A Comparative Study on Language Models for the Kannada Language

Danish Ebadulla , Rahul Raman , Hridhay Kiran Shetty , Mamatha H.R.
PES University Bangalore, India
{danishebadulla, rahulraman, hridhayshetty}@pesu.pes.edu
mamathahr@pes.edu

Abstract

We train word embeddings for Kannada, a Dravidian language spoken by the people of Karnataka, a southern state in India. The word embeddings are trained using the latest deep learning language models, to successfully encode semantic properties of words. We release our best models on HuggingFace, a popular open source repository of language models to be used for further Indic NLP research. We evaluate our models on the downstream task of text classification and small custom analogy and similarity tasks. Our best model attains accuracy on par with the current State of the Art while being only a fraction of its size. 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 NLP, 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. 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.

Indian languages are considered resource poor and have very little monolingual corpora that is 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 generalise. Many corpora are also domain specific, making it difficult for the model to generalise context.

In this paper, we experiment with the latest language models on the Kannada language. Using the monolingual corpora provided by indicNLP[1] we have trained the models from scratch and fine-tuned them. We evaluate these models on the news dataset classification provided by indicNLP[2]. We also perform some custom word similarity and analogy tests on the generated embeddings. We show that lightweight transformer based models such as RoBERTa (Liu et al., 2019) and ELECTRA(Clark et al., 2020) outperform previously used mainstream models. We release these models on the popular transformers open source repository HuggingFace[3] 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 Facebook (Bojanowski et al., 2016). 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 (Mikolov et al., 2013), 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

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

Indic language model papers.

Anoop Kunchukuttan et al. (Kunchukuttan et al., 2020) released the indicNLP corpus in 2020, a monolingual corpora for 10 Indian languages sourced from various domains and sites. Word embeddings trained on FastText using this corpora 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 (Arora, 2020) 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 (Howard and Ruder, 2018) and TransformerXL (Dai et al., 2019) were used to train the embeddings and the data sourced from Wikipedia was only a fraction of the indicNLP corpora's size. A 2 step augmentation technique was used to improve the performance of their models. Kumar Saurav et al. (Kumar et al., 2020) also 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 (Devlin et al., 2018) and ELMo (Peters et al., 2018) and tested them on several custom tasks.

3 Methodology

3.1 Dataset

Our pre-training data is sourced from the indicNLP monolingual corpora, a collection of 10 Indic languages and the iNLTK monolingual corpora. The indicNLP Kannada corpora has 14 million sentences and 174 million tokens. The iNLTK[4] corpora has 26,000 sentences sourced from Wikipedia articles. 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, we train all our models on the same corpora.

3.2 Preprocessing

The corpora was cleaned to remove any foreign tokens and fix formatting errors. Shuffling and deduplication was applied after combining all the corpora sources. An md5 hash was applied to deduplicate the corpora, leaving us with roughly 11 million sentences after it was applied on the corpora. To make initial training easier, any sentences greater than 30 words or having english in more than 30% of the sentence were removed.

3.3 Tokenization and Vocabulary

All our models use either SentencePiece (Kudo and Richardson, 2018) or BertWordPiece for tokenization and vocabulary generation. With the help of Sentence-Piece API [5] tokens were generated by experimenting with the hyper parameters. Vocabulary size ranged from 8,000-32,000 with incremental steps of 4,000. BertWordPiece was trained to generate a vocabulary size of 40,000 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 occuring 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 news classification dataset which has around 24,000 training examples and 3,000 test examples with 3 labels. All models were trained using a single 12GB NVIDIA Tesla K80 GPU.

4 Models and Evaluation

4.1 Word2Vec

We start off with both the Word2Vec models to set an initial benchmark. The tokenization was done with SentencePiece with byte pair encoding algorithm. The model architecture was provided for by the gensim API[6]. The API also provides a simple interface for tuning hyper-parameters.

The CBOW model gave us better results when compared to the skip-gram model in the word similarity test. We noticed that Word2Vec models with lower vocabulary size had more meaningful words in the similarity predictions, with lower vocabulary models predicting synonyms of the input word

[4]https://github.com/goru001/inltk

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

[6]https://radimrehurek.com/gensim

and higher vocabulary models predicting inflected forms of the input word. Some of the similar words in higher vocabulary size models had no meaning by itself, which might probably indicate over tokenization, which might be good for predicting unknown words but results in diminished quality of the word embeddings. We hypothesize that this might be due to the small size of the input corpus.

4.2 FastText

The fastText API[7] was used to train our model. The publicly released Kannada 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 it's input directly and handles the tokenization. Due to very few hyper parameters 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 hyper parameters 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.

4.3 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 (Sanh et al., 2019) configuration using the HuggingFace API. Byte Pair Encoding (Sennrich et al., 2015) 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 2 linear layers were added to fine tune the model on the classification task. As RoBERTa performs in-memory tokenization, due to resource constraints we were unable to train the model on the entire corpora that was used for Word2Vec and fastText. The model was trained for 1 epoch. Hyper parameters such as batch size, hidden layers, number of attention layers and the embedding size were tuned to accommodate the decreased model size.

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

4.4 ELECTRA

We also trained embeddings using one of Google research's newer models, ELECTRA. It is a BERT model that performs Masked Language Modelling using a discriminator instead of a generator. The model is trained to corrupt words with high probability in place of the MASK and the discriminator tries to identify these corrupt words. 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 the entire corpora. The model uses the BertWordPiece tokenizer. The vocabulary size was set to 40,000. We used the 'small' version of the model which has 14M 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 Kannada news articles dataset. The pretrained model has been uploaded to HuggingFace[8] for future use.

5 Results

Our models are compared against Facebook's fastText model trained on Wikipedia and CommmonCrawl, indicNLP and iNLTK on a text classification task using the indicNLP News Classification dataset. The results are documented in Table 1.

We found that our Word2Vec and 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. fastText outperformed Word2Vec models and 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. 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.

[8]https://huggingface.co/DarkWolf/kn-electra-small

Model	Word Similarity					
	ರಾಜ (king)			ಮನುಷ್ಯ (man)		
W2V - cbow	ರಾಜನಾದ Rājanāda 'The king'	ಪ್ರವಾದಿ Pravādi 'Prophet'	997ರಲ್ಲಿ 997Ralli 'In 997'	ಫರಿಸಾಯನು Pharisāyanu 'The Pharisee'	ಪುತ್ರನೇ Putranē 'Son'	ಮನುಷ್ಯನನ್ನು Manuṣyanannu 'The man'
W2V - sg	ದಾವೀದ Dāvīda 'David'	ಸೊಲೊಮೋನ Solomōna 'Solomon'	ಅಮಜ್ಯನು Amājiyā 'Amaziah'	ಬಿದ್ದುಹೋಗುವದು Bidduhōguvadu 'To fall off'	ಮನುಷ್ಯನ Manuṣyana 'Man's'	ಇಸ್ರಾಯೇಲನಿಗೆ Isrāyēlanige 'For the Israelites'
FT - inltk	ರಾಜನ Rājana 'King's'	ರಾಜ್ Rāj 'Raj'	ರಾಜಕೀಯ Rājakīya 'Politics'	ಮನುಷ್ಯರ Manuṣyara 'Human beings'	ಭಗವಂತ Bhagavanta 'Lord'	ದೆವ್ವ Devva 'Devil'
FT - 8K	ಅರಸ Arasa 'King'	ಅರಸನಾದ Arasanāda 'The king'	ಕುಮಾರ Kumāra 'Son'	ಪುರುಷನು Puruṣanu 'The man'	ಪುರುಷ Puruṣa 'Male'	ಮನುಷ್ಯನಿಂದ Manuṣyaninda 'By man'
FT - 16K	ರಾಮ Rāma 'Rama'	ಪ್ರವಾದಿ Pravādi 'Prophet'	ಅರಸ Arasa 'King'	ಕುಮಾರನು Kumāranu 'Son'	ಸ್ತ್ರೀಗೆ Strīge 'Female'	ಹುಡುಗ Huḍuga 'Boy'
FT - 32K	ನಾಥ Nātha 'Nath'	ನಾಟ Nāṭa 'Nata'	ರಾಜನ Rājana 'King's'	ಹುಡುಗಿಯ Huḍugiya 'Girl'	ಮನುಷ್ಯನಿಂದ Manuṣyaninda 'By man'	ಫರಿಸಾಯನು Pharisāyanu 'The Pharisee'
FT - original	ಸಿಂಘನನು Siṅghananu 'Lioness'	ರಾಣಿ Rāṇi 'Queen'	ಎನುತಲೆ Enutali 'Chattering'	ಪ್ರವಾದಿಯೂ Pravādiyū 'Prophetic'	ಪುತ್ರನೇ Putranē 'Son'	ಮನೆಯವರೇ Maneyavarē 'Housekeeper'
FT - sg	ರಾಜನು Rājanu 'The king'	ರಾಜನ Rājana 'King's'	ರಾಜೀ Rājī' 'Rajee'	ಬಿದ್ದುಹೋಗುವದು Bidduhōguvadu 'To fall off'	ಮನುಷ್ಯನಿಗೆ Manuṣyanige 'For the man'	ಇವನು Ivanu 'He'
FT - cbow	ರಾಜನ Rājana 'King's'	ರಾಜನಾದ Rājanāda 'The king'	ರಾಜವೈಭವದ Rājavaibhavada 'Of royalty'	ಮನುಷ್ಯನ Manuṣyana 'Man's'	ರೊಳಗಿಂದ Roḷaginda 'From within'	ಮನುಷ್ಯನಿಂದ Manuṣyaninda 'By man'

Figure 1: **Word Similarity**. W2V - Word2Vec ; FT - fastText ; cbow - continuous bag of words ; sg- skip-gram

The RoBERTa model, despite being trained for only 1 epoch and on only a subset of the corpora, outperforms Word2Vec and fastText in text classification, also managing to beat indicNLP on the same task with a classification accuracy of 97.37%.

Our ELECTRA model is built using the 'small' version with 14M parameters and was fine tuned on the text classification task after pre-training for 200,000 steps. It obtained a classification accuracy of 98.53%. This accuracy is only marginally lower than iNLTK's accuracy on the same task, despite having a fraction of the parameters.

Model Name	Vocab Size	Accuracy
fastText WC	1.7M	96.2%
fastText_32	32K	94.2%
fastText_14	14K	94.2%
fastText _8	8K	94.3%
indicNLP	-	97.2%
RoBERTa	32K	97.37%
ELECTRA	40K	98.53%
iNLTK	25K	98.87%

Table 1: Text classification accuracy. The models in italics are our models while the others are popular Kannada language models for comparison

6 Future Work

Future work will involve training ELECTRA models on all Indic languages. ELECTRA has 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 to the previously utilised 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[9]. This will help us identify frequently occurring suffixes and eliminate the occurrence of inflections in the vocabulary. We also intend to release RoBERTa and ELECTRA pretrained models on HuggingFace for Kannada and other Indic languages to further research on distributed representation.

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

References

Gaurav Arora. 2020. inltk: Natural language toolkit for indic languages. *CoRR*, abs/2009.12534.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jeremy Howard and Sebastian Ruder. 2018. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

Saurav Kumar, Saunack Kumar, Diptesh Kanojia, and Pushpak Bhattacharyya. 2020. "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)*, pages 352–357, Marseille, France. European Language Resources association.

Anoop Kunchukuttan, Divyanshu Kakwani, Satish Golla, Gokul N. C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. Ai4bharat-indicnlp corpus: Monolingual corpora and word embeddings for indic languages. *CoRR*, abs/2005.00085.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

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 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.