Journal of Soft Computing Paradigm (ISSN: 2582-2640)
www.irojournals.com/jscp/

Implications of Tokenizers in BERT Model for Low-Resource Indian Language

N. Venkatesan[1], N. Arulanand[2]

[1]Research Scholar, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, India
[2]Professor, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, India

E-mail: [1]venkatace7@gmail.com, [2]naa.cse@psgtech.ac.in

Abstract

For any deep learning language model, the initial tokens are prepared as a part of the text preparation process, Tokenization. Important de facto models like BERT and GPT de facto utilize WordPiece and Byte Pair Encoding (BPE) as approaches. Tokenization may have a distinct impact on models for low-resource languages, such as the south Indian Dravidian languages, where many words may be produced by adding prefixes and suffixes. In this paper, four tokenizers are compared at various granularity levels, i.e., their outputs range from the tiniest individual letters to words in their most basic form. Using the BERT pretraining process on the Tamil text, these tokenizers as well as the language models are trained. The model is then fine-tuned with numerous parameters adjusted for the improved performance for a subsequent job in Tamil text categorization. The custom-built tokenizer for Tamil text is created and trained with BPE, WordPiece Vocabulary, Unigram, and WordLevel mechanisms and the compared results are presented after the downstream task of Tamil text categorization is performed using the BERT algorithm.

Keywords: Tokenization, WordPiece, Byte Pair Encoding (BPE), Unigram, WordLevel, low resource language, Tamil, BERT

1. Introduction

For deep language models, tokenization is an essential text preparation step. Traditional word embeddings, such as word2vec often employ vocabularies made out of words' surface forms [1]. Deep language models, on the other hand, employ more efficient tokenization algorithms that separate the input text into smaller segments so that uncommon words may still be evaluated. To better understand text semantics, language-based models

DOI: https://doi.org/10.36548/jscp.2022.4.005
Received: 22.11.2022, received in revised form: 20.12.2022, accepted: 07.01.2023, published: 18.01.2023

can benefit from the tokens that represent fundamental semantic units. It is not unexpected that a lot of scholarly work has researched in determining the optimum tokenization technique for different NLP tasks. Deep language models have gained prominence by using the development of masked language modeling based on Transformer architecture [2] to train a multi-purpose understanding of the language with BERT [3] and its variations. The downstream tasks, such as sentiment analysis and named entity recognition, may then be performed by the language models using the previously learned information.

What are the right tokens to use is the main concern of the tokenization process? There are many complex instances in English. Language-specific tokenization problems exist. Therefore, it is necessary to be aware of the document's wording. Short character subsequence can be used as features in classifiers to identify languages, and the majority of low-resource languages have recognizable signature patterns. For low-resource languages like Dravidian languages, where words might have prefixes and suffixes, the effects of tokenization techniques can vary. Additionally, for low-resource languages like Tamil, the effects of various tokenization techniques, such as token representation at various levels from the character level to the word level, are not thoroughly investigated.

The vocabulary size is the total number of distinct tokens needed to train large language models. The chosen tokenization technique would divide any evaluation data into tokens following the taught vocabulary. So, there is a potential of encountering tokens that are unfamiliar or out of the training vocabulary, which means that some of the tokens in the assessment data could not be there. Since unknown tokens are translated to the same encoding without semantic context, there is a risk that performance may suffer [4]. The problems approached in the paper are to find the impact of different tokenization mechanisms for Tamil text and various other tasks related to it.

2. Background

2.1 De Facto Tokenization Techniques

The most used tokenization method is WordLevel. It separates a text passage into words using a delimiter. The delimiter that is most frequently used is space. It is also possible to divide text using more than one delimiter, such as a space or a punctuation character. Different word-level tokens are created depending on the delimiter. In the training data, missing terms are quite likely to occur. Word tokens make it impossible for the model to recognize word variants that weren't the part of the training set of information. If the model saw hand and ball in the dataset but saw handball in the final text, it won't be able to recognize the word and it will be given a UNK> token [4].

Punctuation also presents a challenge. It would be inefficient to use distinct tokens for the words let or let's. Having a broad vocabulary will be necessary to ensure that every possible term variation has been considered. This is very common in English but it shouldn’t affect the text of other languages without using much punctuation. The results can explain whether this method will help in tokenizing Tamil text better. Employing Unicode character sets to tokenize text, is helpful for languages where there are no gaps between words. For languages without gaps between words, this tokenization technique is effective and can boost the efficiency of machine translation systems [10].

Word-based tokenization commonly referred to as word tokenization or word segmentation involves separating words of text. A wide vocabulary is required to cover all potential word variations, such as various verb tenses or other spellings of a name, to accomplish this accurately. This is crucial for commonly used or high-resource languages that have a large number of irregular forms. Because it enables them to interpret the meaning of the text, many major language models, like the transformers and their variations, use this kind of tokenization. These models are extremely effective at tasks like language translation and text generation because they are trained on enormous volumes of text and use this data to better understand patterns in the language [2].

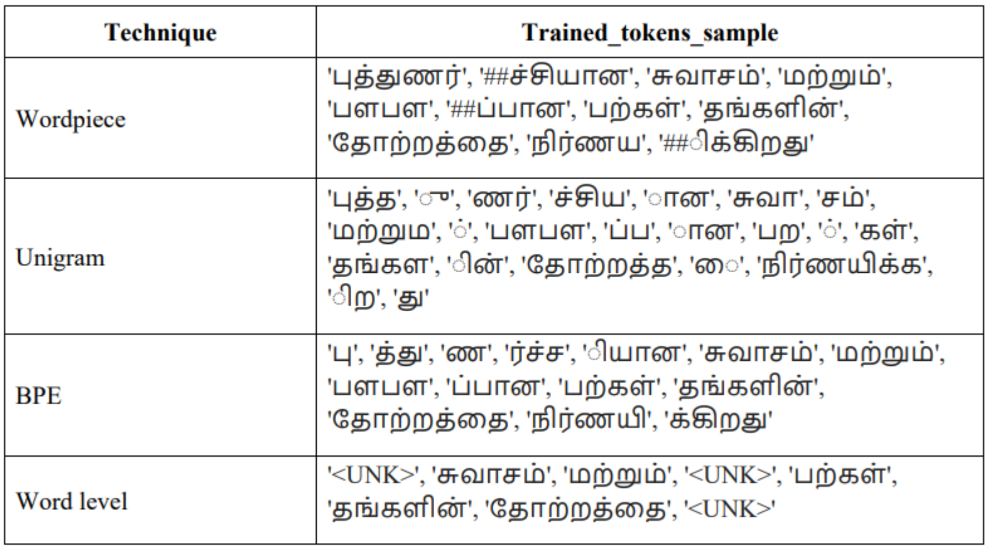
WordPiece [5] and Byte Pair Encoding (BPE) [6] are two current topics of interest in language model pretraining research. Enhancing these subword tokenization techniques is the topic of several notable works in the literature. BPE is determined to be less than ideal for language pretraining since it uses the vocabulary space ineffectively [7]. It is discovered that the inability to represent semantically significant relationships between words is hampered by the vocabulary's frequency-based character combinations. Additionally, representations based on tokenization using word occurrence statistics are dependent on frequency data rather than meanings [8].

Using morphological rules, it is possible to determine words based on Tamil script and to recognize words with inflected forms. A Hidden Markov Model is used to model the order of words in a text, and word boundaries are identified based on the number of incidences of words within the corpus [11].

2.2 Tokenization for Low-Resource Languages

For the tokenized output, BERT always selects the longest in-vocabulary substring from the left at each sub-word unit. Prefixed words are susceptible to bad tokenization, even though this works pretty well for words where the root (or stem) is suffixed [7].

Table 1. Tokenization Method and Trained Tokens



Technique	Trained_tokens_sample
Wordpiece	'புத்துணர்', '##ச்சியான', 'சுவாசம்', 'மற்றும்', 'பளபள', '##ப்பான', 'பற்கள்', 'தங்களின்', 'தோற்றத்தை', 'நிர்ணய', '##ிக்கிறது'
Unigram	'புத்த', 'ு', 'ணர்', 'ச்சிய', 'ான', 'சுவா', 'சம்', 'மற்றும', '்', 'பளபள', 'ப்ப', 'ான', 'பற', '்', 'கள்', 'தங்கள', 'ின்', 'தோற்றத்த', 'ை', 'நிர்ணயிக்க', 'ிற', 'து'
BPE	'பு', 'த்து', 'ண', 'ர்ச்ச', 'ியான', 'சுவாசம்', 'மற்றும்', 'பளபள', 'ப்பான', 'பற்கள்', 'தங்களின்', 'தோற்றத்தை', 'நிர்ணயி', 'க்கிறது'
Word level	'<UNK>', 'சுவாசம்', 'மற்றும்', '<UNK>', 'பற்கள்', 'தங்களின்', 'தோற்றத்தை', '<UNK>'

The sample Tamil text is fed to the individual tokenizers which are already trained with a new set of Tamil corpus and the resulting tokenized text is shown in Table I.

3. Experiments

The effectiveness of tokenization techniques for downstream tasks in Tamil text is shown below with the comparison of tokenization techniques.

3.1 Comparison of Tokenization Techniques

Text classification, also known as text sequence classification, is the process of grouping a collection of news items into a specified set of classifications. Internally developed, specially designed Tamil text categorization datasets that are taken from Wikipedia and tagged appropriately are employed. 2100 news occurrences are produced by combining two datasets using two separate sources.

3.2 Tokenization's Effect on Language Models

How tokenizers that employ BERT perform on downstream Tamil tasks are evaluated and shown in TABLE II. In this experiment, a fixed vocabulary size of 30k tokens is utilized to see how several tokenizers might function in the same environment and the results are shown below.

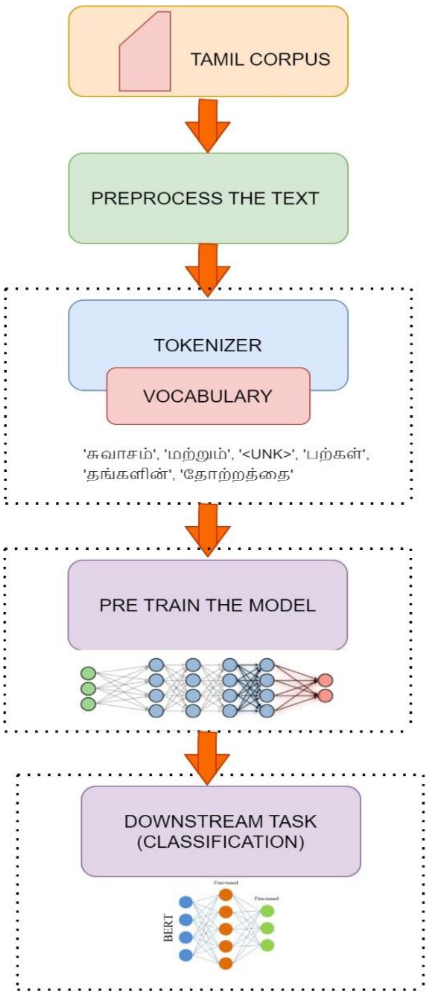


Figure 1. Sequence in Pretraining the Model

Table 2. Fine-Tuning Config for Text Classification Task

Tokenization Technique	**BERT Accuracy (Text Classification)**
WordPiece	**77**
Unigram	**80.4**
BPE	**81**
WordLevel	**72**

3.3 Experimental Results

The key findings and solutions to the problem statement may be summed up as follows. Language modeling frequently uses the mandated industry standards for tokenizers WordPiece and BPE. In most of the reviewed tasks for Tamil, it is found that BPE statistically and substantially outperforms other tokenizers. In Tamil language modeling, the top two tokenizers are BPE and Unigram. Although BPE performs better than WordPiece in Text Classification, the difference between the two is not statistically significant as shown in Figure 2.

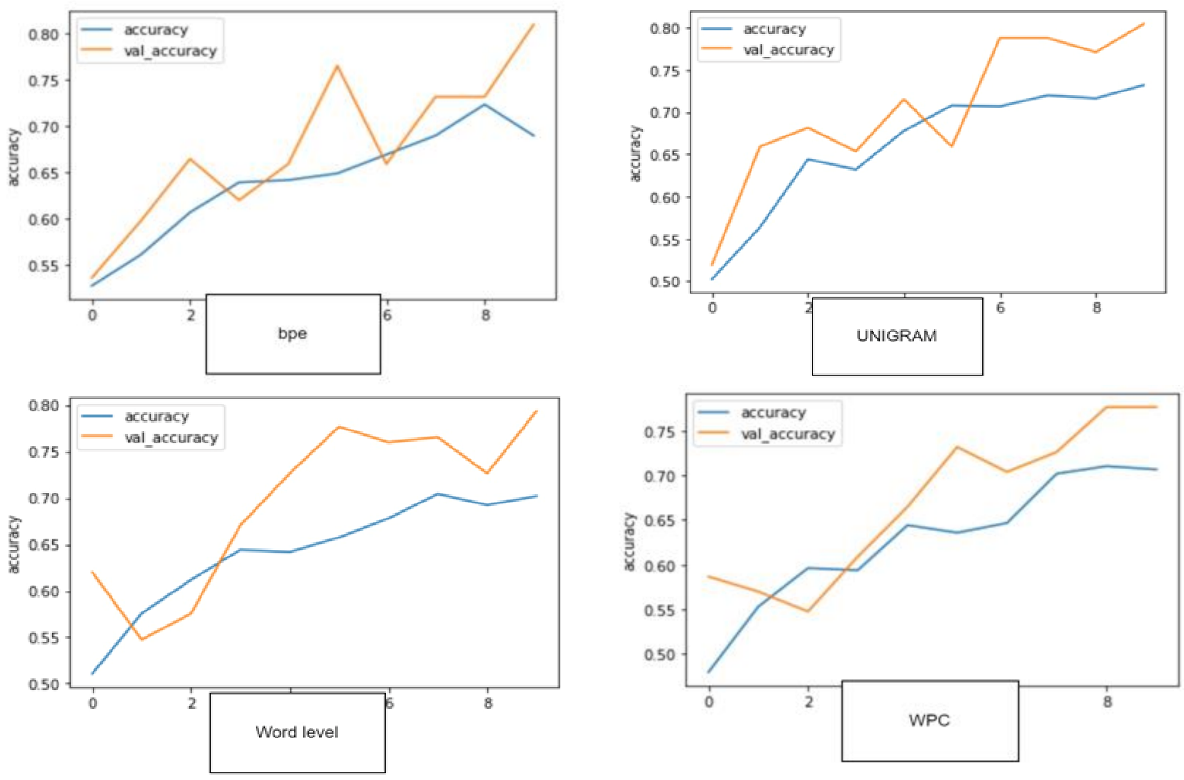


Figure 2. Accuracy graphs of Tokenizers Performance for Tamil Language using BERT

WordLevel tokenizers struggle since there are so many unidentified tokens. Compared to BPE and WordPiece tokenizers, WordLevel tokenizers perform badly, presumably as a result of inefficient use of the available vocabulary. The ability of the word-continued level to achieve comparable results with other tokenizers may be explained by the model's training with the Masked Language Modeling challenge, which develops its ability to infer meaning even with a large number of unknown tokens.

4. Conclusion

An extensive analysis of the consequences of tokenization in Tamil, a low-resource language has been provided in this paper with a few pre-trained deep language models. To accomplish this job, a tokenizer is trained with the mBERT-base-uncased language model using multiple tokenization strategies and varying vocabulary sizes. In future work, this experiment might be extended to other low-resource languages such as Finnish, and

Hungarian and include other tokenization algorithms such as SentencePiece to a relative understanding of the impact of Tokenization [9].

References

[1] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In 1st International Conference on Learning Representations, ICLR, Workshop Track Proceedings. Scottsdale, Arizona, USA.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems. 5998–6008

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[4] Rai, A., Borah, S. (2021). Study of Various Methods for Tokenization. In: Mandal, J., Mukhopadhyay, S., Roy, A. (eds) Applications of Internet of Things. Lecture Notes in Networks and Systems, vol 137. Springer, Singapore. https://doi.org/10.1007/978-981-15-6198-6_18

[5] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Berlin, Germany, 1715–1725. https://doi.org/10.18653/v1/P16-1162

[6] Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean Voice Search. In 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 5149–5152.

[7] Anmol Nayak, Hariprasad Timmapathini, Karthikeyan Ponnalagu, and Vijendran Gopalan Venkoparao. 2020. Domain Adaptation Challenges of BERT in Tokenization and Sub-word Representations of Out-of-Vocabulary words. In Proceedings of the First Workshop on Insights from Negative Results in NLP. Association for Computational Linguistics, Online, 1–5. https://doi.org/10.18653/v1/2020.insights-1.1

[8] Chengyue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2018. Frage: Frequency-agnostic Word Representation. Advances in Neural Information Processing Systems 31 (2018).

[9] Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne, Australia, 66–75

[10] Schuster, Samuel, et al. "Unicode-aware tokenization and text normalization for NLP in low-resource languages." Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). 2018.

[11] Rajeswari R. L., Ramakrishnan K. R. and Srinivasan R. (2010) Tokenizing and Stemming Tamil Text. In: Kinshuk D., Tsai C., Chen N., Huang T. (eds) Emerging Research in Web Information Systems and Mining. ICWISM 2010. Communications in Computer and Information Science, vol 93. Springer, Berlin, Heidelberg

Author's biography

N. Venkatesan is a research scholar who specializes in Natural Language Processing (NLP). With several years of teaching experience, he is currently focusing his research on low-resource languages. This includes studying and developing techniques for processing and understanding languages that have limited available resources, such as datasets and lexical resources. He is also working on developing NLP tools and technologies that can be used to help people who speak low-resource languages, to access information and communicate more effectively.

N. Arulanand is a professor who has a diverse background of industry experience, teaching experience, and research publications. He has experience in Embedded Systems, IoT, Machine Learning and Image Processing, and has spent a significant amount of time researching and developing new technologies and techniques related to these fields, which can be applied to various industries. He has a strong track record of publishing research papers in these fields, which demonstrates his expertise and knowledge in the area. With his industrial experience, he brings a unique perspective to teaching, with a focus on practical applications of these technologies.