Sutskever et al. [14] proposed seq2seq learning framework based on LSTM and Bahdanau et al. [15] proposed attention-based models. After this discovery, several papers published in 2015 used neural networks for summarization tasks. Rush et al. [16] proposed a scalable local attention-based model for generating words of the summary conditioned on the input sentence. Hu et al. [17] created a large corpus of Chinese short text summarization dataset and based on the dataset used recurrent neural networks (RNNs) for summary generation.

Another consequential breakthrough was achieved when Devlin et al. [18] proposed a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. It paved the way for the use of transfer learning to create state-of-the-art models in NLP by adding just an additional output layer to the BERT model. Lewis et al. [19] proposed a model called as BART which is a denoising autoencoder for pre-training sequence-to-sequence models. BART is a combination of BERT and another model which was introduced by Radford et al.known as GPT [20]. BART is particularly fine-tuned for effective text generation. This architecture was used by a large number of researchers to solve text summarization problems in individual fields. [21, 22, 5, 23]

Slowly, multilingual text summarization has been coming into focus for research work. Hu et al. proposed the creation of comprehensive datasets in Chinese, Parida et al. proposed the creation of a synthetic dataset and showed that it increases accuracy in a research paper based on the German language. Sarwadnya et al. [24] proposed the use of graph based models for extractive text summarization in Marathi. Jain et al. [25] used Real Coded Genetic Algorithm to perform abstractive text summarization in Hindi. Sunitha et al. [26] provided a survey of existing methods in abstractive text summarization of Indian languages. Text summarization of low resource languages was mainly challenging due to limited dataset, language understanding and processing, grammar of language and unavailability of sufficient literature.

3. Proposed Architecture

The entire pipeline for this task is built using PyTorch and its libraries. Below describes the entire pipeline which is given in figure 1 from preprocessing the articles to final summary prediction. The entire training dataset is split with the training data containing 7600 examples and validation data containing 357 examples.

The first step in the pipeline involves preprocessing of article text. Since the dataset is created using web scrapper. There are some leftover HTML tags that pose no significance to the article. Since the HTML tags appearing in the articles are of similar format, these are removed easily. Next, any extra new lines and tab spaces and special characters are also removed so that the sentences are coherent.

The model we chose to use is a pre-trained IndicBART model provided by AI4Bharat. BART is a denoising autoencoder that maps a corrupted document to the original document and is implemented as a sequence-to-sequence model with a bidirectional encoder over corrupted text and a left-to-right autoregressive decoder. BART uses the standard sequence-to-sequence

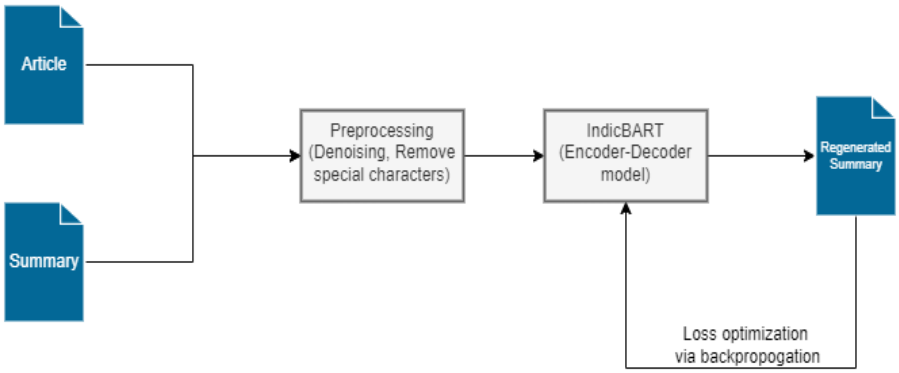

Figure 1: Proposed pipeline

Transformer architecture from [27] , except, following GPT, modifies ReLU activation functions to GeLUs [28] and initialise parameters from N (0, 0.02). For our IndicBART model, we use 6 layers in the encoder and decoder.The architecture is closely related to that used in BERT, with the following differences:

- Each layer of the decoder additionally performs cross-attention over the final hidden layer of the encoder (as in the transformer sequence-to-sequence model); and
- BERT uses an additional feed-forward network before word prediction, which BART does not. In total, BART contains roughly 10% more parameters than the equivalently sized BERT model.

IndicBART model can be used to build natural language generation applications for Indian languages by finetuning the model with supervised training data for tasks like machine translation, summarization, question generation, etc. The model is much smaller than the mBART and mT5(-base) models, so less computationally expensive for finetuning and decoding. The model is trained on large Indic language corpora (452 million sentences and 9 billion tokens) which also includes Indian English content. The loss used here is the language modelling loss.

Two new datasets(training and validation) are created by extracting the article text and summary which are needed as an input to our selected model. The model is finetuned on this dataset with summarization as downstream task. Hugging Face have provided a script which can be run with suitable parameters to begin finetuning.

4. Experimentation Details

this section is divided into dataset statistics and performance evaluation which is described in the sections 4.1 and 4.2.

4.1. Dataset Description

The dataset given by ILSUM for this task is built using articles and headline pairs from several leading newspapers of the country. The datasets are used for training, validation and testing

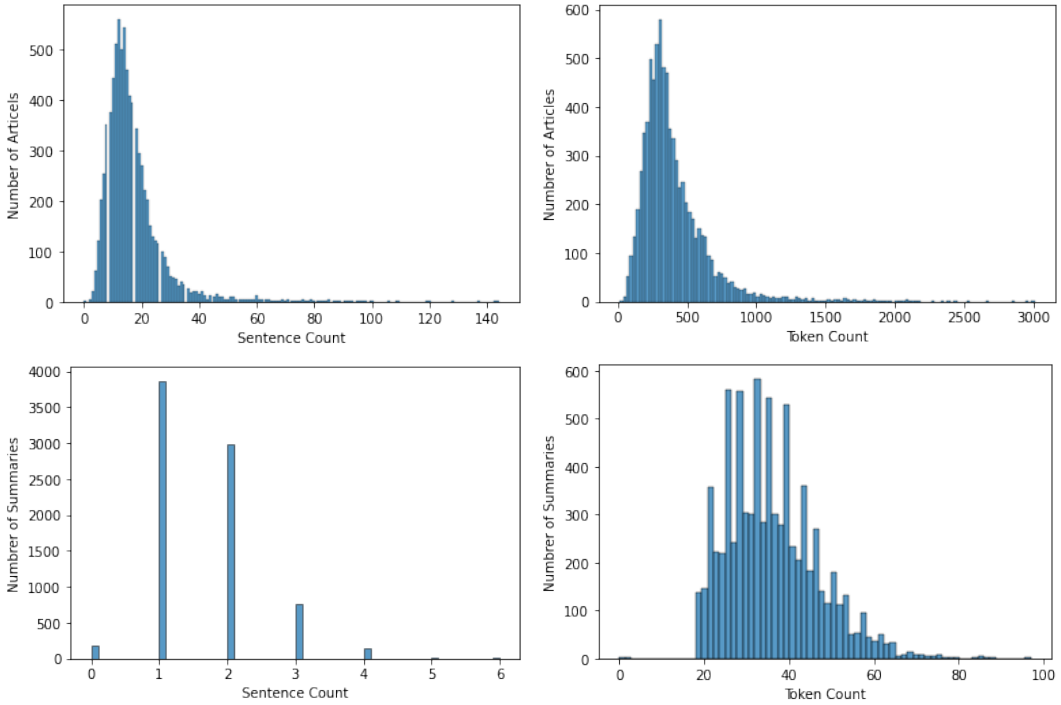

Figure 2: Number of tokens and sentences in training data

purposes. The dataset is private used for text memorization and is only available to teams that have registered for the ILSUM competition.

The training dataset was released in the beginning which consisted of 7957 training examples. The detail statistics is given in figure 2. Each training example consisted of an article headline, text, it's corresponding summary and an article ID unique to each article. The validation dataset consisted of 569 examples each having an article ID and the article. This dataset was used as a test dataset to be used for testing the submission portal. The third and final test dataset was similar to the validation dataset and it consisted of 2842 examples.

4.2. Performance evaluation

The model was finetuned using using the script provided by Hugging Face using Nvidia K80 with 12GB of available memory. Most of the parameters originally used to train the model are used with some exceptions.

An additional weight decay of 0.0001 is added. Since the memory available to finetune the model is not sufficient for including higher batch sizes, the training batch size is kept at 4 and gradient_accumulation _steps is kept as 16 which effectively enables a batch size of 64 with the trade-off for higher finetune times for the same number of epochs. The evaluation batch size and eval _accumulation _steps is kept at 1 so that the final score is calculated by averaging over each validation sample. To make the finetuning more stable, the first 100 steps are reserved as warmup steps. The model is restricted to generate 75 tokens in the summary to make the more

concise and aligned with the task grading requirements.

5. Results

The accuracy of the generated summary is evaluated by using the ROUGUE score. It stands for Recall-Oriented Understudy for Gisting Evaluation. It is the most widely used metric for the evaluation of abstractive text summaries because of its correlation with the human generated summaries [29]. ROGUE relies mainly on the lexical overlaps such as the n-grams and sequence overlaps between the generated summaries and the actual summaries. Higher the overlap, higher the ROGUE score and hence greater the similarity between generated summary and actual summary.

	Metrics	Validation Dataset	Test Dataset
Rouge-1	**F1-Score**	**0.551466**	**0.544284**
	Precision	0.498257	0.489564
	Recall	0.657105	0.652449
Rouge-2	**F1-Score**	0.457683	0.443253
	Precision	0.413388	0.396962
	Recall	0.546676	0.534722
Rouge-3	**F1-Score**	0.431636	0.414829
	Precision	0.389511	0.370719
	Recall	0.517708	0.503161
Rouge-4	**F1-Score**	0.417699	0.399905
	Precision	0.376664	0.356794
	Recall	0.503248	0.487699

Table 1
ROUGE Score

Table 1 shows the results we obtained after running 3 epochs on the test and validation set. The table 1 contains ROUGE scores which are calculated for different numbers of n-grams, which range from 1 to 4. The score shows the rouge score is decreasing as the Ngrams are increasing. The test and validation rouge score was calculated by taking the first 75 tokens along with the right number of sentences.

6. Sample Input and Generated Summary

Sample Input Article:

हिंगोजंग (मणिपुर): मणिपुर के हिंगोरानी में सुरक्षाबलों ने 4 आतंकवादियों को मार गिराया. असम राइफल्स और भारतीय सेना की 3 कोर ने एक संयुक्त ऑपरेशन में इन आतंकियों को ढेर किया. रक्षा पीआरओ ने यह जानकारी दी. उन्होंने बताया कि 'मणिपुर के हिंगोरानी में संयुक्त ऑपरेशन के दौरान असम राइफल्स, भारतीय सेना की 3 कोर सहित सुरक्षाबलों ने 4 आतंकवादियों को ढेर कर दिया. ऑपरेशन कल (शनिवार) शुरू किया गया था और आज (रविवार) सुबह गोलीबारी शुरू हुई. आतंकवादी कुकी समूह के थे.'

Generated Summary:

मणिपुर के हिंगोरानी में सुरक्षाबलों ने 4 आतंकवादियों को मार गिराया. असम राइफल्स और भारतीय सेना की 3 कोर ने एक संयुक्त ऑपरेशन में इन आतंकियों को ढेर किया.

Expert Summary:

मणिपुर के हिंगोरानी में सुरक्षाबलों ने 4 आतंकवादियों को मार गिराया. असम राइफल्स और भारतीय सेना की 3 कोर ने एक संयुक्त ऑपरेशन में इन आतंकियों को ढेर किया.

The above sample generated summary has an Rouge1 F-Score of 1.0

7. Conclusion and Future Work

Hindi Language summarization is a challenging task. IndicBART is trained on high quality news corpus. In this paper, we have explored pretrained IndicBART for Hindi language. The performance of pre-trained IndicBART model has been fine tuned to achieve good results for text summarization in Hindi language.

As a next step, We plan to apply better pre-processing techniques to enhance the quality of input data so that the performance of the model also increases. We also plan to increase the hardware requirements to create a robust pipeline so that a higher number of epochs can be trained in a less amount of time.

References

[1] 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.

[2] S. Parida, P. Motlicek, Abstract text summarization: A low resource challenge, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),

Association for Computational Linguistics, Hong Kong, China, 2019, pp. 5994–5998. URL: https://aclanthology.org/D19-1616. doi:10.18653/v1/D19-1616.

[3] 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: Working Notes of FIRE 2022 - Forum for Information Retrieval Evaluation, Kolkata, India, December 9-13, 2022, CEUR Workshop Proceedings, CEUR-WS.org, 2022.

[4] S. Satapara, B. Modha, S. Modha, P. Mehta, Fire 2022 ilsum track: Indian language summarization, in: Proceedings of the 14th Forum for Information Retrieval Evaluation, ACM, 2022.

[5] 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.

[6] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, 2004, p. 10.

[7] 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.

[8] V. N. Gudivada, Chapter 12 - natural language core tasks and applications, in: V. N. Gudivada, C. Rao (Eds.), Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications, volume 38 of *Handbook of Statistics*, Elsevier, 2018, pp. 403–428. URL: https://www.sciencedirect.com/science/article/pii/S0169716118300257. doi:https://doi.org/10.1016/bs.host.2018.07.010.

[9] J. Plisson, N. Lavrac, D. Mladenic, A rule based approach to word lemmatization, in: Proceedings of IS04, 2004.

[10] M. Gupta, N. K. Garg, Text summarization of hindi documents using rule based approach, in: 2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE), 2016, pp. 366–370. doi:10.1109/ICMETE.2016.104.

[11] E. Marsh, H. Hamburger, R. Grishman, A production rule system for message summarization, in: Proceedings of the Fourth AAAI Conference on Artificial Intelligence, AAAI'84, AAAI Press, 1984, p. 243–246.

[12] L. Suanmali, N. Salim, M. S. Binwahlan, Fuzzy logic based method for improving text summarization, CoRR abs/0906.4690 (2009). URL: http://arxiv.org/abs/0906.4690. arXiv:0906.4690.

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

[14] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, 2014. URL: https://arxiv.org/abs/1409.3215. doi:10.48550/ARXIV.1409.3215.

[15] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, 2014. URL: https://arxiv.org/abs/1409.0473. doi:10.48550/ARXIV.1409.0473.

[16] A. M. Rush, S. Chopra, J. Weston, A neural attention model for abstractive sentence summarization, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 379–389. URL: https://aclanthology.org/D15-1044. doi:10.18653/v1/D15-1044.

[17] B. Hu, Q. Chen, F. Zhu, LCSTS: A large scale Chinese short text summarization dataset, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1967–1972. URL: https://aclanthology.org/D15-1229. doi:10.18653/v1/D15-1229.

[18] 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.

[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. URL: https://arxiv.org/abs/1910.13461. doi:10.48550/ARXIV.1910.13461.

[20] A. Radford, K. Narasimhan, Improving language understanding by generative pre-training, 2018.

[21] J. Xu, Abstractive summarization on covid-19 publications with bart, 2020.

[22] J. Zhang, Y. Zhao, M. Saleh, P. J. Liu, Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, 2019. URL: https://arxiv.org/abs/1912.08777. doi:10.48550/ARXIV.1912.08777.

[23] I. Beltagy, M. E. Peters, A. Cohan, Longformer: The long-document transformer, CoRR abs/2004.05150 (2020). URL: https://arxiv.org/abs/2004.05150. arXiv:2004.05150.

[24] V. V. Sarwadnya, S. S. Sonawane, Marathi extractive text summarizer using graph based model, in: 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1–6. doi:10.1109/ICCUBEA.2018.8697741.

[25] A. Jain, A. Arora, J. Morato, D. Yadav, K. V. Kumar, Automatic text summarization for hindi using real coded genetic algorithm, Applied Sciences 12 (2022). URL: https://www.mdpi.com/2076-3417/12/13/6584. doi:10.3390/app12136584.

[26] C. Sunitha, A. Jaya, A. Ganesh, A study on abstractive summarization techniques in indian languages, Procedia Computer Science 87 (2016) 25–31. URL: https://www.sciencedirect.com/science/article/pii/S1877050916304604. doi:https://doi.org/10.1016/j.procs.2016.05.121, fourth International Conference on Recent Trends in Computer Science Engineering (ICRTCSE 2016).

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[28] D. Hendrycks, K. Gimpel, Bridging nonlinearities and stochastic regularizers with gaussian error linear units, CoRR abs/1606.08415 (2016). URL: http://arxiv.org/abs/1606.08415. arXiv:1606.08415.

[29] A. Cohan, N. Goharian, Revisiting summarization evaluation for scientific articles, 2016. URL: https://arxiv.org/abs/1604.00400. doi:10.48550/ARXIV.1604.00400.