Studying the Effect of Hindi Tokenizer Performance on Downstream Tasks

Rashi Goel
Manipal Institute of Technology
Manipal Academy of Higher Education, India
rashigoel2017@gmail.com

Fatiha Sadat
Université du Québec à Montréal
Montréal, Québec, Canada
sadat.fatiha@uqam.ca

Abstract

This paper deals with a study on the effect of training data size and tokenizer performance for Hindi language on the eventual downstream model performance and comprehension. Multiple monolingual Hindi tokenizers are trained for large language models such as BERT and intrinsic and extrinsic evaluations are performed on multiple Hindi datasets. The objective of this study is to understand the precise effects of tokenizer performance on downstream task performance to gain insight on how to develop better models for low-resource languages.

1 Introduction

Large Language Models (LLMs) have shown extraordinary performance in a range of Natural Language Processing (NLP) tasks, including both text classification and text generation. They are made use of across the world. After the success of many monolingual LLMs such as BERT (Devlin, 2018) and GPT, multilingual LLMs were built over these foundational models, increasing the number of languages they were pre-trained on, using different architectures and expanding the number of parameters. Some multilingual language models such as mBERT, mBART (Liu, 2020), Llama (Touvron et al., 2023), the more recent GPT versions, BLOOM (Workshop et al., 2022) have been trained on more than hundred languages. However, there is a skewed distribution in the quantity of the different languages they have been trained, causing bias in their predictions, in terms of languages as well as cultures. For Indic languages in specific, many LLMs have been built by using the aforementioned models trained on large corpora of multiple Indian languages. These include IndicBERT (Doddapaneni et al., 2022), IndicBART (Dabre et al., 2021), MuRIL (Khanuja et al., 2021), OpenHathi (sar). While India has hundreds of languages, most of them are however very low-resource, making training on them very hard.

The process of pre-training these LLMs involve processing large amounts of text data and make them perform tasks like Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) to learn semantic embeddings of the sentences that are input. While a lot of work has been done for the mechanisms and architecture of the models, a relatively under-investigated aspect of tokenizers is the impact of the tokenizer performance on the performance of the model.

Tokenizer performance plays a crucial role in the performance of LLMs. The quality of tokenization can have a huge effect on the contextual understanding and linguistic ability of the model. This project aims to investigate the effect of tokenizer performance on LLMs for Indian Languages. Since there are hundreds of Indian Languages, for this project, only Hindi was chosen as a representative language to conduct experimentation and evaluate results. Initial rudimentary investigations showed that there exist issues even with existing Indian LLMs. Hindi represents its vowel sounds within a word using 'matras' for the vowel letters, as do many other Indian Languages. However, these are encoded as accent marks in digital representations. It was seen that when IndicBERT (Doddapaneni et al., 2022) was used to tokenize Hindi text, it removed these accents as part of its pre-tokenization. The removal of these 'matras' would remove a lot of the semantic sense behind the words, equivalent to removing all the vowel characters from English words. Furthermore, it can also be seen in Figure 1 that the model splits the words very frequently into small-length tokens, to an average of around 2-4 characters, essentially removing the meaning behind the words, to simply represent repeating character groups. Such tokenization would deprive the model of any semantic understanding of the words, and would not allow it to gain meaningful context of the given text in order to perform its

Proceedings of the First Workshop on Natural Language Processing for Indo-Aryan and Dravidian Languages (IndoNLP2025), pages 71–76
January 20, 2025.

desired task.

This project thus extensively investigates the effect of the training data and tokenizer performance on subsequent downstream task performance by developing several monolingual tokenizers and models in Hindi, making use of different training data sizes as well as tokenization algorithms.
The remainder of the paper is structured as follows. Section 2 discusses related studies of tokenizer parameters and their effect on low-resource languages, section 3 outlines the methodology of the research in terms of tokenizers used, models trained and intrinsic and extrinsic evaluations performed along with the data used. Finally, section 4 presents the results of the experiments. Section 5 highlights the final findings and inferences and section 6 discusses limitations of the current research with future scope.

2 Related Work

Tokenizers are a relatively unexplored aspect in the training of LLMs. Ali et al. (2023) investigated the intrinsic and extrinsic performance of tokenizers in monolingual and multilingual settings for 5 European languages. 24 tokenizers were trained with corresponding transformer-based decoder models making use of them, which were fine-tuned on a range of downstream tasks. There experiments showed that there was some correlation between certain metrics and downstream task performance, however, a more fine-grained analysis was required.
Kaya and Tantuğ (2024) also investigate tokenizers for Turkish, a morphologically rich and relatively less-studied language. They studied the fine-grained effect of tokenization granularity based on the training data, vocabulary size and algorithm. They displayed that these factors play a role in tokenization quality as well as downstream task performance, especially for morphologically complex words so the model can attain contextually meaningful tokens.
Rajab (2022) investigated the effect of the tokenization algorithm on low-resource African languages for Neural Machine Translation. Being agglutinative languages, they showed the improvement in performance when SentencePiece BPE was used instead of BPE tokenization, since it encodes the whitespace character and does not require words to be space separated.

3 Methodology

3.1 Tokenizers

To carry out the experimentation, several monolingual tokenizers were trained from a Hindi corpus. Four tokenization algorithms were used:

- Wordpiece (Schuster and Nakajima, 2012)
- Unigram (Kudo, 2018b)
- Byte-pair encoding (BPE) (Sennrich, 2015)
- Sentencepiece BPE (Kudo, 2018a)

These are all subword-based tokenizers, which deal with the challenges faced by word based tokenization and character based tokenization. Word-based tokenizers usually require large vocabularies and are unable to handle out-of-vocabulary words. Character-based tokenizers output long tokenized sequences with less meaningful individual tokens. Subword tokenizers use the training corpus to learn split or merge rules, based on their algorithm, to effectively separate given words, more often than not, into their stem and suffixes, so more meaningful tokens are created and it can deal with variations of the same word.
For each algorithm, a tokenizer was developed with a subset of 2M sentences from the raw corpus. The same dataset and vocabulary size was used for each of the algorithms.

3.2 Models

For each tokenizer, a subsequent BERT (Bidirectional Encoder Representation from Transformers) model was pre-trained for a Masked Language Modeling task. To save on computation, smaller BERT models were developed. These consisted of only 6 attention heads instead of 12, with a total of 82M parameters. The same architecture was used for all tokenizers with the intent of simply performing ablations of data and tokenizer performance on downstream tasks. The models were fine-tuned on multiple tasks, such as Sentiment Analysis and Named-Entity Recognition. Sentiment analysis involves classifying a text as having either positive, negative or neutral sentiment, by making use of the words, semantics as well as tone of the text. Named-Entity Recognition locates and classifies the individual words of a text as Named Entities such as person (PER), location (LOC), etc. To fine-tune for these tasks, a classification head was also

	Text	Tokenization
1	तब वह मोटी थी और जेरी ने उस पर ध्यान नहीं दिया था।	_तब, _वह, _मट, _थ, _और, _जर, _न, _उस, _पर, _ध, यन, _न, ह, _द, य, _थ, ।,
2	अक्षरा हासन इस फिल्म से अपना डेब्यू कर रही हैं।	_अक, ष, र, _हसन, _इस, _फल, म, _स, _अपन, _डब, य, _कर, _रह, _ह, ।,
3	साजन और इला हिंदी फिल्मों के नियमित किरदार नहीं हैं।	_स, जन, _और, _इल, _हद, _फल, म, _क, _न, यम, त, _कर, दर, _न, ह, _ह, ।,

Figure 1: Tokenization of sample texts by IndicBERT

added to the model, consisting of linear layers followed by the final softmax layer. Fine-tuning was performed for 15 epochs.

3.3 Evaluations

Once developed, several intrinsic and extrinsic evaluations were carried out on the tokenizers and models respectively. The tokenizer performance was evaluated using 3 primary metrics:

- Number of unique tokens : This is number of unique tokens the model splits the text in the dataset into. A higher number of unique tokens indicates that the model captures the different words more effectively. A large number of repeated tokens (fewer unique tokens), conversely, indicates that the model splits the words into a large number of smaller repeating units, which would take away some of the semantic sense of the different words.

- Subword fertility ratio: It measures the average number of subwords per word in the text, as a ratio of the total number of tokens produced and the number of words in the text. A higher value means the model is producing a larger number of subwords per word, leading to over-segmentation and lesser contextual value due to the lower sequence length.

- Proportion of continued words: This is the ratio of words the tokenizer splits into two or more subwords, that is, the ratio of continued words in the tokenizer output and the total number of words in the text. While the fertility ratio gives a measure of the extent to which each word is split, this metric indicates how often words in the text are split. A higher value means the tokenizer is segmenting a large proportion of words and has not captured many words in the language.

These metrics provide a broad view of the effectiveness of the tokenizers in terms of how well they can segment meaningful subwords from texts to garner generalizability to unseen data while still retaining semantic sense. They are calculated on a held-out test set. The finetuned models are then evaluated on their performance in their respective tasks, using the accuracy of predictions as the metric, since these tasks are both multi-class classification tasks. Further, for Sentiment Analysis, the quality of the sentence embeddings were also examined. Sentence embeddings are usually extracted as the embedding of the [CLS] token from the pooler layer of the model. This embedding passes to the classification head to be segregated into its corresponding sentiment label. These sentence embeddings were examined to see how well they represented the positive, negative or neutral sentiment of the text by checking how well they cluster into their ground truth labels. The silhouette score for each model was calculated to evaluate how well sentences sharing similar sentiments were clustering. The silhouette score provides a metric over the inter-class and within-class distance. A high score indicates low intra-class distance and high inter-class distance. Ideally, sentences sharing similar sentences in the fine-tuned models should align closely with each other, and be far apart from the clusters of other sentiments.

3.4 Data

Several sources of digital Hindi text data were used to carry out the experiments in the project. The raw text corpus for training the tokenizers and pretraining the BERT models was obtained from the IndicNLP corpus (Git). This is a corpus developed by AI4Bharat, a research lab in IIT Madras which develops tools, models and datasets for NLP in Indian Languages. The corpus consists of crawled data from numerous web sources, including news-

papers, books and magazines in several Indian languages. The Hindi subset of this corpus was used, which in total consisted of 62.9M sentences.
For sentiment analysis, AI4Bharat's Hindi movie reviews dataset was used. This is part of the Indic_GLUE (Kakwani et al., 2020) dataset which consists of datasets for several Natural Language Understanding tasks to evaluate model performance. The sentiment analysis dataset consists of movie reviews, collected by IIT Patna, with each review annotated with its corresponding sentiment (positive, neutral or negative).
For Named-Entity Recognition, AI4Bharat's Naamapadam dataset (Mhaske et al., 2023) was used. This consists of annotated data for 11 Indian Languages. The data is produced by using the English-Indian Language parallel corpus and transferring the labels from the English side to the correct corresponding word on the Indian Language side.

4 Results

To carry out evaluations, intrinsic tokenizer metrics were first calculated for a held-out test corpus of text, which consisted of 54961 words. Table 1 shows the intrinsic metrics of the 4 tokenizers created using the corresponding algorithms.

Tokenizer	Unique Tokens	Fertility	Continued Words
Unigram	6990	1.2768	0.1307
Wordpiece	6961	1.1599	0.0219
BPE	1938	3.3367	0.8735
SentencePiece	7724	1.2082	0.0787

Table 1: Intrinsic metrics

It can be seen that Wordpiece shows the best performance in both subword fertility and proportion of continued words. The results of SentencePiece and Unigram are also comparable. BPE shows the worst performance, with the lowest number of unique tokens and the highest subword fertility and proportion of continued words. This suggests that it splits each word in the text into a large number of small, repeating units which would likely fail to capture the semantics or nuances of the words.
It can be seen from Table 2 that the monolingual Hindi Unigram, Wordpiece and SentencePiece tokenizers perform better than tokenizers of benchmark LLMs, IndicBERT and mBERT, despite being trained on a significantly lower amount of data.

Tokenizer	Unique Tokens	Fertility	Continued Words
IndicBERT	1327	1.6643	0.4589
mBERT	1280	2.0424	0.4284

Table 2: Intrinsic metrics of benchmark LLMs

Table 3 shows the performance of the BERT models, pretrained from the corresponding tokenizer, fine-tuned for the Sentiment Analysis task. While the models trained on the Unigram, Wordpiece and Sentencepiece algorithm show comparable performance, there is a large drop in the performance of the BPE tokenizer based BERT model. This follows the hypothesis that the poor tokenizer performance caused worse downstream task performance, as all other factors in the models were kept constant.

Tokenizer	Accuracy	Silhouette score
Unigram	0.6355	0.1160
Wordpiece	0.6483	0.1263
BPE	0.5774	0.0706
SentencePiece	0.6581	0.1141

Table 3: Results of Sentiment Analysis

The silhouette scores of the sentence embeddings (before being processed through the classification head) also show similar trends, being the lowest for BPE, and comparable for the other 3 algorithms. This indicates the model's inherent understanding of the language based on how well it can represent the sentences. Table 4 shows the performance of the models fine-tuned for the Named Entity Recognition task. Once again, the Unigram, Wordpiece and SentencePiece based models show comparable performance, whereas there is a drop in the performance of the BPE based model. This shows a significant correlation between the quality of the tokenizer and the downstream performance of the model.

Tokenizer	Accuracy
Unigram	0.9384
Wordpiece	0.9381
BPE	0.8878
SentencePiece	0.9400

Table 4: Results of Named Entity Recognition

5 Conclusion

In this paper, fine grained analysis of the impact of tokenizer performance on downstream performance of BERT models in Hindi was conducted. The results showed that there is a significant correlation between intrinsic tokenizer performance and extrinsic downstream task performance. The Unigram, Wordpiece and SentencePiece models that showed the best tokenizer performance also showed the best results in Sentiment Analysis as well as in Named-Entity Recognition tasks. This suggests that the quality of words in the models' vocabulary allows it to segment words in the input text more meaningfully, thereby allowing it to learn better semantics during the pre-training phase and subsequently when being fine-tuned for the downstream tasks.

6 Limitations

This research investigates the effect of several tokenizer algorithms on downstream task performance of the model, specifically for the Hindi language. While the results strongly back the hypothesis, the research is limited in its scope. Due to computation requirements, the tokenizers and models were trained on only a limited subset of the raw corpus, for only a single language. Further, only two downstream tasks were evaluated. Investigation can still be done into the effect of tokenizer vocabulary size as well as the amount training data to form a learning a learning curve. The research can also be extended to more Indian languages, which are morphologically rich and more low-resourced. Further manual evaluations could help to better understand the nuanced analysis as well as the strengths and shortcomings of the tokenizers by observing the types of subwords and splits generated for the input text, especially for morphologically complex languages.

References

Github - ai4bharat/indicnlp_corpus: Description describes the indicnlp corpus and associated datasets. https://github.com/AI4Bharat/indicnlp_corpus. (Accessed on 09/17/2024).

sarvamai/openhathi-7b-hi-v0.1-base · hugging face. https://huggingface.co/sarvamai/OpenHathi-7B-Hi-v0.1-Base. (Accessed on 09/17/2024).

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Schulze Buschhoff, et al. 2023. Tokenizer choice for llm training: Negligible or crucial? *arXiv preprint arXiv:2310.08754*.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M Khapra, and Pratyush Kumar. 2021. Indicbart: A pre-trained model for indic natural language generation. *arXiv preprint arXiv:2109.02903*.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2022. Towards leaving no indic language behind: Building monolingual corpora, benchmark and models for indic languages. *arXiv preprint arXiv:2212.05409*.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. 2020. Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961.

Yiğit Bekir Kaya and A Cüneyd Tantuğ. 2024. Effect of tokenization granularity for turkish large language models. *Intelligent Systems with Applications*, 21:200335.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, et al. 2021. Muril: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

T Kudo. 2018a. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Taku Kudo. 2018b. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

Y Liu. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Arnav Mhaske, Harshit Kedia, Sumanth Doddapaneni, Mitesh M. Khapra, Pratyush Kumar, Rudra Murthy, and Anoop Kunchukuttan. 2023. Naamapadam: A large-scale named entity annotated data for Indic languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10441–10456, Toronto, Canada. Association for Computational Linguistics.

Jenalea Rajab. 2022. Effect of tokenisation strategies for low-resourced southern african languages. In *3rd Workshop on African Natural Language Processing*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.