EasyChair Preprint

№ 2073

Nepali POS Tagging using Deep Learning Approaches

Sarbin Sayami, Tej Bahadur Shahi and Subarna Shakya

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

December 1, 2019

Nepali POS Tagging Using Deep Learning Approaches

Abstract — **Part of Speech (POS) tagging is one of the fundamental task in Natural Language Processing (NLP). It plays vital role in various NLP applications such as machines translation, text-to-speech conversion, question answering, speech recognition, word sense disambiguation and information retrieval. It is also referred as grammatical tagging or word-category disambiguation which is a process of labeling every word in sentences with tag based on its context and syntax of the language. It is challenging to develop promising POS tagger for morphologically rich language like Nepali. This paper focuses on implementing and comparing different deep learning based POS tagger for Nepali such as Simple Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bi-directional Long Short Term Memory (Bi-LSTM). These approaches were trained and tested in a corpus of Nepali tag set. The result shows that Bi-directional LSTM outperforms all other three approaches.**

Index Terms— POS, NLP, Simple RNN, LSTM, GRU, Bi-LSTM, Nepali tag set

I. Introduction

Natural Language Processing is a field of computer science which focuses on making computer understands the words or sentences written in human languages. It is a subset of Artificial Intelligence and linguistic which has a start history backed in 1950s. Alan Turing introduced the concept of Turing test from the article titled “Machine and Intelligence”. SHRDLU were one of the Natural Language System which worked in restricted “block worlds” developed around 1960s. It can be classified as symbolic approach, statistical approach and connectionist approach. Symbolic approach is based on deep understanding of linguistic knowledge and a wide variety of rules. It should also consist of knowledge representation schemes to represent the facts about language unambiguously. Statistical approach focuses on creation of large text corpora which is a repository of real examples of linguistic phenomena. These corpora are used to generate approximate models to represent linguistic phenomena without adding linguistic rules. This approach has been extensively used in speech recognition, lexical acquisition, parsing, part-of-speech tagging, collocations, statistical machine translation, statistical grammar learning etc. Connectionist approach combines statistical learning with other theories of representation to allow transformation, inference and manipulation of logic formula. It focuses on creation of mesh network where weights are used to represent knowledge.

POS tagging, also referred as grammatical tagging or word-category disambiguation is a process of labeling every word in sentences with tag. It can be carried out manually or automatically. POS commonly includes verbs, nouns, adjectives, adverbs, determiner, and so on [1].

POS tagging is one of the fundamental task in NLP. It plays vital role in various NLP applications such as machines translation, text-to-speech conversion, question answering, speech recognition, word sense disambiguation and information retrieval [2].

POS tagging can be carried out with various approaches rule-based, Stochastic and neural network. These taggers are based on the assumption that a word can be assigned a single POS tag in a given context, or at least one of the possible parts of speech is most likely [3].

A. *Rule Based*

It uses linguistic knowledge to formulate simple rules that assign POS to an ambiguous word using context information [4]. These rules are often known as context frame rules. Similarly, transformation based approaches can also be implemented in conjunction to pre-defined set of rules where new transformation rules are automatically induced during the training [5].

The rule based approach requires pre-annotated training corpora. However, transformation rules can be induced via training. It initially assigns tags to words with reference to the tag with the highest frequency for a particular word. These tags are then refined and then trained to learn new rules [5].

Brill Tagger is tagging program introduced by Eric Brill in 1992. The program is based on Regulation and transformation. Brill’s Tagger is a transformation-based tagger. It uses a series of rules to correct the results of an initial tagger. These rules are scored based on how many errors are in the correct minus the number of new error.

Briefly Brill Tagger algorithm includes the initialization process consisting of known and unknown words, as well as the learning phase consisting of repeating in calculating the error value of each candidate rule, choosing the rule of the best, adding a set of rules and applied to the text, and repeating until no rules that have a value above a certain threshold or who have granted. Brill Tagger performs annotation corpus provision in three ways: lexicon, lexical, and contextual. [5].

B. Stochastic Approach

A stochastic approach includes frequency, probability or statistics. Frequency based approach are based on tagging the word in an unannotated text using the most frequently used tag for a specific word in the annotated text. However, it might come up with rules that do not agree with the standard grammatical rules of a particular language. Unigram is a simple statistical tagging algorithm. A unigram generally refers to a single token. Therefore, a unigram tagger only uses a single word as its context for determining the part of speech tag [6]. Unigram builds a context model from the list of tagged sentences. It uses to give a tag with unigram tagger prior to be trained on a training corpus. That corpus will be determined in which tags are most common for each word. Default tag 'none' assign to any token is not encountered in training data. Unigram Tagger behaves like a search tagger unless there is an easier technique for setting it up, called training. Training Unigram Tagger by specifying sentence data is marked as a parameter when initializing tagger. The training process involves checking the dictionary tags stored inside the tagger [6].

Probabilistic approach is also known as n-gram approach which can be unigram, bigram and trigram. It determines the best tag for a word by calculating the probability of that it occurs with n previous tags. The most common model used is the Hidden Markov Model (HMM). HMM is a statistical Markov model with the system model assumed to be a Markov process with some hidden states. The HMM established the probabilistic method for Part-of-speech tagging. The mathematics behind the HMM was developed by L. E. Baum and coworkers HMM are used in many applications such as speech recognition, or bioinformatics. HMM taggers choose the tag sequence that maximizes the following formula:

$$P(\text{word}|\text{tag}) * P(\text{tag}|\text{previous n tag})$$

The HMM approach is different from other POS tagging approaches in tag combinations. HMM considers the best combination of tags for word order, while other tagging methods greedily tag one word at a time, regardless of the optimal combination.

Entities in Hidden Markov Models are based on tagging approach, $\{w_1, w_2 \ldots w_w\}$ is a set of words, $\{t_1, t_2 \ldots t_T\}$ is a set of POS Tags, $W_{1,n} = W_1 W_2 \ldots W_n$ is a sentence of n words and $T_{1,n} = T_1 T_2 \ldots T_n$ is a sequence of n POS Tags

The probability in HMM using following formula can be used to find the most likely sequence of POS tags for a given sequence of words.

$$\Pr(t_{1,n}, w_{1,n}) \approx \Pi_{i=1}^{n} \ (\Pr(t_i \mid {}_{ti-k,\ i-1}) \text{ X } \Pr(w_1 \mid t_i)$$

Probability current tag t_1 depends on previous k tags and probability current word depends on only current tag t_1 [6].

The unigram and HMM based taggers are easy to build, however given the nature of their probability models; it is hard to incorporate more complex features into them. The Maximum Entropy (ME) based tagger is introduced to provide a principled way of incorporating complex features into probability models.

Given a sentence $w_1 \ldots w_n$, an ME based tagger models the conditional probability of a tag sequence $t_1, \ldots t_n$ as:

$$\Pr(t_{1,\ldots} t_{n\,|}\, w_{1,\ldots} w_{n}) \approx \Pi_{i=1}^{n} \ P(t_i \mid C_i)$$

where $C_1, \ldots C_n$ are the corresponding contexts for each word appearing in the sentence. The context C of a word w includes the previous assigned tags before w.

An ME based tagger introduces the concept of features which encode elements of a context C useful for predicting the tag t of a word w. Features are binary valued functions that represent constraints. An ME based tagger will use the features to compute $P(t_i|C_i)$. It will learn the weights of the features that can maximize the entropy of the probability model using the training corpus [6].

C. Neural Network Approach

Apart from feed forward network, several variations of RNN have been implemented for POS tagging. Some of them are Simple RNN, GRU, LSTM and bidirectional RNN's [7, 8].

D. Problem Statement

POS tagger plays an important role and can critically affect the performance of the complex NLP systems if error occurs. Rule based and statistical techniques do not show significant results as they do not take care of context and sequence. Very few implementations of deep learning approaches can be found in context of morphologically rich languages like Nepali.

E. Objective

To implement deep learning approaches for POS tagging of Nepali text and to compare the results of these approaches

II. Literature Review

RNN has been broadly applied to NLP problems. This kind of neural network is designed for modeling sequential data and has been testified to be quite efficient in sequential tagging tasks. Similarly, the LSTM unit was initially proposed by Hochreiter and Schmidhuber. These units are used to propagate an important feature that came early in the input sequence over a long distance, which helps in capturing potential long distance dependencies.

LSTM networks perform very well with respect to other learning algorithms when word dependencies are long. Although without a big improvement, POS tagging systems which exploit

LSTM as learning algorithm have been proven to reach state-of-the-art performances both when analyzing text at character level and at word level [9].

In [10], bidirectional LSTMs was used to read the character sequences that constitute each word and combine them into a vector representation of the word. This model assumed that each character type is associated with a vector, and the LSTM parameters encoded both idiosyncratic lexical and regular morphological knowledge.

To evaluate the model, they used a vector based model for POS tagging and for language modeling and they carried out experiments on these tasks in several languages. Their results show that their model obtained state-of-the art performance on POS tagging including establishing a new best performance in English.

In [11], bi-directional RNN with LSTM units was used for Chinese word segmentation, which is a crucial preprocess task for modeling Chinese sentences and articles. Classical methods focus on designing and combining hand-craft features from context, whereas BI-LSTM does not need any prior knowledge or pre-designing, and it is expert in keeping the contextual information in both directions. Experiment result shows that their approach gets state-of-the-art performance in word segmentation on both traditional Chinese datasets and simplified Chinese dataset.

More specifically they used a bidirectional LSTM which allowed capturing long-range dependencies from both directions of a sentence by constructing bidirectional links in the network.

In [12], a sequence to sequence approach was followed to model the problem with various deep learning algorithms such as RNN, LSTM, GRU, and their bi-directional variants. Bi-directional versions of RNN, LSTM and GRU achieved the maximum performance scores with binary cross entropy as the loss function. The accuracy of the system also increased with the increase in the size of word embedding vector. The accuracy obtained by RNN was 91.68%, LSTM was 91.74%, GRU was 91.66 and Bi-LSTM was 92.66 respectively.

III. BACKGROUND STUDY

A. Simple RNN

In simple RNN architecture, there is simple multiplication of Input (xt) and Previous Output (ht-1). It can be seen from Figure 1. This output is passed through Tanh activation function to get the final output. There are no Gates present in the architecture.

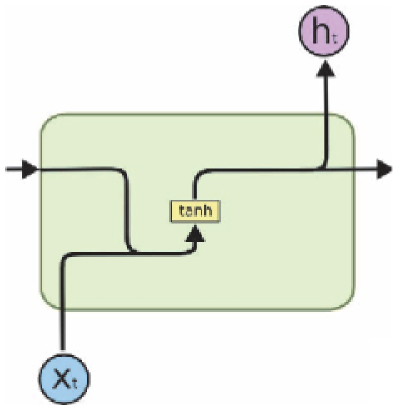

Figure 1: Simple RNN Architecture

B. GRU

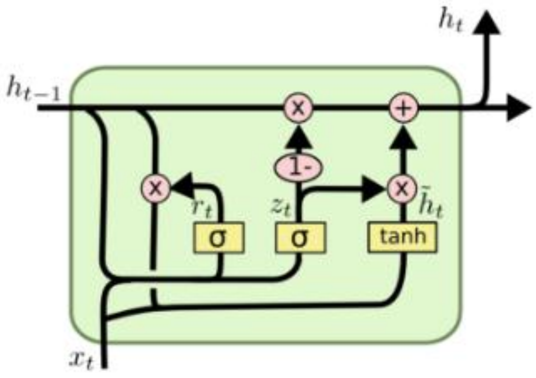

Figure 2: GRU Architecture

In this architecture, Update gate is introduced, to decide whether to pass previous output (h_{t-1}) to next cell (as h_t) or not. Forget gate is an additional mathematical operation with a new set of Weights (W_t) as shown in Figure 2.

The main equations used are as follows

$$z_t = \sigma\, (W_z\, .[h_{t-1}\, ,\, x_t])$$

$$r_t = \sigma\, (W_r\, .[h_{t-1}\, ,\, x_t])$$

$$\check{h}_t = \tanh\, (W.[\, r_t * h_{t-1}\, ,\, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \check{h}_t$$

C. LSTM

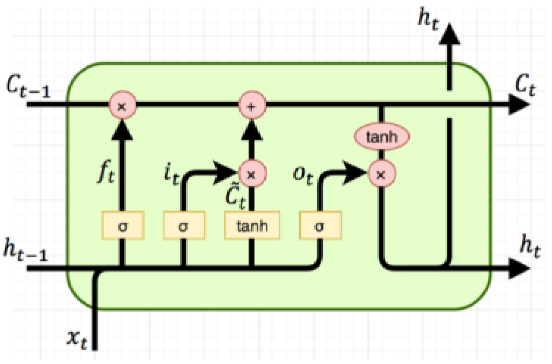

Figure 3: LSTM Architecture

So overall, LSTM has introduced two math operations having two new sets of Weights. Its architecture is shown in Figure 3.

The main equations used are as follows

$$f_t = \sigma (W_f .[h_{t-1} , x_t] + b_f)$$

$$i_t = \sigma (W_i .[h_{t-1} , x_t] + b_i)$$

$$\check{c}_t = \tanh (W_c .[h_{t-1} , x_t] + b_c)$$

$$Ct = f_t * C_{t-1} + i_t * \check{c}_t$$

$$o_t = \sigma (W_o .[h_{t-1} , x_t] + b_o)$$

$$h_t = o_t * \tanh (C_t)$$

D. Bidirectional Neural Network

Bi-LSTM neural network is similar to LSTM network in structure because both of them are constructed with LSTM units. The special unit of this network is capable of learning long-term dependencies without keeping redundant context information. They work tremendously well on sequential modeling problems, and are now widely used in NLP tasks.

Bi-LSTM network is designed to capture information of sequential dataset and maintain contextual features from past and future. Different from LSTM network, Bi-LSTM network has two parallel layers propagating in two directions, the forward and backward pass of each layer are carried out in similar way of regular neural networks, these two layers memorize the information of sentences from both directions.

IV. RESEARCH METHODOLOGY

A. Data Collection

Data were collected from Madan Puraskar Pustakalaya. It consists of Nepali English parallel corpus annotated with 43 POS tag developed and contains nearly 88000 words. The design of this Nepali POS Tag-set was inspired by the PENN Treebank POS Tag-set. Hence, whenever possible, the same naming convention has been used as in the case of the Penn Treebank Tag-set. The sample of POS tagged Corpus is as shown in Figure 4.

हामी<PP> कसै<DUM>ले<PLE> अस्बेस्टस<NNP>मा<POP> आपत्तिजनक<JJ> गुण<NN> रहेको<VBKO> सुन्नु<VBI>भन्दा<VBO> अगाडि<RBO> वर्षौँ<NN> अघि<POP>को<PKO> बारेमा<POP> कुरा<NN> गरिरहेका<VBKO> छौँ<VBX> ।<YF>

Figure 4: Sample of Nepali POS tagged corpus

B. Data Preparation

Parallel corpus was developed where first consisted of plane Nepali text and other consisted of only tags as shown in Figure 5.

हामी कसै ले अस्बेस्टस मा आपत्तिजनक गुण रहेको सुन्नु भन्दा अगाडि वर्षौँ अघि को बारेमा कुरा गरिरहेका छौँ।

<PP> DUM> <PLE> <NNP> <POP> <JJ> <NN> <VBKO> <VBI> <VBO> <RBO> <NN> <POP> <PKO> <POP> <NN> <VBKO> <VBX> <YF>

Figure 5: Sample of plane and its corresponding tags

C. Training

The training was carried out using RNN architecture as shown in Figure 6.

RNN were used for training as it does not require input data to be fixed and their future input information is reachable from the current state. Simple RNN, LSTM, and GRU were implemented. However, Bi-LSTM was also implemented to demonstrate its ability to outperform all the former three architectures.

It connects two hidden layers of opposite directions to the same output. BI-LSTM are able to understand context better due to its ability to approach a unit from both the directions.

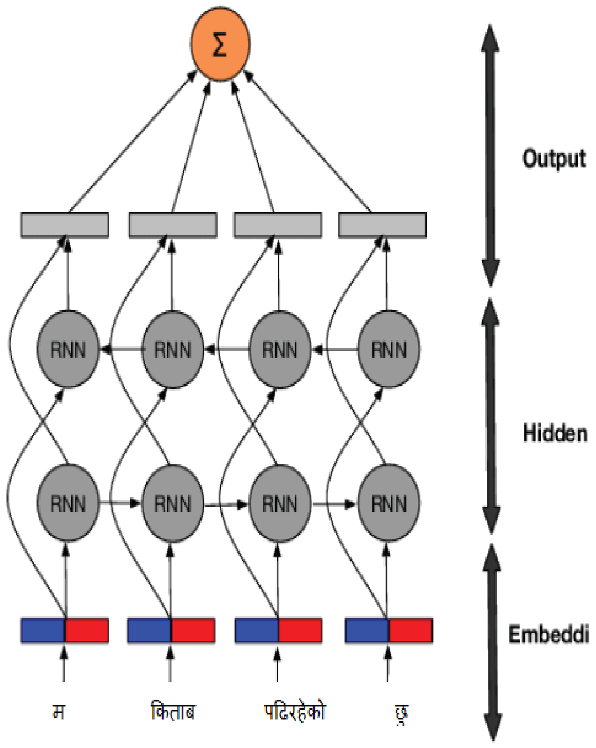

Figure 6: Bi-LSTM architecture for Training

D. *Testing*

To implement deep learning approaches for POS tagging of Nepali text and to compare the results of these approaches

The model was tested using the testing set. The data set was divided into three parts i.e. training, development and testing.

E. *Architecture*

To implement deep learning approaches for POS tagging of Nepali text and to compare the results of these approaches

Table 1: Architecture of RNN

Layer (type)	**Output Shape**
Embedding	100, 25
Simple RNN/ LSTM/ BI-LSTM/ GRU	100, 50
TimeDist	None, 100, 40
TimeDist	None, 100, 40

As shown in Table 1, the architecture consists of 4 layers. The first one is the embedding layer, followed by variants of RNN's and finally there are two distributed layers. The final output layer consists of 40 units as there are 40 tags defined and output has to be one of them.

F. *Vocabulary Size*

The vocabulary size set was 10000 with a maximum sequence length of 100. The untokenized word was set to '_unk_' as shown in Figure 7.

```
VOCAB_SIZE = 6850

MAX_SEQUENCE_LENGTH = 100

UNK_TOKEN = '__unk__'
```

Figure 7: Setting of vocabulary size and maximum sequence

G. *POS Tags*

A total of 38 POS tags were used and were given integer numbers as shown in Table 2.

Table 2: POS Tags and their corresponding integer values

POS_TAG={	
'NOTAG': 0,	'JJM': 21,
'CD': 1,	'RP': 22,
'JJ': 2,	'VBNE': 23,
'NNP':3,	'CS': 24,
'POP': 4,	'YQ': 25,
'NN': 5,	'CL': 26,
'PKO': 6,	'PP': 27,
'VBX': 7,	'PP$': 28,
'YF': 8,	'CC': 29,
'FB': 9,	'SYM': 30,
'VBF': 10,	'PPR': 31,
'PLAI': 11,	'DM': 32,
'DUM': 12,	'OD': 33,
'VBKO': 13,	'QW': 34,
'RBO': 14,	'UNW': 35,
'VBI': 15,	'RBM': 36,
'VBO': 16,	'FW': 37,
'HRU': 17,	'YB': 38,
'JJD': 18,	'ALPH': 39,
'YM': 19,	}
'PLE': 20,	

H. *Preparation of Text and Labels in List*

The data was loaded and was preprocessed as shown in Figure 8.

```
[(['\ufeff६१', ' वर्षीय', ' पियरे', ' भिन्केन', ' नोभेम्बर', ' २९', ' बाट', ' सल्लाहकार', 'को', ' रूप', ' मा', ' सञ्चालक', ' समिति', 'मा', ' आउनुहुनेछ', ' ।'], [1, 2, 3, 3, 3, 1, 4, 5, 6, 5, 4, 5, 5, 4, 7, 8])]
```

Figure 8: Text Preparation

As shown in Figure 10, the first half represents the tokens and the second half represents their POS with their respective integer values as described in section 5.2.

I. *Preparation of train and test data*

The data were divided for training, development and testing respectively as shown in Figure 9.

Train Instances: 1341

Dev Instances: 191

Test Instances: 383

Figure 9: Division of data

J. *Preparation of vocabulary*

The vocabulary was prepared based on the occurrences of tokens. It is based on most frequent tokens, least frequent tokens and unknown tokens. It can be seen from Figure 10.

Most frequent tokens

को: 1548

।: 1272

मा: 1128

ले: 1127

हरू: 948

,: 659

लाई: 573

का: 530

र: 464

छ: 265

Least frequent tokens

भित्र्याएको: 1

भड्किला: 1

क्लव: 1

मात्र: 1

बेभर्ली: 1

बर्निलर: 1

ओलिभर: 1

अनगिन्ती: 1

घुस्न: 1

महानगरिय: 1

Figure 10: Sample of most frequent and least frequent tokens

The unknown tokens are then filtered out to reduce the memory consumption. The total numbers of unknown tokens are listed out in Figure 11.

Train: 0/34082

Dev: 558/4642

Test: 1321/10226

Figure 11: Total number of unknown tokens filtered

K. *Simple RNN POS tagger*

The summary of the model is as shown in Table 3.

Table 3: Simple RNN Summary

Layer (type)	**Output Shape**	**Param #**
Embedding	(None, 100, 25)	171300
Simple RNN	(None, 100, 50)	1275
TimeDist	(None, 100, 40)	1040
TimeDist	(None, 100, 40)	0

```
==========================================
=================
Total params: 173,615
Trainable params: 173,615
Non-trainable params: 0
```

The accuracy obtained after training is 85.04% with a loss function of 0.5662. The test accuracy obtained for simple RNN was 96.84%.

L. *Unidirectional LSTM based POS tagger*

The summary of the model is as shown in Table 4.

Table 4: Unidirectional LSTM Summary

Layer (type)	**Output Shape**	**Param #**
Embedding	(None, 100, 25)	171300
LSTM	(None, 100, 50)	5100
TimeDist	(None, 100, 40)	1040
TimeDist	(None, 100, 40)	0

```
==========================================
=================
Total params: 177,440
Trainable params: 177,440
Non-trainable params: 0
```

The accuracy obtained after training is 98.97% with a loss function of 0.0612. The test accuracy obtained for unidirectional LSTM was 96.48%.

M. *Bidirectional LSTM based POS tagger Neural Network*

The summary of the model is as shown in Table 5.

Table 5: bidirectional LSTM Summary

Layer (type)	Output Shape	Param #
Embedding	(None, 100, 25)	250050

Bidirection	(None, 100, 50)	10200
TimeDist	(None, 100, 40)	2040
TimeDist	(None, 100, 40)	0

```
=========================================
=================
Total params: 183,540
Trainable params: 183,540
Non-trainable params: 0
_________________________________________
_________________
```

The accuracy obtained after training is 99.62% with a loss function of 0.0190. The test accuracy obtained for bidirectional LSTM was 97.27%.

N. GRU based POS tagger

The summary of the model is as shown in Table 6.

Table 6: GRU Summary

Layer (type)	Output Shape	Param #
Embedding	(None, 100, 25)	171300
GRU	(None, 100, 50)	3825
TimeDist	(None, 100, 40)	1040
TimeDist	(None, 100, 40)	0

```
=========================================
=================
Total params: 176,165
Trainable params: 176,165
Non-trainable params: 0
_________________________________________
_________________
```

The accuracy obtained after training is 99.60% with a loss function of 0.0195. The test accuracy obtained for GRU was 96.86%.

O. Results of Nepali POS tagging

Simple RNN, LSTM, and GRU were implemented. However, Bi-LSTM was also implemented to demonstrate its ability to outperform all the former three architectures. It connects two hidden layers of opposite directions to the same output. Bi-LSTM are able to understand context better due to its ability to approach a unit from both the directions.

The data were divided for training, development and testing. The total sentences in the corpus were divided into 1341 as training instances, 191 as development instances and 383 as testing instances respectively. Three deep learning based model were trained and tested namely: Simple RNN, LSTM, Bi-LSTM and GRU.

Table 7: Accuracy and Loss value of RNN variants

Model	Accuracy %	Loss value
Simple RNN	96.84	0.0221
Unidirectional LSTM	96.48	0.0612
Bidirectional LSTM	97.27	0.0190
GRU	96.86	0.0195

The results of Nepali POS tagging comprising of loss value calculated using cross-entropy and accuracy with simple RNN, unidirectional LSTM, and bidirectional LSTM is shown in 7.

V. CONCLUSION AND FUTURE WORK

POS tagging of Nepali Text was carried out using simple RNN, LSTM, GRU and Bi-directional LSTM in a Nepali tagged corpus of tag size 40. The data set was divided into three sections i.e. training, development and testing. The accuracy obtained for simple RNN, LSTM, GRU and Bi-directional LSTM was 96.84%, 96.48%, 96.86% and 97.27% respectively. Therefore, Bi-directional LSTM outperformed all other three variants of RNN. In future, the vocabulary size and tag set can be increased to increase the efficiency. Similarly, reinforcement learning can be added for efficient training.

REFERENCES

[1] A. Z. Ahmad, H. Rudy , and I. W. Mustika, “A Comparison of Different Part-of-Speech Tagging Technique for Text in Bahasa Indonesia” in *7th International Annual Engineering Seminar (InAES), Yogyakarta, Indonesia,* 2017.

[2] F. Rana , S. Mehrnoush, and M. Pouyan, “An Efficient Meta Heuristic Algorithm for POS-Tagging” in *International Conference on Computing in the Global Information Technology (ICCGI*), IEEE, 2010.

[3] J.A. Perez-Ortiz , M.L. Forcada, “Part-of-speech tagging with recurrent neural networks” in *International Joint Conference on Neural Networks Proceedings*, IEEE, 2001.

[4] P. J. Antony, K.P. Soman, “Kernel Based Part Of Speech Tagger For Kannada” in *International Conference on Machine Learning and Cybernetics,* IEEE, 2010.

[5] H. M. Fahim, “Comparison Of Different Pos Tagging Techniques For Some South Asian Languages” , *A Thesis Submitted to the Department of Computer Science and Engineering of BRAC University,* 2006.

[6] T. Yuan, L. David, “A Comparative Study on the Effectiveness of Part-of-Speech Tagging Techniques on Bug Reports” in *SANER 2015, Montréal, Canada,* IEEE, 2015.

[7] A. Firoj, A.C Shammur, “Bidirectional LSTMs - CRFs Networks for Bangla POS Tagging” in *19th International Conference on Computer and Information Technology, North South University, Dhaka, Bangladesh,* IEEE, 2016.

[8] Y. Archit, “ANN Based POS Tagging For Nepali Text” in *International Journal on Natural Language Computing (IJNLC) Vol.7, No.3,* 2018.

[9] T. L. Wang, L. Tiago, L. Marujo, A. F. Ram´on, A. Silvio , D. Chris, B. W. Alan and T. Isabel, “Finding Function in Form: Compositional Character Models for Open VocabularyWord Representation” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1520–1530,Lisbon, Portugal, 17-21 September 2015. c 2015 Association for Computational Linguistics,* ACL, 2015.

[10] W. Peilu, Q. Yao, S.K. Frank, H. Lei and Z. Hai, “Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network” in *arXiv:1510.06168v1 [cs.CL]* , 2015.

[11] Y. Yushi, H. Zheng, “Bi-directional LSTM Recurrent Neural Network for Chinese Word Segmentation” in *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part IV (pp.345-353)*, 2016.

[12] P. Greeshma, P.V. Jyothsna, K.K. Shahina, B. Premjith, and K.P. Soman, “A Deep Learning Approach for Part-of-Speech Tagging in Nepali Language” in *International Conference on Advances in Computing, Communications and Informatics,* IEEE, 2018.