NPVec1: Word Embeddings for Nepali - Construction and Evaluation

Pravesh Koirala
Institute of Engineering, Pulchowk Campus
Lalitpur, Nepal
praveshkoirala@gmail.com

Nobal B. Niraula
Nowa Lab
Madison, Alabama, USA
nobal@nowalab.com

Abstract

Word Embedding maps words to vectors of real numbers. It is derived from a large corpus and is known to capture semantic knowledge from the corpus. Word Embedding is a critical component of many state-of-the-art Deep Learning techniques. However, generating good Word Embeddings is a special challenge for low-resource languages such as Nepali due to the unavailability of large text corpus. In this paper, we present *NPVec1* which consists of 25 state-of-art Word Embeddings for Nepali that we have derived from a large corpus using GloVe, Word2Vec, fastText, and BERT. We further provide intrinsic and extrinsic evaluations of these Embeddings using well established metrics and methods. These models are trained using 279 million word tokens and are the largest Embeddings ever trained for Nepali language. Furthermore, we have made these Embeddings publicly available to accelerate the development of Natural Language Processing (NLP) applications in Nepali.

1 Introduction

Recent Deep Learning (DL) techniques provide state-of-the-art performances in almost all Natural Language Processing (NLP) tasks such as Text Classification (Conneau et al., 2016; Yao et al., 2019; Zhou et al., 2015), Question Answering (Peters et al., 2018; Devlin et al., 2018), Named Entity Recognition (Huang et al., 2015; Lample et al., 2016) and Sentiment Analysis (Zhang et al., 2018; Severyn and Moschitti, 2015). DL techniques are attractive due to their capacity of learning complex and intricate features automatically from the raw data (Li et al., 2020). This significantly reduces the required time and effort for feature engineering, a costly step in traditional feature-based approaches which further requires considerable amount of engineering and domain expertise. Thus, DL techniques are very useful for low-resource languages such as Nepali.

Many Deep Learning techniques require Word Embeddings to represent each word by a vector of real numbers. Word Embeddings learn a meaningful representation of words directly from a large unlabeled corpus using co-occurrence statistics (Bojanowski et al., 2017). The closer the word representations to actual meanings, the better the performance. Consequently, Word Embeddings have received special attention from the research community and are predominantly used in current NLP researches.

Word Embeddings can generally be divided into two categories: Context-Independent embeddings such as GloVe (Pennington et al., 2014), Word2Vec (Mikolov et al., 2013), and fastText (Bojanowski et al., 2017), and Context-Dependent embeddings such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) and ELMo (Embeddings from Language Models) (Peters et al., 2018). Context-dependent word embedding is generated for a word as a function of the sentence it occurs in. Thus, it can learn multiple representations for polysemous words (Peters et al., 2018). To learn these deep contextualized representations, BERT uses a transformer based architecture pretrained on Masked Language Modelling and Next Sentence Prediction tasks, whereas, ELMo uses a Bidirectional LSTM architecture for combining both forward and backward language models.

In this paper, we present *NPVec1*, a suite of Word Embedding resources for Nepali, a low-resource language, which is the official language and de-facto lingua franca of Nepal. It is spoken by more than 20 million people mainly in Nepal and many other places in the world including Bhutan, India, and Myanmar (Niraula et al., 2020). Even though Word Embeddings can be directly learned from raw texts in an unsupervised fashion, gathering a large amount of data for its training remains a huge challenge in itself for a low-resource lan-

Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), pages 174–184
Bangkok, Thailand (Online), August 6, 2021. ©2021 Association for Computational Linguistics

guage such as Nepali. In addition, Nepali is a morphologically rich language which has multiple agglutinative suffixes as well as affix inflections and thus proves challenges during its preprocessing i.e. tokenization, normalization and stemming.

We have collected data over many years and combined it with multiple other publicly available data sets to generate a suite of Word Embeddings, i.e. *NPVec1*, using GloVe, Word2Vec, fastText and BERT. It consists of 25 Word Embeddings corresponding to different preprocessing schemes. In addition, we perform the intrinsic and extrinsic evaluations of the generated Word Embeddings using well established methods and metrics. Our pre-trained Embedding models and resources are made publicly available[1] for the acceleration and development of NLP research and application in Nepali language.

The novel contributions of this study are:

- First formal analyses of different Word Embeddings in Nepali language using intrinsic and extrinsic methods.
- First study of effects of preprocessing such as normalization, tokenization and stemming in different Word Embeddings in Nepali language.
- First contextualized word embedding (BERT) generation and evaluation in Nepali language.
- The largest Word2Vec, GloVe, fastText and BERT based Word Embeddings ever trained and made available for Nepali language to date.

The rest of this paper is organized as follows. We review related works in Section 2. We describe the data collection and corpus construction in Section 3. We describe our experiments to develop Word Embedding methods in Section 4. We present model evaluations in Section 5 and conclusion and future directions in Section 6.

2 Related Works

Word Embeddings provide continuous word representations and are the building blocks of many NLP applications. They capture distributional information of words from a large corpora. This information helps the generalization of machine learning models especially when the data set is limited (Mikolov et al., 2017). Word Embedding tools, technologies and pre-trained models are widely available for resource rich languages such as English (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017) and Chinese (Li et al., 2018; Chen et al., 2015). Due to the wide use of Word Embeddings, pre-trained models are increasingly available for resource poor languages such as Portuguese(Hartmann et al., 2017), Arabic (Elrazzaz et al., 2017; Soliman et al., 2017), and Bengali (Ahmad and Amin, 2016).

Most Word Embedding algorithms are unsupervised. Which means that they can be trained for any language as long as the corpus data is available. One such effort is by Grave et al. (2018) who generated and made available word vectors for 157 languages, including Nepali, using Wikipedia and Common Crawl data. The pre-trained models for Skip-gram and CBOW are available at `https://fasttext.cc`. Another useful resource is `http://vectors.nlpl.eu/repository` which is a community repository for Word Embeddings maintained by Language Technology Group at the University of Oslo (Kutuzov et al., 2017). It currently hosts 209 pre-trained word Embeddings for most languages but not Nepali.

Word Embeddings for Nepali are derived in small scale by Grave et al. (2018) using fastText and by Lamsal (2019) using Word2Vec. Both of these efforts have major limitations. First, they have limited diversity in the corpus. Grave et al. use Wikipedia and Common Crawl data while Lamsal uses news corpus. Second, their corpus is very small compared to ours (Section 3). Third, they do not provide any evaluation of the generated models. Fourth, they have done limited or no prepossessing on the data. We show later in Section 3.3 that tokenization and text normalization are critical for processing morphologically rich Nepali text. In contrast, we have conducted a large scale study of Word Embeddings in more diverse and large data sets using GloVe, fastText, and Word2Vec. Our corpus is nearly four times bigger than the corpus used by aforementioned approaches (see Section 3). We have constructed 8 inputs for each combination of binary variables: Tokenization, Normalization and Stemming which has resulted in 24 pre-trained Embeddings for GloVe, Word2Vec, and fastText combined. Additionally, we have trained BERT for one of these preprocess-

[1] `https://github.com/nowalab/nepali-word-embeddings`

ing schemes and performed intrinsic and extrinsic evaluations for each of these 25 models.

3 Corpus Preparation

In this Section, we present our data sources and preprocessing techniques for the corpus. To help readers understand the Nepali words used in this paper, we have provided a gloss in Section 8 with their transliterations and English translations.

3.1 The Corpus

Our corpus consists of a mixture of news, Wikipedia articles, and OSCAR (Ortiz Suárez et al., 2019) corpus. We summarize the data sets in Table 1.

3.1.1 News Corpus

We crawled Nepali online news media over a year and collected more than 700,000 unique news articles ($\sim$ 3GB). As expected, the news articles cover diverse topics including politics, sports, technology, society, and so on. We obtained another news data set from IEEE DataPort (Lamsal, 2020) (1.7GB).

3.1.2 OSCAR Nepali Corpus

We obtained the shuffled data in deduplicated form (1.2GB) for Nepali language from OSCAR (Open Super-large Crawled ALMAnaCH coRpus) (Ortiz Suárez et al., 2019).[2] It is a large multilingual corpus obtained by language classification and filtering of the Common Crawl corpus. Common Crawl[3] is a non-profit organization which collects data through web crawling and makes it publicly available.

3.1.3 Nepali Wikipedia Corpus

We obtained Nepali Wikipedia corpus from Kaggle (Gaurav, 2020). It consists of 39k Wikipedia articles for Nepali (83MB).

3.2 Deduplication

We collected data from multiple sources which might have crawled the same data. Furthermore, there were some boilerplate text in the data. Thus, it was important to remove duplicate texts from the corpus. To remove these duplicates, we followed an approach similar to Grave et al. (2018). With this approach, we computed hash for each sentence and collected the sentence only if the hash was not known before. We were able to remove $\sim$ 22% duplicated sentences from our corpus.

3.3 Preprocessing

After removing duplicates, we discarded sentences with less than 10 characters as they provide little context to learn Word Embeddings. We also removed punctuations and replaced numbers with a special *NN* token. We then applied following *Normalization*, *Tokenization* and *Stemming* preprocessing techniques to derive corpus for the study.

3.3.1 Normalization

Analogous to how there are different cases (lower/upper) in English with no phonetic differences, there are different written vowels sounds in Nepali which, when spoken, are indistinguishable from each other. For example: the two different words नेपाली (Nepali) and नेपालि are spoken the same way even though their written representations differ. Thus, people often mistakenly use multiple written version of the same words which introduces noise in the data set. Normalization, in the context of this study, is identification of all these nuances and mapping them to a same word.

3.3.2 Tokenization

Nepali language has multiple post-positional and agglutinative suffixes like ले, मा, बाट, देखि etc., which can be compounded together with nouns and pronouns to produce new words. For example, the word नेपाली (Nepalese) can be compounded as नेपालीले (Nepalese did), नेपालीहरु (Nepalese+plural), नेपालीको (Of Nepalese), so on and so forth. Thus, these different words can be tokenized as नेपाली + ले , नेपाली + हरु, नेपाली + को which serves to drastically reduce the vocabulary size without the loss of any linguistic functionality. Tokenization, in this context, means the same.

3.3.3 Stemming

In addition, there are also other case markers and bound suffixes that primarily inflect verbs to produce new words. For example, from the same root word खा (eat), words such as खायो (ate), खाँदै (eating), खाएको (had eaten), खाएर (after eating), etc can be constructed. Stemming, in this context, means the reduction of all such inflected words to their base forms.

For the purpose of this study, we have improved upon the preprocessing techniques developed by Koirala and Shakya (2018) for preprocessing (normalizing, tokenizing and stemming) our

[2] https://oscar-corpus.com

[3] https://commoncrawl.org/

Corpus	Tokens	Types	Genre	Description
Our News Corpus	216M	3.3M	News	Online news
Lamsal (Lamsal, 2020)	58.8M	1.2M	News	Online news
OSCAR (Ortiz Suárez et al., 2019)	71.8M	2.2M	Mixed	Mixed Genre
Wikipedia (Gaurav, 2020)	5.1M	0.3M	Mixed	Mixed Genre

Table 1: Corpus Description

Preprocessing Scheme	Code	#Tokens	#Types
Base	(B)	279M	**3.14M**
Base+Normalized	(BN)	279M	2.6M
Base+Normalized+Tokenized	(BNT)	**360M**	1.4M
Base+Normalized+Stemmed	(BNS)	279M	2.04M
Base+Normalized+Tokenized+Stemmed	(BNTS)	359M	**1.09M**
Base+Tokenized	(BT)	357M	1.8M
Base+Tokenized+Stemmed	(BTS)	357M	1.4M
Base+Stemmed	(BS)	279M	2.5M

Table 2: Eight Corpus of NepVec1. Base refers to the raw text.

corpus. Specifically, we generated eight corpus corresponding to different combination of these three preprocessing techniques. The final eight corpus are listed in Table 2.

4 Embedding Methods

4.1 Context-independent Word Embeddings

We chose three state-of-the-art methods for obtaining context-independent Word Embeddings, namely Word2vec, fastText and GloVe. Word embeddings from these methods were learned with the same parameters for fair comparison. We fixed vector dimension to 300 and set minimum word frequency, window size, and the negative sampling size to 5 respectively. Word2vec and fastText models were trained via the Gensim (Řehůřek and Sojka, 2010) implementation using skip-gram method. Whereas, GloVe embeddings were trained via the tool provided by StanfordNLP [4].

4.2 Context-dependent Word Embeddings

We chose BERT to learn context-dependent embeddings. We trained a BERT model using the Huggingface's transformers library (Wolf et al., 2019). BERT model, unlike the other word embedding models, was only trained in one pre-processing scheme i.e. base+normalized+tokenized (BNT)[5] due to resource constraints. Due to the same reason, we reduced both the number of hidden layers and the attention heads to 6 and the hidden dimensions to 300 unlike the original implementation of 12 hidden layers and attention heads and 768 hidden dimensions. The maximum sequence size was chosen to be 512 whereas maximum vocabulary size for the BERT's wordpiece tokenizer was set to 30,000. Our implementation of BERT has 22.5M parameters (in contrast to the 110M parameters of the original implementation i.e. BERT-base) and unlike BERT's original implementation, where it is pre-trained on the task of Masked Language Modelling (MLM) and Next Sentence Prediction, we only pre-trained it for the MLM objective for just a single epoch due to limited computing resources.

5 Evaluation

5.1 Intrinsic Evaluation

Intrinsic evaluation of word embedding models is commonly performed in tasks such as analogies (Grave et al., 2018). There is, however, no such data set available for Nepali language. Thus, we followed the clustering approach suggested in

[4] https://github.com/stanfordnlp/GloVe

[5] Our motivation for training BERT in this scheme was the superior performance of context-independent word embeddings in our intrinsic evaluation task for this particular scheme as per section 5.1

Relatedness Set		Sentiment Set	
Kitchen	Nature	Positive	Negative
रोटि, तरकारी, चिनि, नुन, मसला, अदुवा, लसुन, तेल, मरिच, दाल, थाल, कराइ, भाडो, खोर्सानि, चामल, पिठो, डाडु, पुन्यु, चुल्हो, कचौरा, ग्लास	हिमाल, पहाड, हाइकिङ्ग, ट्रेकिङ्ग, फोटो, जङ्गल, गन्तव्य, खोला, नाला, झरना, गोरेटो, बाटो, घुम्ती, चौतारा, यात्रा, हिउ, हरियाली, देउराली, ताल, उकाली	राम्रो, सस्तो, जाँगरिलो, ठूलो, अग्लो, सफा, हलुको, कोमल, उज्यालो, बुद्धिमान, साँचो, लाभ, निशुल्क, छिटो, सफल, अर्थ, न्याय, सक्षम, धनी	नराम्रो, महगो, पातलो, सानो, होचो, फोहोर, भारी, कठोर, अँध्यारो, अल्छे, मुर्ख, झुठो, हानि, ससुल्क, ढिलो, असफल, अनर्थ, अन्याय, असक्षम, गरिब

Table 3: Data Set for Intrinsic Evaluation of Word Embeddings

(Soliman et al., 2017) which requires a manually constructed data set of terms in different themes (clusters). The goal then is to recover these themes (clusters) using the learned word representations. We constructed following two data sets for the evaluation purposes.

5.1.1 Relatedness Set

This set consisted of twenty one word examples each from two different topics i.e. kitchen and nature. The kitchen topic included words such as चिनि (sugar) नुन (salt) भाडो (pot) etc. whereas, the nature topic included words such as हिमाल (mountain), पहाड (hill), खोला (river) etc. The Relatedness data set is presented in Table 3.

5.1.2 Sentiment Set

This set consisted of nineteen examples each of positive and negative sentiments. The positive sentiment set included words such as राम्रो (good), ठूलो (big), न्याय(justice), etc. whereas the negative sentiment set included their antonyms such as नराम्रो (bad), सानो(small), अन्याय(injustice) etc. The Sentiment data set is presented in Table 3.

Ideally word embeddings should capture both word relatedness and word similarity properties of a word. These two terms are related but are not the same (Niraula et al., 2015; Banjade et al., 2015). For example, chicken and egg are less similar (living vs non-living) but are highly related as they often appear together. Relatedness and Sentiment sets were developed to evaluate the models in these these two aspects.

For each of these cases (sentiment and relatedness), K-Means clustering was applied to the constituent words to generate two clusters (i.e. K=2). The obtained clusters were evaluated using the purity metric which is further elaborated in Section 5.1.3. Since Word2Vec and GloVe cannot handle out-of-vocabulary (OOV) words, unlike fastText and BERT, the average of all corresponding word vectors were used to represent the OOV words.

While Word2Vec, fastText and GloVe models provide a simple word to vector mapping, BERT's learned representations are a bit different and thus, need to be extracted accordingly. For the sake of simplicity, we have averaged the hidden state of the last two hidden layers to get the embeddings for each word token. The words were run without any context.

5.1.3 Purity

The purity metric is an extrinsic cluster evaluation technique (Manning et al., 2008) which requires a gold standard data set. It measures the extent to which a cluster contains homogeneous elements. The purity metric ranges from 0 (bad clustering) to 1 (perfect clustering). Thus, the higher the purity score, the better the results.

5.1.4 Results for Intrinsic Evaluation

The results for the intrinsic evaluations are listed in Table 4. All models performed better in recovering original clusters in the Relatedness Set compared to that of the Sentiment Set i.e. they have higher purity scores in the Relatedness Set than the Sentiment Set. This is expected as semantically opposite words often appear in a very similar context (e.g. This is a *new* model vs. This is an *old* model). Relying on neighboring terms alone would provide little context to capture the semantic meaning of a word. Of all three models, however, GloVe performed the best in the sentiment set by an average of 10% (except in the BNTS scheme). This seem to make it more suitable for tasks such as Sentiment Analyses. Interestingly, BERT model did not perform well compared to other models in the Relatedness set. It, however, provided very competitive score in the Sentiment Set.

Models in the BNT scheme scored highest in

		Intrinsic		**Extrinsic**		
Scheme	**Model**	Purity (Sen)	Purity (Rel)	Precision	Recall	F_1
B	Baseline			0.76	0.68	0.69
	Word2Vec	0.54	0.98	0.80	0.79	0.79
	fastText	0.51	1	0.79	0.78	0.78
	GloVe	0.67	0.95	0.78	0.77	0.77
BN	Baseline			0.77	0.72	0.72
	Word2Vec	0.56	1	0.79	0.78	0.78
	fastText	0.51	1	0.79	0.78	0.78
	GloVe	0.62	0.98	0.78	0.77	0.77
BT	Baseline			0.77	0.72	0.72
	Word2Vec	0.51	0.98	0.78	0.77	0.77
	fastText	0.54	0.98	0.78	0.76	0.76
	GloVe	0.67	1	0.79	0.77	0.77
BS	Baseline			0.76	0.70	0.70
	Word2Vec	0.51	0.93	0.79	0.77	0.77
	fastText	0.54	0.93	0.79	0.78	0.78
	GloVe	0.59	0.93	0.78	0.77	0.77
BNT	Baseline			0.77	0.73	0.73
	Word2Vec	0.54	1	0.76	0.74	0.74
	fastText	0.51	1	0.78	0.76	0.76
	GloVe	0.69	1	0.77	0.76	0.75
	BERT	0.59	0.83	0.77	0.76	0.76
BNS	Baseline			0.77	0.71	0.72
	Word2Vec	0.51	0.95	0.79	0.77	0.77
	fastText	0.51	0.95	0.79	0.78	0.78
	GloVe	0.64	0.95	0.79	0.77	0.77
BTS	Baseline			0.76	0.73	0.74
	Word2Vec	0.51	0.95	0.78	0.76	0.75
	fastText	0.51	0.95	0.78	0.77	0.76
	GloVe	0.62	0.95	0.76	0.73	0.73
BNTS	Baseline			0.76	0.73	0.74
	Word2Vec	0.51	0.95	0.76	0.74	0.74
	fastText	0.54	0.95	0.78	0.76	0.76
	GloVe	0.51	0.95	0.78	0.77	0.77

Table 4: Intrinsic and Extrinsic Results. Sen and Rel refer to Sentiment and Relatedness respectively. Similarly, B=Base i.e. Raw Text, N=Normalized, T=Tokenized, and S=Stemmed.

both of the intrinsic data sets. Purity for relatedness task for all of the three models in this scheme was 1 whereas GloVe model obtained the global best score of 0.69 in the sentiment set in this scheme. In general, it seems that applying the Normalization scheme has a positive effect on model's capacity to learn the representation which makes sense because Normalization reduces differently spelled versions of the same word to a single representation. Purity dropped significantly for all tasks in all schemes that included Stemming. This may be attributed to the possible over-stemming of the words (under-stemming doesn't seem to be a problem because the model is performing well in the Base scheme).

5.2 Extrinsic Evaluation

The primary objective of extrinsic evaluation for this study was to compare how the word embeddings helped generalize the training of other supervised models with very few data labels. For this purpose, a feed-forward neural network architecture was used for a classification objective in a multi-class classification setup.

5.2.1 Data

The data set for classification was derived from a publicly available Github repository i.e. Nepali News Dataset [6]. It consists of Nepali news articles in 10 different categories. Each category has 1000 articles. As mentioned, the goal of extrinsic evaluation here is to see how the learned word representations help the generalization of machine learning model for text classification task when limited training data set is available, a practical scenario for low resource language. If we use large training examples, virtually any classifier would learn to perform better even if the word representations are poor. For this reason, we extracted 3000 samples from the dataset with uniform representation from each categories (i.e. 300 examples each) and further split them randomly into chunks of sizes 10%, 10%, and 80% each. This yielded us examples of sizes 313, 326, and 2361 respectively which were subsequently used for training, validation and testing purposes. Training set had at least 21 examples per class whereas the testing set had at least 227 examples per class. The test set was deliberately chosen to be larger to better estimate the generalization of the classification model across different embedding schemes.

5.2.2 Architecture

We implemented a very simple text classification model using Keras[7]. For each example (news article), we only used the first five hundred tokens and obtained their embedding vectors from the word embedding model under the study. These vectors were then fed to a Keras model where they were first pooled together by a one-dimensional averaging layer and then passed to a hidden layer with 64 units with the ReLU activation and then to the output layer of 10 units with Sigmoid activation. Binary crossentropy function was used to calculate the loss and the model was trained using the Adam Optimizer (Kingma and Ba, 2014) for 60 epochs each. In case of BERT, we averaged the hidden states from the last two hidden layers to get the embeddings, whereas, for getting the baseline results, instead of using any pre-trained word vectors, a trainable Keras embedding layer was used in front of the architecture mentioned above which automatically learns the word embeddings by only using the provided training examples.

5.2.3 Results for Extrinsic Evaluation

Macro Precision, Recall and F_1 metrics were used for the evaluation of the classification model. On average, the F_1 scores for word embedding models exceeded the baseline scores by a margin of 5 percent. This suggests that the use of pre-trained word embeddings helps to generalize classification models better than simply using the embeddings learned from the training set. Interestingly, the global maximum F_1 score was obtained in the Base scheme i.e. with no preprocessing applied, and Normalization seemed to make no difference to the score. This can be attributed to the fact that our data set came from highly reputed newspapers i.e. all word spellings were grammatically correct. We foresee significant increase due to Normalization in data sets such as tweets, social media posts and blogs where grammatical errors are more frequent.

Similarly, Tokenization schemes seemed to drop the classification scores for embedding models but increase the scores for the baseline models in general. This leads us to believe that the representations of the post-positions and agglunitative suffixes, which are the most frequently occurring words in Nepali language, learned by the Word Embedding models may be partial to particular top-

[6] https://github.com/kamalacharya2044/NepaliNewsDataset

[7] https://keras.io

ics. We suggest the omission of post-positions and other frequently occurring words from the data set before using these embeddings in a classification setting.

The standard deviation in the F-scores of Word2Vec model, fastText and GloVe model across the different pre-processing schemes are 2.4%, 1% and 1.4% respectively, which suggests that fastText might be more resilient to problems like over-stemming. We thus recommend the usage of fastText models in applications where it is desirable to stem words.

Interestingly, BERT model, while produced competitive results, did not exceed our expectations on the classification task. We expect a raise in performance of this model if trained in the architecture proposed in its original implementation i.e. 12 attention heads and 12 hidden layers unlike our slimmed down version of 6 attention heads and 6 hidden layers trained for only one epoch. Training on more data and with more epochs are potential future directions to this end.

6 Conclusion and Future Work

In this paper, we trained 25 Word Embedding models for Nepali language with multiple preprocessing schemes and made them publicly available for accelerating NLP research in low-resource language Nepali[8]. This, to our knowledge, is the first formal and large scale study of Word Embeddings in Nepali. We compared the performances of these models using intrinsic and extrinsic evaluation tasks. Our findings clearly indicate that these word embedding models perform exceptionally well in identifying related words compared to discovering semantically similar words. We also suggest that further comparisons be made with an improved stemmer, which has fewer over-stemming error rates than what we've used, to study the effects of over-stemming in word embeddings. Performance of these Word Embeddings in clustering of related words also suggest us that these models will obtain good results in tasks such as Named Entity Recognition and POS Tagging. This is something that we would like to explore in future.

As far as our study with BERT goes, we obviously recommend training the original BERT architecture, rather than what we have used, with more data. For comparison, the original BERT model was trained on a total of 3.3 billion words whereas we've trained our model in just 360 million words. Unfortunately, for a resource poor language like Nepali, this is not a trivial task. Similarly, it would be most interesting to see performances of other context-dependent embedding models such as ELMo, GPT2 (Radford et al., 2019), XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019) in case of Nepali language.

7 Acknowledgments

We would like acknowledge Mr. Ganesh Pandey for his valuable contribution in providing the hardware setup for our experiments. Similarly, we thank Ms. Samiksha Bhattarai and Dr. Diwa Koirala for their continued support and encouragement.

8 Glossary

Original	Transliteration	Meaning
नेपाली	Nepali	Nepalese
रोटि	Roti	Flatbread
तरकारी	Tarkari	Vegetable
चिनि	Cheeni	Sugar
नुन	Noon	Salt
मसला	Masala	Spices
अदुवा	Aduwa	Ginger
लसुन	Lasun	Garlic
तेल	Tel	Oil
मरिच	Marich	Pepper
दाल	Daal	Lentils
थाल	Thaal	Plate
कराइ	Karai	Cooking Pot
भाडो	Bhaado	Utensils
खोर्सानि	Khorsani	Chili
चामल	Chaamal	Rice
पिठो	Peetho	Wheat
डाडु	Daadu	Ladle
पुन्यु	Punyu	Spatula
चुल्हो	Chulho	Stove
कचौरा	Kachaura	Bowl
ग्लास	Glass	Glass
हिमाल	Himal	Mountain
पहाड	Pahad	Hill
हाइकिङ्ग	Hiking	Hiking
ट्रेकिङ्ग	Trekking	Trekking
फोटो	Photo	Photo
जङ्गल	Jungle	Jungle
गन्तव्य	Gantabya	Destination
खोला	Khola	River

[8] https://github.com/nowalab/nepali-word-embeddings

Original	Transliteration	Meaning
नाला	Naala	Rivulets
झरना	Jharana	Waterfall
गोरेटो	Goreto	Trail
बाटो	Baato	Road
घुम्ती	Ghumti	Bend
चौतारा	Chautara	Rest area
यात्रा	Yatra	Travel
हिउ	Hiu	Snow
हरियाली	Hariyali	Greenery
देउराली	Deurali	Hilltop
ताल	Taal	Lake
उकाली	Ukali	Uphill
राम्रो	Ramro	Good
सस्तो	Sasto	Inexpensive
जाँगरिलो	Jagarilo	Energetic
ठूलो	Thulo	Big
अग्लो	Aglo	Tall
सफा	Safaa	Clean
हलुको	Haluko	Lightweight
कोमल	Komal	Soft
उज्यालो	Ujyalo	Bright
बुद्धिमान	Buddhiman	Wise
साँचो	Sacho	Truth
लाभ	Laabh	Gain
निशुल्क	Nisulka	Free
छिटो	Cheeto	Fast
सफल	Safal	Successful
अर्थ	Aartha	Meaning
न्याय	Nyaya	Justice
सक्षम	Sakchyam	Capable
धनी	Dhani	Rich
नराम्रो	Naramro	Bad
महगो	Mahango	Expensive
पातलो	Patalo	Skinny
सानो	Saano	Small
होचो	Hocho	Short
फोहोर	Fohor	Waste
भारी	Bhaari	Heavy
कठोर	Kathor	Hard
अँध्यारो	Adhyaro	Dark
अल्छे	Alche	Lazy
मुर्ख	Murkha	Fool
झुठो	Jhutho	Lies
हानि	Haani	Damage
ससुल्क	Sasulka	Not-Free
ढिलो	Dhilo	Late
असफल	Asafal	Failure
अनर्थ	Anartha	Meaningless
अन्याय	Anyaya	Injustice
असक्षम	Asakchyam	Incompetent
गरिब	Gareeb	Poor

References

Adnan Ahmad and Mohammad Ruhul Amin. 2016. Bengali word embeddings and it's application in solving document classification problem. In *2016 19th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 425–430.

Rajendra Banjade, Nabin Maharjan, Nobal B Niraula, Vasile Rus, and Dipesh Gautam. 2015. Lemon and tea are not similar: Measuring word-to-word similarity by combining different methods. In *International conference on intelligent text processing and computational linguistics*. Springer, 335–346.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146.

Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. 2015. Joint learning of character and word embeddings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2016. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781* (2016).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

Mohammed Elrazzaz, Shady Elbassuoni, Khaled Shaban, and Chadi Helwe. 2017. Methodical evaluation of Arabic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 454–458.

Gaurav. 2020. Nepali Wikipedia Corpus. https://www.kaggle.com/disisbig/nepali-wikipedia-articles

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Nathan Hartmann, Erick Fonseca, Christopher Shulby, Marcos Treviso, Jessica Rodrigues, and Sandra Aluisio. 2017. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. *arXiv preprint arXiv:1708.06025* (2017).

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Pravesh Koirala and Aman Shakya. 2018. A Nepali Rule Based Stemmer and its performance on different NLP applications. In *Proceedings of the 4th International IT Conference on ICT with Smart Computing and 9th National Students' Conference on Information Technology, (NaSCoIT 2018)*. 16–20.

Andrei Kutuzov, Murhaf Fares, Stephan Oepen, and Erik Velldal. 2017. Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In *Proceedings of the 58th Conference on Simulation and Modelling*. Linköping University Electronic Press, 271–276.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* (2016).

Rabindra Lamsal. 2019. 300-Dimensional Word Embeddings for Nepali Language. https://doi.org/10.21227/dz6s-my90

Rabindra Lamsal. 2020. A large scale Nepali text corpus. https://doi.org/10.21227/jxrd-d245

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering* (2020).

Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical Reasoning on Chinese Morphological and Semantic Relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 138–143. http://aclweb.org/anthology/P18-2023

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. *arXiv preprint arXiv:1907.11692* (2019).

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. ISBN 978-0-521-86571-5.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2017. Advances in pre-training distributed word representations. *arXiv preprint arXiv:1712.09405* (2017).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

Nobal B Niraula, Saurab Dulal, and Diwa Koirala. 2020. Linguistic Taboos and Euphemisms in Nepali. *arXiv preprint arXiv:2007.13798* (2020).

Nobal Bikram Niraula, Dipesh Gautam, Rajendra Banjade, Nabin Maharjan, and Vasile Rus. 2015. Combining word representations for measuring word relatedness and similarity. In *The twenty-eighth international flairs conference*.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*, Piotr Bański, Adrien Barbaresi, Hanno Biber, Evelyn Breiteneder, Simon Clematide, Marc Kupietz, Harald Lüngen, and Caroline Iliadi (Eds.). Leibniz-Institut für Deutsche Sprache, Cardiff, United Kingdom. https://doi.org/10.14618/IDS-PUB-9021

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 959–962.

Abu Bakr Soliman, Kareem Eissa, and Samhaa R El-Beltagy. 2017. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science* 117 (2017), 256–265.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019.

Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*. 5753–5763.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 7370–7377.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8, 4 (2018), e1253.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630* (2015).