A Comparative Study on Language Models for Dravidian Languages

Rahul Raman, Danish Mohammed Ebadulla, Hridhay Kiran Shetty, and Mamatha H.R.

Abstract We train embeddings for four Dravidian languages, a family of languages spoken by the people of South India. The embeddings are trained using the latest deep learning language models, to successfully encode semantic properties of words. We demonstrate the effect of vocabulary size on word similarity and model performance. We evaluate our models on the downstream task of text classification and small custom similarity tasks. Our best model attains accuracy on par with the current state of the art while being only a fraction of its size. Our models are released on the popular open-source platform HuggingFace. We hope that by publicly releasing our trained models, we will help in accelerating research and easing the effort involved in training embeddings for downstream tasks.

1 Introduction

Distributed representation is the foundation of natural language processing, as advances in language modelling serve as a stepping stone for many NLP tasks. Popular domains like text classification, text generation, translation, sentiment analysis, NER, etc. can be advanced with access to contextualized word embeddings. A rise in quality of embeddings is synonymous with an improvement in downstream NLP tasks.

India is a diverse and rapidly growing country. With advances in technology, electronic devices are making their way into the hands of every citizen of the country, giving them the ability to access information that was previously out of reach for them. But this also presents another problem. India has over 22 official languages and several thousand more languages and dialects. It is of paramount importance that we develop NLP tools that bridge this gap and help India progress faster.

R. Raman (✉) · D. M. Ebadulla · H. K. Shetty · Mamatha H.R.
PES University, Bangalore, India
e-mail: mamathahr@pes.edu

M. Abbas (ed.), *Analysis and Application of Natural Language and Speech Processing*, Signals and Communication Technology,
https://doi.org/10.1007/978-3-031-11035-1_8

Indian languages are considered resource poor and have very little monolingual corpora that are publicly available for NLP tasks. Dravidian languages in particular are far behind Indo-Aryan languages. With access to such few resources, training a language model is very challenging, as it is very easy to overfit your model and lose its ability to generalize. Many corpora are also domain specific, making it difficult for the model to generalize context.

In this paper, we experiment with the latest language models on four Dravidian languages: Kannada, Tamil, Telugu and Malayalam. We first train word embedding models and run word similarity tests to evaluate the effect of vocabulary size on model performance and word choice. We then train contextual embedding models on all four languages and evaluate these models on the news article classification provided by indicNLP.[1] We show that lightweight transformer-based models such as RoBERTa [13], DeBERTa [6] and ELECTRA[3] outperform previously used mainstream models. We release these models on the popular transformer's open-source repository HuggingFace[2] where our fine-tuned models, capable of generating quality word embeddings, will significantly improve all Kannada language downstream tasks.

2 Related Work

One of the earliest papers to perform embedding generation on Kannada at scale was fastText by Meta [2]. They proposed an improvised approach for the skip-gram model, representing each word as a bag of character n-grams. This overcame the main drawback in Word2Vec [14], where words were considered as atomic units leading to subpar performance on morphologically rich languages such as Kannada. FastText's embeddings are used as a benchmark for comparison of results in several Indic language model papers.

Kunchukuttan et al. [12] released the indicNLP corpus in 2020, a monolingual corpora for ten Indian languages sourced from various domains and sites. Word embeddings trained on fastText using this corpus were also released. A news classification dataset to be used as a downstream evaluation task was also released. Their embeddings were compared against the original fastText embeddings and were found to outperform the latter in several languages.

Gaurav Arora [1] released the Natural Language Toolkit for Indic languages a few months later, which also released embeddings for 13 Indic languages that outperformed indicNLP and fastText. ULMFiT [7] and Transformer-XL [5] were used to train the embeddings, and the data sourced from Wikipedia was only a fraction of the indicNLP corpora's size. A two-step augmentation technique was used to improve the performance of their models. Kumar Saurav et al. [11] also

[1] https://github.com/AI4Bharat/indicnlp_corpus#indicnlp-news-article-classification-dataset.

[2] https://huggingface.co.

released word embeddings for 14 Indian languages in a single repository, although their results are not competitive with Anoop Kunchukuttam or Gaurav Arora. They trained their embeddings on several transformer architectures such as BERT [9] and ELMo [15] and tested them on several custom tasks.

Kakwani et al. [8] presented the IndicNLPSuite, a collection of large-scale, general-domain, sentence-level corpora of 8.9 billion words across 11 Indian languages, along with pre-trained models and NLU benchmarks available to the public. Yinhan Liu et al. [13] carefully studied the impact of various hyperparameters in pre-training BERT models and released and improved procedure – RoBERTa. Pengcheng He et al. [6] released a model called DeBERTa that improves on BERT and RoBERTa using a disentangled attention mechanism and an enhanced mask decoder. As an alternative to masked language models like BERT, Kevin Clark et al. [3] suggested a discriminative model leveraging replaced token detection called ELECTRA which was shown to be more efficient than BERT particularly for smaller models.

3 Methodology

The following section elaborates on the data pipeline, the preprocessing steps and the experimental setup of this work.

3.1 Dataset

Our pre-training data is sourced from the indicCORP [8], a collection of ten Indic languages. We use a small subset of the datasets available to prove that our models can perform in resource-constrained situations. We use the news classification released by indicNLP for the downstream task of text classification. We also build small custom datasets for word similarity and word analogy tests. To ensure fair comparison for downstream tasks across all models for a particular language, we train all our models on the same corpora.

3.2 Preprocessing

The corpora were cleaned to remove any foreign tokens and fix formatting errors. Shuffling and deduplication were applied after extracting the data subset. An md5 hash was applied to deduplicate the corpora, leaving us with roughly four to five million sentences per language after it was applied on the corpora. To make initial

Table 1 Dataset statistics. Pre-training data is the indicCORP subset used to pre-train our models. News classification data is the indicNLP news category classification dataset

Language	Pre-training	News classification data	
		Train	Test
Kannada	4.07M	24,000	2400
Telugu	4.82M	19,000	2400
Tamil	5.16M	5346	669
Malayalam	5.85M	5036	630

training easier, any sentences greater than 30 words or having English in more than 30% of the sentence were removed. The exact statistics of each language's dataset are given in Table 1.

3.3 Tokenization and Vocabulary

Our word embedding models are tokenized using SentencePiece [10] with varying vocabulary sizes. The RoBERTa and DeBERTa models use the ByteBPE tokenizer, and ELECTRA uses BertWordPiece. With the help of SentencePiece API,[3] tokens were generated by experimenting with the hyperparameters. Vocabulary size ranged from 8000 to 32,000 with incremental steps of 4000. BertWordPiece and ByteBPE were trained to generate a vocabulary size of 32,000 with words having a minimum frequency of 4.

Previous works claim that higher vocabulary sizes correspond to a lower chance of out-of-vocabulary words occurring, and this usually translates to better performance in downstream tasks. But without a morphologically motivated technique to segment subwords, increasing the vocabulary size might lead to an increased occurrence of different inflections of the same word. Hence, we decide to compare varying vocabulary sizes and their performance.

3.4 Experimental Setup

We evaluate our models on the downstream task of text classification using the indicNLP and iNLTK news classification dataset. All information relevant to the datasets is tabulated in Table 1. All models were trained using a single 12GB NVIDIA Tesla K80 GPU.

[3] https://github.com/google/sentencepiece.

4 Models and Evaluation

The following section covers the models we used starting from word embedding models and going up to contextual embeddings. It covers their architecture and the downstream task setup.

4.1 Word Embedding Models

Our word embedding models are trained using the fastText API.[4] The publicly released language model has an approximate vocabulary size of 1.7 million. With the API, we pre-trained a fastText model from scratch with both CBOW and skip-gram architecture. The fastText API takes its input directly and handles the tokenization. Due to very few hyperparameters provided by the fastText API for tuning the model, further experimentation was done with the gensim API. With the gensim API, first the input data was tokenized with SentencePiece. The API provides hyperparameters for tokenization, vocabulary frequency and the architecture which helped us fine-tune our model for better accuracy in the news classification dataset. The API's supervised module was used to perform text classification on the news dataset.

4.2 Contextual Embedding Models

We train a different language model for each language and use three BERT-based architectures: RoBERTa, DeBERTa and ELECTRA. The following subsections will cover these models in more detail.

4.2.1 RoBERTa

Since base BERT models require a large corpus and access to heavy computation resources, we trained embeddings on a RoBERTa model with distilBERT's [16] configuration. When compared to the BERT pre-training technique, one of the key aspects of the design feature in the RoBERTa model is the removal of the next sentence prediction objective from the pre-training phase and the addition of dynamic masking for the training data, which has shown a significant improvement in performance.

[4] https://github.com/facebookresearch/fastText.

Our model was trained using the HuggingFace API. Byte Pair Encoding [17] was used to tokenize the corpus after which the tokenizer weights were transferred to the RoBERTa tokenizer. The vocabulary size was set to 32,000, and the model's configuration was set to 6 hidden layers, 12 attention heads and 768 embedding size. The size of the model was 68 M parameters. After the pre-training phase, two linear layers were added to fine-tune the model on the classification task. We pre-trained the model for 300,000 steps and stopped the model when loss flattened out. Hyperparameters such as batch size, hidden layers, number of attention layers and the embedding size were tuned to accommodate the decreased model size.

4.2.2 DeBERTa

In transformers, the input word vector for the multi-head attention mechanism is a mathematical combination of the word embedding and positional encoding. In absolute positional encoding, used in language models like BERT and RoBERTa, each token will have its own positional encoding vector. But in relative positional embedding, each token will have 'n' (size of the tokenized sentence) positional vectors indicating the positional relation between the current token and other tokens, and these positional vectors are shared amongst all the tokens.

The DeBERTa architecture utilizes relative positional encoding and incorporates two techniques, disentangled attention mechanism and enhanced mask decoder. Disentangled attention mechanism computes the attention weights using disentangled matrices on the word and positional encodings instead of mathematically combining word and positional encodings. The enhanced mask decoder incorporates the absolute positional embedding at the last layer to address the disambiguation relation between the generated word and the context.

The DeBERTa model was also trained using the HuggingFace API with Byte Pair Encoding for tokenization. The vocabulary size was set to 32,000. We used the DeBERTa v2 model with 6 hidden layers, 12 attention heads and an embedding size of 768. The final model had 75M parameters. Pre-training steps of the model are given in Table 2. Two additional linear layers were added to the model during the classification tests.

Table 2 Number of pre-training steps for all our language models

Model	Kannada	Tamil	Telugu	Malayalam
RoBERTa	330K	360K	280K	210K
ELECTRA	200K	200K	200K	200K
DeBERTa	210K	200K	200K	200K

4.2.3 ELECTRA

We also trained embeddings using one of Google research's newer models, ELECTRA. Unlike the RoBERTa and DeBERTa models, which were pre-trained with masked language model task, the ELECTRA model was trained with replaced token detection task. The architecture setup for pre-training comprises of two components: a generator and a discriminator. During the pre-training task, the ELECTRA model predicts whether the sentence has been generated by the BERT model or if the sentence is from the dataset.

The efficiency gains by replacing the token detection approach are due to the loss being defined over all tokens rather than just the mask token (which is the case for MLM) and because there is no masked token discrepancy between pre-training and fine-tuning phases.

Since ELECTRA generates `tf.pretrain` records of the input corpora and stores them offline, it is not limited by memory and is capable of training on large datasets. The model uses the BertWordPiece tokenizer. The vocabulary size was set to 32,000. We used the 'small' version of the model which has 14 M parameters and trained it for 200,000 steps. Maximum sequence length was set to 512. After pre-training the model, it was fine-tuned and evaluated on a text classification task using the ktrain library on the news article dataset.

5 Results

5.1 Word Similarity

We found that our fastText models trained with a vocabulary size of 8,000 had more meaningful similar word predictions compared to the same models with a 32,000 vocabulary size. As a baseline, we also present results on a simple Word2Vec model trained on the same data. Our fastText model's accuracy was marginally lower than the original fastText model. Figure 1 shows some notable results from our experiments on word similarity. Word similarity results were largely comparable across all languages; hence, Fig. 1 only shows the results for the Kannada language. Word2Vec results were observed to be heavily influenced by the domain of the dataset and contained pronouns in word similarity results as it considers word as the atomic token value. In comparison, fastText produces significantly better results at it considers the n-gram characters' information as an atomic unit.

We can also observe that the lower vocabulary models produce words that are synonyms of the input word, while the large vocabulary models produce inflections of the same word. The official fastText model had very different words at the morpheme level, but these words were distinctly similar to the actual word.

Model	Word Similarity					
	ರಾಜ (king)			ಮನುಷ್ಯ (man)		
FastText_R - CBOW	ರಾಜನಾದ Rājanāda 'The king'	ಪ್ರವಾದಿ Pravādi 'Prophet'	997ರಲ್ಲಿ 997Ralli 'In 997'	ಫರಿಸಾಯನು Pharisāyanu 'The Pharisee'	ಪುತ್ರನೇ Putranē 'Son'	ಮನುಷ್ಯನನ್ನು Manuṣyanannu 'The man'
FastText_R - SG	ದಾವೀದ Dāvīda 'David'	ಸೊಲೊಮೋನ Solomōna 'Solomon'	ಅಮಚ್ಯನು Amājiyā 'Amaziah'	ಬಿದ್ದುಹೋಗುವದು Bidduhōguvadu 'To fall off'	ಮನುಷ್ಯನ Manuṣyana 'Man's'	ಇಸ್ರಾಯೇಲನಿಗೆ Isrāyēlanige 'For the Israelites'
FastText - inltk	ರಾಜನ Rājana 'King's'	ರಾಜ್ Rāj 'Raj'	ರಾಜಕೀಯ Rājakīya 'Politics'	ಮನುಷ್ಯರ Manuṣyara 'Human beings'	ಭಗವಂತ Bhagavanta 'Lord'	ದೆವ್ವ Devva 'Devil'
FastText_G - 8K	ಅರಸ Arasa 'King'	ಅರಸನಾದ Arasanāda 'The king'	ಕುಮಾರ Kumāra 'Son'	ಪುರುಷನು Puruṣanu 'The man'	ಪುರುಷ Puruṣa 'Male'	ಮನುಷ್ಯನಿಂದ Manuṣyaninda 'By man'
FastText_G - 16K	ರಾಮ Rāma 'Rama'	ಪ್ರವಾದಿ Pravādi 'Prophet'	ಅರಸ Arasa 'King'	ಕುಮಾರನು Kumāranu 'Son'	ಸ್ತ್ರೀಗೆ Strīge 'Female'	ಹುಡುಗ Huḍuga 'Boy'
FastText_G - 32K	ನಾಥ Nātha 'Nath'	ನಾಟ Nāṭa 'Nata'	ರಾಜನ Rājana 'King's'	ಹುಡುಗಿಯ Huḍugiya 'Girl'	ಮನುಷ್ಯನಿಂದ Manuṣyaninda 'By man'	ಫರಿಸಾಯನು Pharisāyanu 'The Pharisee'
FastText - pretrained	ಸಿಂಘನನು Siṅghananu 'Lioness'	ರಾಣಿ Rāṇi 'Queen'	ಎನುತಲೆ Enutali 'Chattering'	ಪ್ರವಾದಿಯೂ Pravādiyū 'Prophetic'	ಪುತ್ರನೇ Putranē 'Son'	ಮನೆಯವರೇ Maneyavarē 'Housekeeper'

Fig. 1 *Word similarity*. *FastText_R*: fastText's Meta Research implementation with character-level embeddings. *FastText_G*: is the gensim implementation which takes in SentencePiece embeddings

Table 3 News article classification results. FT models are all fastText models trained by Kakwani et al. [8]. Our models are italicized

Model	Kannada	Tamil	Telugu	Malayalam
FT-W	95.93	95.99	98.67	89.02
FT-WC	96.53	95.90	98.08	89.18
IndicFT	97.43	97.26	99.17	92.83
RoBERTa	**98.30**	95.36	99.16	94.76
ELECTRA	97.43	91.47	97.29	89.36
DeBERTa	97.96	93.87	99.16	93.01
indicBERT base	97.87	96.60	99.67	93.33
indicBERT large	97.87	95.24	**99.67**	85.33
indicNLP	97.20	97.01	98.79	92.50
XLM-R	97.60	**97.28**	99.33	**96.00**
mBERT	97.87	94.56	98.67	81.33

The bold values indicate the best performing model

5.2 News Article Classification

Our models are compared against Meta Research's fastText model trained on Wikipedia and Commmon Crawl, fastText models trained by Kakwani et al. [8], indicNLP, indicCORP and large BERT-based models like XLM-R [4] and mBERT on a text classification task using the indicNLP news classification dataset. The results are documented in Table 3.

All three of our contextual embedding models manage to outperform the fastText models. Of our three language models, RoBERTa performs the best, especially in Kannada where it manages to outperform the previous state-of-the-art models as well with a classification accuracy of 98.30%.

The ELECTRA model managed to keep up with the other models despite being considerably smaller than them. Our ELECTRA model is built using the 'small' version with 14 M parameters and was fine-tuned on the text classification task after pre-training for 200,000 steps. The accuracy it obtains is only marginally lower than the other models' accuracy on the same task, despite having a fraction of the parameters. All our models were trained on lesser data and with smaller parameters but still managed to deliver performance comparable to the state of the art. This proves that with the right tokenization and hyperparameter choices, we can overcome the morphological richness of Indic languages and build compact models that can deliver a high level of performance on downstream NLP tasks.

6 Conclusion

In this work, we present a detailed comparative study of language models and their performance on the agglutinative languages of South India. We start our comparison with basic word embedding models like Word2Vec and fastText and build our way up to the latest contextual embedding models like RoBERTa and ELECTRA. We explore the effect of vocabulary size on language models when we use subword segmentation techniques on our corpus. We show that larger vocabulary sizes correspond to the models choosing inflections of the original word in similarity tasks. We also train BERT-based lightweight models like RoBERTa, DeBERTa and ELECTRA and compare them against other state-of-the-art Indic language models. Our models perform on par with their much larger counterparts, and our RoBERTa model achieves the best performance on the news classification task, beating larger models like XLM-R and mBERT. All our models are released on HuggingFace[5] for the open research community to experiment with.

7 Future Work

Future work will involve training contextual embedding models on all Indic languages and uploading them on a popular site like HuggingFace. The models used in this work have proven to be a competitive and efficient choice to develop language models for Indic languages by achieving accuracy on par with much larger and compute-intensive BERT models. BERT-based models have proved to be superior

[5] https://huggingface.co/RahulRaman.

to the previously utilized mainstream models like Word2Vec and fastText. With more training and fine-tuning, lightweight BERT models might even be able to outperform their mainstream counterparts in low-resource settings.

We believe that the vocabulary size dilemma can be overcome by using a linguistically motivated subword segmentation technique like Morfessor.[6] This will help us identify frequently occurring suffixes and eliminate the occurrence of inflections in the vocabulary.

References

1. Arora, G.: inltk: Natural language toolkit for indic languages. In: Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS), pp. 66–71 (2020)
2. Bojanowski, P., Grave, É., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)
3. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: Electra: Pre-training text encoders as discriminators rather than generators. In: International Conference on Learning Representations (2019)
4. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. In: ACL (2020)
5. Dai, Z., Yang, Z., Yang, Y., Carbonell, J.G., Le, Q., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2978–2988 (2019)
6. He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention. In: International Conference on Learning Representations (2020)
7. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 328–339. Association for Computational Linguistics, Melbourne, Australia (2018). https://doi.org/10.18653/v1/P18-1031. https://aclanthology.org/P18-1031
8. Kakwani, D., Kunchukuttan, A., Golla, S., Gokul, N., Bhattacharyya, A., Khapra, M.M., Kumar, P.: Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 4948–4961 (2020)
9. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT, pp. 4171–4186 (2019)
10. Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 66–71 (2018)
11. Kumar, S., Kumar, S., Kanojia, D., Bhattacharyya, P.: "a passage to India": Pre-trained word embeddings for Indian languages. In: Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), pp. 352–357. European Language Resources association, Marseille, France (2020). https://aclanthology.org/2020.sltu-1.49
12. Kunchukuttan, A., Kakwani, D., Golla, S., N.C., G., Bhattacharyya, A., Khapra, M.M., Kumar, P.: Ai4bharat-indicnlp corpus: Monolingual corpora and word embeddings for indic languages.

[6] https://github.com/aalto-speech/morfessor.

Preprint (2020). arXiv:2005.00085
13. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. Preprint (2019). arXiv:1907.11692
14. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR (2013)
15. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations (2018)
16. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. Preprint (2019). arXiv:1910.01108
17. Sennrich, R., Haddow, B., Birch, A.: 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), pp. 1715–1725 (2016)