Efficient Estimation of Nepali Word Representations in Vector Space

Janardan Bhatta[1], Dipesh Shrestha[1], Santosh Nepal[1], Saurav Pandey[1], and Shekhar Koirala[*,1]
[1]Department of Electronics and Computer Engineering, Thapathali Campus, Institute of Engineering, Tribhuvan University, Kathmandu, Nepal

[*]**Corresponding Email**: shekharkoirala4@gmail.com

ABSTRACT

Word representation is a means of representing a word as mathematical entities that can be read, reasoned and manipulated by computational models. The representation is required for input to any new modern data models and in many cases, the accuracy of a model depends on it. In this paper, we analyze various methods of calculating vector space for Nepali words and postulate a word to vector model based on the Skip-gram model with NCE loss capturing syntactic and semantic word relationships.
This is an attempt to implement a paper by Mikolov on Nepali words.

Keywords: CBOW, NCE loss, One Hot Encoding, Skip-gram, TF-IDF, Word2Vec

1. INTRODUCTION

Images are computed based on pixels which are numeric. Similarly, for the text, we need an efficient method of converting words to vectors. For the particular task, a novel approach of creating word2vec is done by [8] but it uses the BOW method which may not fulfill the requirements of state-of-the-art NLP models. There are various methods of representing words in vector space like One hot encoding. In this method, a single bit is set to one and the rest is set to zero. The dimensionality of the vector will be the size of vocabulary: 'n' length of the vocabulary represents n-dimension of the vector space where every word vector is orthogonal to each other, meaning every word is unique. Two words are similar when the vectors representing the words are close. Since every vector is orthogonal, this kind of word representation technique doesn't preserve the semantic property of words.

It is very important to represent the word so that it does not lose the semantic property. The distributional hypothesis in linguistics is derived from the semantic theory of language usages to keep the semantic property i.e,

- Words that are used and occur in the same contexts tend to purport similar meanings. [Harris, Z(1954). Distributional Structure]
- A word is characterized by the company it keeps. [Firth, J.R(1957). A synopsis of linguistic theory (1950-1955)]

This technique is based on the concept of the Co-occurrence Matrix. Co-occurrence can be interpreted as an indicator of semantic proximity of the word. Word2vec captures the semantic implicitly whereas

the glove is used to represent explicitly. The problem with the glove and new models like Fast Text from facebook is, these libraries are tuned based on English language and give poor results when implemented on Nepali Dataset. The advantage of using custom neural network with custom pipeline for text data is to tune the data according to general or domain use cases.

2. METHODOLOGY

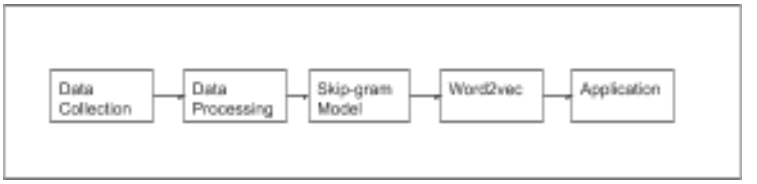

Figure 1: Basic Workflow

2.1 DATA COLLECTION

The source of data is online portals. Scraping and collecting the data from different Nepali news portals, Nepali tweets, and Nepali wiktionary. These data are raw and have to be cleaned.

2.2 DATA PREPARATION

Cleaning the data is very essential such that the system learns well. The raw sentence contains different punctuation, numbers, and symbols. These are to be removed from the sentence. Tokenization is a method to separate the sentence or word based on the symbol. We have tokenized sentences from the group of sentences and then tokenized words from the sentence.

2.3 WORD REPRESENTATION MODEL

2.3.1 TF-IDF

Term Frequency - Inverse Document Frequency is a numerical statistic which is the product of term frequency and inverse document frequency. Term frequency is a measure of the count of a particular term in the document. Let term frequency be denoted by f(t,d) where t is the term and d is the document. Similarly, inverse document frequency is a measure of how much information the word provides whether it is rare or common. It is a logarithmic scale of the fraction of the total number of documents 'N' by the number of documents the term appeared 'n' and denoted as:

$$idf = log(\frac{N}{n}) \quad (1)$$

Then,

$$tfidf = f_{(t,d)} * idf \quad (2)$$

Since it is the statistic of the product of frequency and measure of the rarity of word/term, it doesn't carry any context or semantic property of the word with it. Our neural network-based model out-forms the tf idf model. The complex grammatical structure of Nepali language and many variations of a single word like 'दे', 'दिनुस', 'दिनुहोस', 'दिस्यो', 'दिबक्सियोस'.

This limits us to use stemmer on Nepali language and tfidf could not capture the real semantic of these words as to how these words are in the real world.

2.3.2. CBOW MODEL

In this architecture, the one-hot representation of context words is input to the model and the probability of a center word is the output.

Our implementation of CBOW does not express the meaning of a word in the vector. Rare words have deep meaning and are hard to categorize. The model was too naive for our use cases.

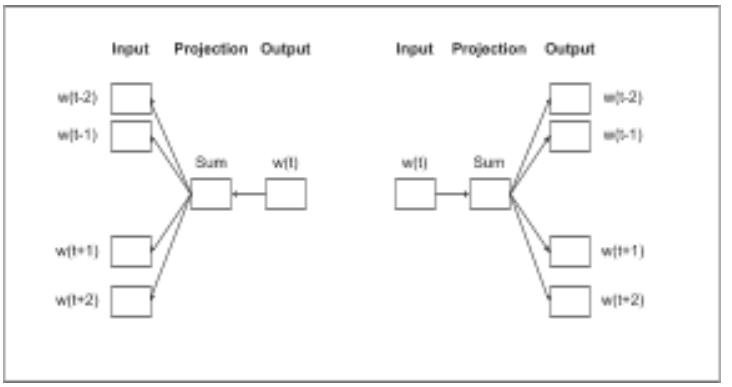

Figure 2: CBOW architecture and Skip-gram architecture

2.3.4 SKIP-GRAM MODEL

The Skip-Gram model [1] is an architecture to find word representations that are useful for predicting the surrounding words.

The tokenized sentence is actually a sequence of training words. Input to the network is the one hot representation of the word and output to the system is the context words. The weight that leads to the output is the matrix of interest.

A sequence of training words w1.....wt is passed to the network and tries to maximize the average log probability.

$$p(c \mid w; \theta) = \frac{e^{v_c . v_w}}{\sum_{c' \in C} e^{v_{c'} . v_w}} \tag{3}$$

Taking Log of equation (3);

$$log\ p(c \mid w; \theta) = log(\frac{e^{v_c . v_w}}{\sum_{c' \in C} e^{v_{c'} . v_w}}) \tag{4}$$

Cost function:

$$L(\theta) = \prod_{w \in Text} \prod_{c \in C(w)} p(c \mid w; \theta) = \prod_{(w,c) \in D} p(c \mid w; \theta) \tag{5}$$

Maximizing the log-likelihood:

$$arg\ max \sum_{(w,c) \in D} log\ p(c \mid w; \theta) = \sum_{(w,c) \in D} (v_c . v_w - log \sum_{c'} e^{v_{c'} . v_w}) \tag{6}$$

It is computationally expensive to calculate $p(c \mid w; \theta)$ since it computes $\sum_{c' \in C} e^{v_{c'} . v_w}$ over all the context. Thus negative sampling is more efficient

{ window size : 1}	Training Example
विभिन्न एपहरुको निम्तो वा अनुरोधबाट कत्तिको दिक्क हुनुभएको	विभिन्न एप
विभिन्न एपहरुको निम्तो वा अनुरोधबाट कत्तिको दिक्क हुनुभएको	एपहरुको विभिन्न एपहरुको निम्तो
विभिन्न एपहरुको निम्तो वा अनुरोधबाट कत्तिको दिक्क हुनुभएको	निम्तो एपहरुको निम्तो वा
विभिन्न एपहरुको निम्तो वा अनुरोधबाट कत्तिको दिक्क हुनुभएको	वा निम्तो वा अनुरोधबाट
विभिन्न एपहरुको निम्तो वा अनुरोधबाट कत्तिको दिक्क हुनुभएको	अनुरोधबाट वा अनुरोधबाट कत्तिको

The above table explains our data pipeline for training the neural network.

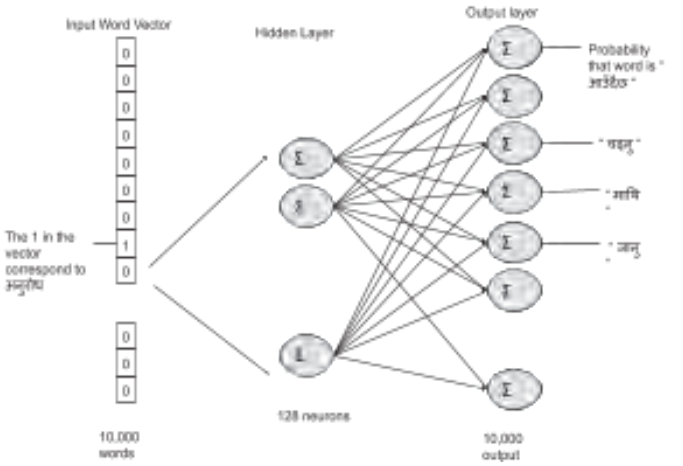

Figure 3: Our Neural Network Architecture

Once we train our neural network. The weights obtained in the hidden layer will be our vector matrices for our given dataset. The normalization on the denominator is costly using softmax so we used NCE loss in the last layer of neural network for better performance.

2.3.5 NOISE CONTRASTIVE ESTIMATION

Negative sampling is a variation of NCE used by the popular word2vec tool which generates a proxy corpus and also learns θ as a binary classification problem, but it defines the conditional probabilities given (w, c) differently. NCE reduces the language model estimation problem to the problem of

estimating the parameters of a probabilistic binary classifier that uses the same parameters to distinguish samples from the empirical distribution from samples generated by the noise distribution [6]. The basic idea is to classify if the sample is from true distribution raising D = 1 or from noise distribution raising D = 0. Such that it can be trained as a binary classifier.

$$p(D = 0 \mid c, w) = k \times q(w)\, u_\theta(w, c) + k \times q(w) \tag{7}$$

$$p(D = 1 \mid c, w) = u_\theta(w, c)\, u_\theta(w, c) + k \times q(w) \tag{8}$$

3. RESULTS

We computed the model and compared "किताब" and "पुस्तक " are found to be similar, even though they never come as nearby words. Here the context of their presence in a sentence comes to play. Since they are used in almost the same context in a sentence as in "किताब पढ्यौ" or "पुस्तक पढ्यौ".

As in Fig 4. "जनजाती" is found at the nearest point in space with "थारु", "मुस्लिम" and so on which all signify a type of caste in Nepali.

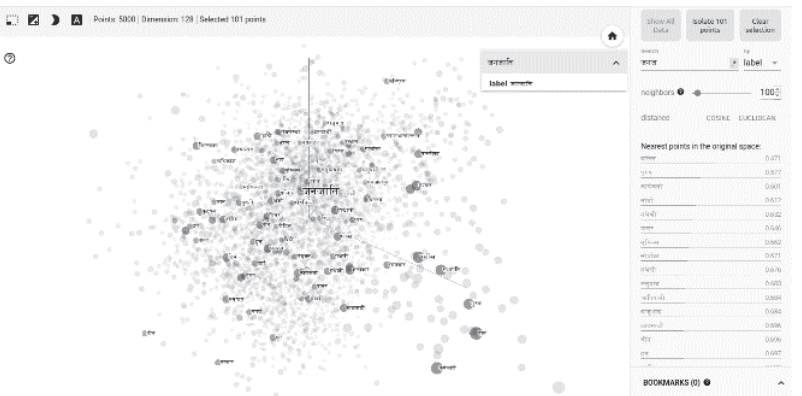

Figure 4: Word Embedding Visualization, "जनजाती"

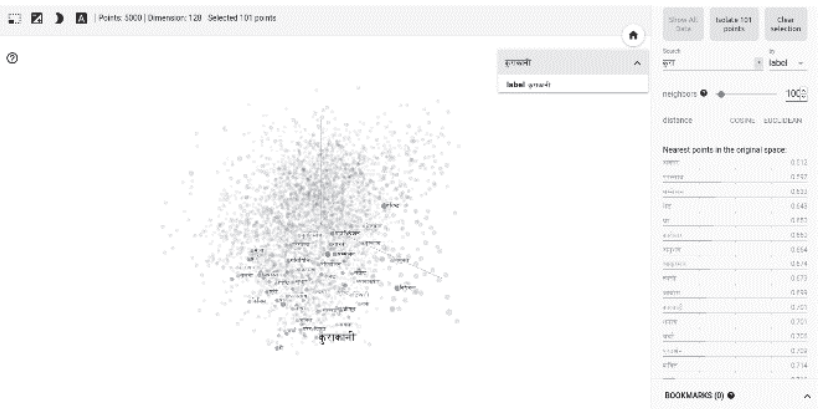

Figure 5: Word Embedding Visualization, "कुराकानी"

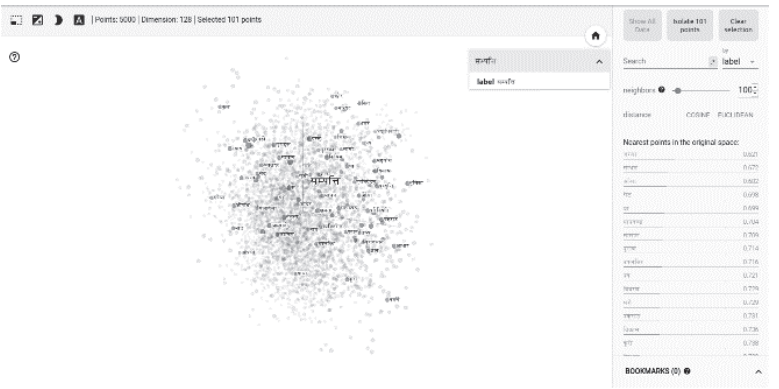

Figure 6: Word Embedding Visualization, "सम्पत्ति"

4. CONCLUSION

From the collection of scarce raw data from news portals to computing a model based on Skip-gram, we have demonstrated the ways to vectorize Nepali words. While digital processing is gaining new heights in terms of speed and size, there was always a need for an efficient method to parse Nepali natural language with fewer resources and complex syntactic structures.

We implemented our model to find out that it successfully highlights similar words based on Euclidean distance with fair accuracy.

5. FOLLOW UP WORK

We could treat the most common words as a single word phrase.
For example:

लोक सेवा आयोग = "लोक_सेवा_आयोग "

अनि के = "अनि_के "

सही हो = "सही_हो "

Similarly, we could reduce the frequent words based on probability. i.e, effectively delete it using Probability

$$P(w_i) = \left(\sqrt{\frac{Z(w_i)}{0.001}} + 1\right) \cdot \frac{0.001}{Z(w_i)} \quad (9)$$

Where,

$$Z(w_i) = \frac{number\ of\ occurence\ of\ a\ particular\ word}{Total\ number\ of\ words}$$

ACKNOWLEDGEMENT

The authors gratefully acknowledge the contributions of Mikolov, T., Chen, K., Corrado, G., and Dean, J. for their work on the original version of this document. We are grateful to Nepali news portals from where we acquired the data to train our model. We would like to acknowledge with gratitude, the support and encouragement from the Department of Electronics and Computer Engineering, Institute of Engineering, Thapathali Campus.

REFERENCES

[1] Mikolov, T., Chen, K., Corrado, G., and Dean, J, “Efficient estimation of word representations in vector space”, 2013.
[2] Mikolov, Tomas & Sutskever, Ilya & Chen, Kai & Corrado, G.s & Dean, Jeffrey, “Distributed Representations of Words and Phrases and their Compositionality”,Advances in Neural Information Processing Systems. 2013.
[4] Goldberg, Yoav & Levy, Omer, “word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method”, 2014.
[5] Wongsuphasawat, Kanit & Smilkov, Daniel & Wexler, James & Wilson, Jimbo & Mane, Dandelion & Fritz, Doug & Krishnan, Dilip & Viegas, Fernanda & Wattenberg, Martin.“Visualizing Dataflow Graphs of Deep Learning Models in TensorFlow”. IEEE Transactions on Visualization and Computer Graphics. PP. 1-1. 10.1109/TVCG.2017.2744878, 2017.
[6] Dyer, Chris. “Notes on Noise Contrastive Estimation and Negative Sampling.”, 2014.
[7] Gutmann, Michael & Hyvärinen, Aapo. “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models.” Journal of Machine Learning Research - Proceedings Track. 9. 297-304. 2010.
[8] Rabindra Lamsal, "300-Dimensional Word Embeddings for Nepali Language", IEEE Dataport, 2019. [Online]. Available: http://dx.doi.org/10.21227/dz6s-my90. Accessed: Dec. 25, 2019.