Deep Neural Model for Manipuri Multiword Named Entity Recognition with Unsupervised Cluster Feature

Jimmy Laishram
Department of CSE
Manipur Technical University
Imphal, Manipur, India
jimmy_l@mtu.ac.in

Kishorjit Nongmeikapam
Department of CSE
IIIT, Manipur
Imphal, Manipur, India
kishorjit@iiitmanipur.ac.in

Sudip Kumar Naskar
Department of CSE
Jadavpur University
Kolkata, West-Bengal, India
sudip.naskar@cse.jdvu.ac.in

Abstract

The recognition task of Multi-Word Named Entities (MNEs) in itself is a challenging task when the language is inflectional and agglutinative. Having breakthrough NLP researches with deep neural network and language modelling techniques, the applicability of such techniques/algorithms for Indian language like Manipuri remains unanswered. In this paper an attempt to recognize Manipuri MNE is performed using a Long Short Term Memory (LSTM) recurrent neural network model in conjunction with Part Of Speech (POS) embeddings. To further improve the classification accuracy, word cluster information using K-means clustering approach is added as a feature embedding. The cluster information is generated using a Skip-gram based words vector that contains the semantic and syntactic information of each word. The model so proposed does not use extensive language morphological features to elevate its accuracy. Finally the model's performance is compared with the other machine learning based Manipuri MNE models.

1 Introduction

Multi Word Named Entity (MNE) is a part of Multiword Expression (MWE) which is an ordered group of words that can exist independently and carries different meaning as opposed to its constituent word (Nongmeikapam and Bandyopadhyay, 2011). Accurate recognition of such MNE plays a vital role in various NLP tasks such as POS tagging (Nongmeikapam et al., 2011b), Chunking (Nongmeikapam et al., 2014), NER classification (Singh and Bandyopadhyay, 2010). Manipuri, being a highly inflectional language where affixes define the nature of the words (Choudhury et al., 2004), machine recognition of MNE presents a challenging task for NLP researchers. In Manipuri, the majority of researches on MNE classification are done using machine learning approaches such as CRF (Nongmeikapam and Bandyopadhyay, 2010, 2011; Nongmeikapam et al., 2011a) and SVM (Singh and Bandyopadhyay, 2010). These researches use extensive morphological features to obtain accurate recognition of the MNE. Such morphological features include affixes, context words, digit features etc. For an agglutinative and low-resource language, the inflectional nature amounts to the large Out of Vocabulary (OOV) words, thus making any sequence labelling task or morphological feature creation tasks difficult.

Word embedding is a distributed representation of text in low dimensional real valued vectors and are known to contain semantic or syntactic information and has shown to be an effective feature for many natural language classification tasks of English language (Wang et al., 2015). Word embedding has also been extremely useful for Chinese language processing (Yin et al., 2016), Japanese language processing (Kitagawa and Komachi, 2017) and for some Indian languages processing (Ajay et al., 2016; Bhattacharya et al., 2016). In Manipuri, the effectiveness of word embeddings to any NLP tasks, till this date remains unanswered. The general attempt to work in Manipuri NLP task using embedding follows the idea used in English i.e. to learn the embedding of a word from the context. Unlike English, Manipuri words are a composite of complicated structure with several affixes producing OOV words. Regardless of its context, affixes to a word plays a major role in defining the semantic meaning (Choudhury et al.,

Proceedings of the 17th International Conference on Natural Language Processing, pages 420–429
Patna, India, December 18 - 21, 2020. ©2020 NLP Association of India (NLPAI)

2004). In this paper, a Manipuri MNE classification task is attempted using a bi-directional Long Short Term Memory (Bi-LSTM) with Skip-gram word embeddings. The usage of extensive morphological features for deep neural network is avoided, as these features would require vector representations which may lead to considerable large input layer to update. To segregate the Manipuri MNE words based on the semantics, a K-means algorithm based cluster information of each word is added as a feature to the LSTM model.

1.1 Manipuri Multi Word Named Entity

Manipuri is an Indian language which is highly agglutinative in nature, tonal, has reduplicated words and no gender marking. It belongs to the Tibeto-Burman languages spoken mostly in North-Eastern region of India which includes Manipur and Assam. The Manipuri MNEs can be decomposed into multiple lexemes and displays lexical, syntactic, semantic, pragmatic idiomatic behaviour. These encompasses all the multi-word named entities such as a person's name, location name etc (Nongmeikapam et al., 2014). Some examples of MNEs are given in table 1 below:

MNE Details	Example
Beginning of Person's Name	আরডি (RD)
Internal of Person's Name	মেহতা (Mehta)
Beginning of Location Name	ন্যু (New)
Internal of Location Name	দিল্লিগী (Delhi−gi)
Beginning of Organization Name	ঠিয়েটর (Theatre)
Internal of Organization name	সেন্টর (Center)

Table 1: Manipuri MNE

Nongmeikapam and Bandyopadhyay (2010) have identified various challenges of MNE recognition task as described below:

- Manipuri lacks capitalization of named entities unlike English or other European languages.
- The MNE inflections can be found in the language because of nominal suffixes and pronominal prefixes.
- Due to free word order in the language, the MNE can appear in subject or object position.
- Many Named Entities (NEs) can appear in a dictionary that carry different meanings which creates a homonym effect.
- Manipuri is a resource constraint language. Annotated corpus, name dictionaries, morphological analyser, POS tagger etc are not readily available.

1.2 Motivation

The agglutinative and inflectional nature of Manipuri has poised a challenge for any computational processing task. Manipuri is a language where NLP resources such as annotated corpus, accurate morphological analyzer etc are not readily available. Above all, the present deep neural network NLP algorithms and language modelling techniques have not been proven its efficiency for language such as Manipuri. The introduction of deep neural network in the Manipuri NLP task such as MNE classification can provide a beneficial step towards the challenges faced in POS tagging, NER etc.

2 Related Works

Notable amount of Manipuri MWE classification research have been done using traditional machine learning approach such as CRF (Conditional Random Field) and SVM (Support Vector Machine). Nongmeikapam and Bandyopadhyay (2010) reported an improvement of Manipuri MWE identification using CRF and Reduplicated Multi-Word Expressions (RMWE). The MWE identification model uses a dictionary and rule-based approach to first detect the complete, mimic, partial, double and echo RMWEs and then prepare a model training set using features such as affix information, POS information word frequency and word length. The CRF model attained a performance F-Score measure of 72.24%.

Nongmeikapam et al. (2011a) conducted a research for identification of RMWE using CRF with various features. The features were stem words, number of affixes, stemmed affixes, POS of surrounding words, surrounding words, length of the word, word frequency and digit features. With the features, the CRF classification model predicted with Recall, Precision and an F-Score measure of

92.91%, 91.90% and 92.40% respectively.

Nongmeikapam and Bandyopadhyay (2011) conducted a research on CRF based MWE identification with Genetic Algorithm (GA) based feature selection method. In genetic algorithm, the MWE features are represented as genes in a chromosome which are binary valued (1 or 0). Selection of a feature is done when the gene value is 1. Random crossover is performed to select the best possible combination of features. To avoid chromosome repetition, random mutation is performed. As a fitness function, three fold cross validation technique is used. Using the technique, the best features were surrounding words, affixes, surrounding POS, word length and word frequency. The CRF model with the aforementioned GA based selected features, attained an increase in F-score by 2.91% as compared to baseline CRF model without GA based feature selection. Overall the model performed with F-Score= 73.74%, Precision= 86.84% and Recall= 64.08%.

Singh and Bandyopadhyay (2010) proposed a web based Manipuri corpus for Multiword NER and RMWE identification model using SVM. For classification purpose, the best features were selected for SVM classifier. The features were context word, word affixes, MNE and RMWE information, digit features, infrequent words, word length and POS information. The model was trained over 1235 sentences with 28629 words and predicted with an F-Score measure of 93.96% for MNE and 94.07% for RMWE.

3 System Design

The Manipuri MNE classification is shown in Figure 1. Being a low resource language with limited information on morphological features, extraction of necessary information for a particular word is crucial as these information will help the model in accurate MNE classification. In this research, two models have been implemented for Manipuri MNE classification. They are

1. A baseline model with Bi-LSTM deep neural networks with Word embedding using Skip-gram with default POS embedding.

2. A model 1 in which the cluster information is added as additional feature to the baseline model.

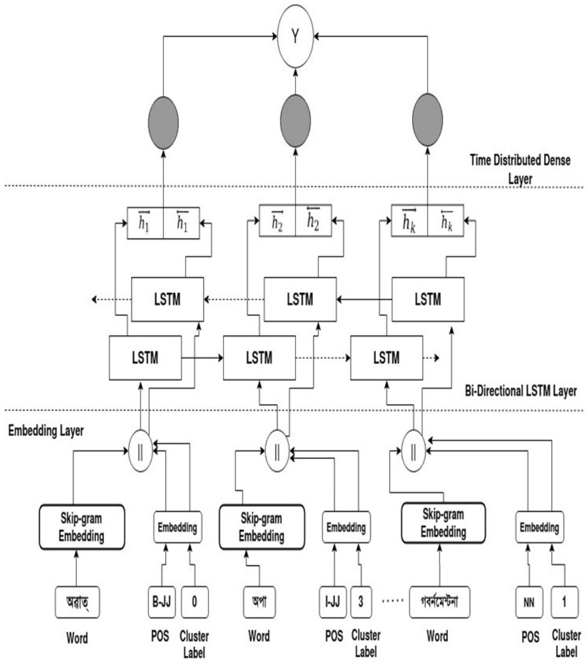

Figure 1: Manipuri MNE Classification System Architecture

The use of word embedding, feature embedding and Bi-LSTM are described in the following subsections.

3.1 Word Embeddings

As deep neural network function on real-valued vectors, it is essential that the input words ($S = x_1, x_2, x_3 ... x_t$ where x_t are the word sequence in a sentence S) are converted to a D-dimensional real-valued word vector that carries semantic and syntactic information. The Skip-gram (Mikolov et al., 2013a) word embedding algorithm is used to create the word vectors because for a small corpora with infrequent words, the skip-gram embedding can represent rare words (Mikolov et al., 2013b) with precision as compared to other word embedding techniques. The skip-gram model is described below.

Skip-gram Model: (Mikolov et al., 2013a) introduced Skip-gram model for vector representation of large amount of unstructured words without the need of dense matrix multiplication. The objective of the Skip-gram model is to find the word representation that can predict surrounding words in a sentence.

The Skip-gram model takes in a sequence of words $W = w_1, w_2, w_3....w_N$ and generates the context word $C = c^1, c^2...., c^k$ on the basis of the center word w_i.

Given a sequence of Manipuri words $w_1, w_2, w_3, ..., w_N$, the Skip-gram model maximizes the average log probability P as given below:

$$\frac{1}{N} \sum_{(n=1)}^{N} \sum_{(-m \leq j \leq m, m \neq 0)} \log(p(w_{n+m}|w_n)) \quad (1)$$

where m is the size of the training context which can be a function of the center word w_i. When $p(w_{n+m}|w_n)$ is put to a softmax function, we get:

$$p(w_c|w_n) = \frac{e^{v_c \cdot v_w}}{\sum_{w=1}^{W} e^{(v_w \cdot v_c')}} \quad (2)$$

where v^w and v^c are the input and output vectors of word vocabulary W and context words C respectively. Now putting the probability of equation 2 in equation 1, we get:

$$\sum_{(w_n \in W, w_c \in C)} log(p(w_c|w_n))$$
$$= \sum_{(w_n \in W, w_c \in C)} (loge^{v_c \cdot v_w} - log \sum e^{v_c' \cdot v_w}) \quad (3)$$

Using the above described Skip-gram model (actual implementation is described in 4.3), a word embedding is obtained that encodes the semantic and syntactic information to its real-values vectors. The hyper-parameter for the model is as follows: Minimum word count = 3, window size= 3 and embedding dimension= 120.

3.2 Word Cluster Formation

To elevate the accuracy, cluster information is added as an additional feature to train the dataset using the K-means algorithm and the Skip-gram word vectors as shown in Figure 2 where each word is assigned to a specified cluster using the Euclidean distance similarity.

The K-means clustering is performed on a normalized word vectors X_{norm}, as cosine similarity and euclidean similarity are connected linearly and bears same result in clustering

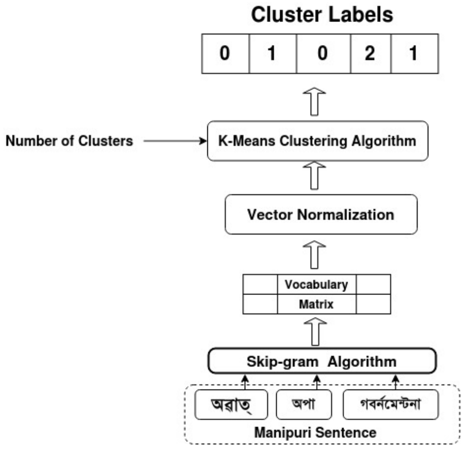

Figure 2: Word Clustering using K-Means Algorithm

(Qian et al., 2004). The normalized word vector is given by:

$$X_{Norm} = \frac{X}{max(X)} \quad (4)$$

With the normalized word vectors $X_{norm} = (x_{norm}^1, x_{norm}^2, ..., x_{norm}^n)$, each word is assigned to its appropriate k-cluster category using the Euclidean distance similarity measure which is given by:

$$d_{Euclidean}(p, q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \quad (5)$$

Where p and q are data points. Finally the objective function with k random data points chosen to be the initial centroid is given by:

$$argmin(\sum_{i=1}^{k} \sum_{v \in V} d_{Euclidean}(v, s_i)) \quad (6)$$

3.3 POS and Cluster Embedding

As neural network work with tensors, each additional feature needs to be in a matrix form, called embedding matrix where it relates each index of the object with its translation to tensors. Selecting a vector of a specific object can be translated into a matrix product in the following way:

$$v_i = \begin{cases} 1, if i \neq Object_{index} \\ 0, if i = Object_{index} \end{cases} \quad (7)$$

$$[Object_{vector}]_{Dim \times 1}$$
$$= [M]_{Dim \times words} \cdot [\vec{V}]_{(pos/cluster) \times 1} \quad (8)$$

Where $\vec{V}$ is the one-hot vector that determines which word needs to be translated. And M is the embedding matrix. Two matrices will be created each for POS and cluster feature.

3.4 LSTM Layer

In this Manipuri MNE recognition research, the recognition task is done using a bi-directional LSTM RNN. The purpose of choosing the LSTM network is because of its capability to overcome the diminishing gradient problem when the input sequence is large. In our MNE classification research, it has been found that the longest sequence is of 120 Manipuri words as shown in Figure 3 which led us to choose the LSTM RNN.

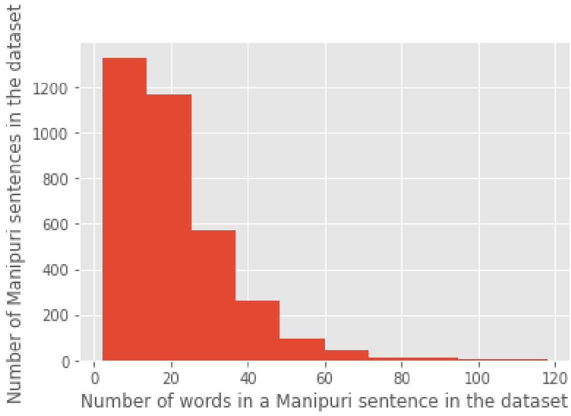

Figure 3: Manipuri Sentence length

LSTM is an Recurrent Neural Network (RNN) that works on a sequential data(Lample et al., 2016). A sequence of data) $\{w_1, w_2, ...w_n\}$ as input (Concatenation of Word, POS, Cluster and Affix vectors) and return another sequence $\{h_1, h_2, ...h_n\}$ that represents some information at every time step in the input which is given by:

$$h_t \quad = \quad lstm(h_{t-1}, [E(w_t)||e(w_t^{pos}, p_t)]) \quad (9)$$

$[E(w_t)||e(w_t^{pos}, p_t)] = x_t$ is the embedding where $E(w_t)$ is the word embedding for the word w_t using Skip-gram, $e(w_t^{pos})$ is the POS embedding and $e(p_t)$ cluster embedding for the Manipuri word w_t . The symbol $||$ in equation 9 represents concatenation of embedding vectors. The LSTM architecture is shown in 4.

LSTM(Reddy et al., 2018) consists of three gates that control the proportion of the input

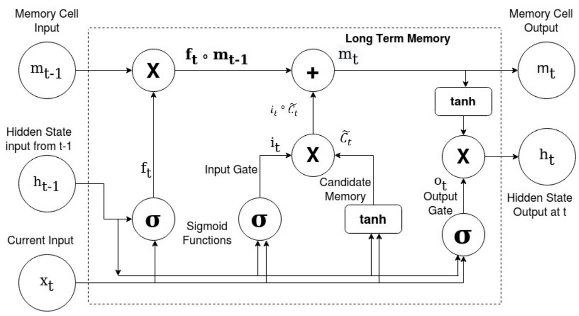

Figure 4: Long Short Term Memory Architecture

to give to the memory cells, and the proportion from the previous states to forget gate, that helps to overcome the diminishing gradient problem faced by RNN. At any given time t over Manipuri input sequence, three gates composite the LSTM unit cell:

1. An input gate i_t with the corresponding weight matrix: W_i and b_i, which is mathematically represented as:

$$i_t = \sigma(W_i[h_{t-1} + x_t] + b_i) \quad (10)$$

2. A forget gate f_t with corresponding weight matrix: W_f and b_f. Mathematically f_t is represented as:

$$f_t = \sigma(W_f[h_{t-1} + x_t] + b_f) \quad (11)$$

3. An output gate o_t with corresponding weight matrix: W_o and b_o. Mathematically o_t is represented as:

$$o_t = \sigma(W_o[h_{t-1} + x_t] + b_o) \quad (12)$$

where σ is the sigmoid function

All of these aforementioned gates are set to generate a certain state using the current input x_t, the state h_{t-1} from the previous step and current state of this cell, for the decisions whether to forget the memory stored, to take the input or to output the state generated. $\tilde{C}$ is the new candidate to be added to the new state which is given by:

$$\tilde{C}_t = \tanh(W_t[h_{t-1} + x_t] + b_c) \quad (13)$$

The current state c_t will be generated by calculating the weighted sum using both previous cell and current information generated by the current cell. The following equation provides the current state c_t:

$$c_t = i_t \tilde{C}_t + f_t . c_{t-1} \tag{14}$$

Finally, the hidden state h_t to the next LSTM unit is calculated as:

$$h_t = o_t \tanh(c_t) \tag{15}$$

As Manipuri is a context dependent language, it is beneficial to have access to the future and past contextual information which led to implementation of bidirectional LSTM modifier which is given by:

$$\overrightarrow{h_t} = lstm(\overrightarrow{h_{t-1}}, x_t))$$
$$\overleftarrow{h_t} = lstm(\overleftarrow{h_{t-1}}, x_t))$$

3.5 Output layer

The output layer of the model consist of Time-Distributed Dense function with softmax activation. This function allows us to apply the same function across every output over the time. Finally the softmax classifier calculates a probability distribution over the sequence labels.

4 Experimental Setup

4.1 Dataset

The dataset for Manipuri MWE classification is collected from a leading newspaper agency "The Sangai Express[1]" in Manipur with 76526 words. The appropriate POS and MNE tags are manually annotated and the dataset is split into training and testing set in 80:20 ratio. Further, 1 Million unannotated Manipuri dataset is used for the Skip-gram model training. Table 2 describes the dataset used in training and testing of the proposed Manipuri MWE model:

Details	Values
Number of Sentences	3504
Total number of words	76526
Number of distinct words	16297
Maximum number of words in a sentence	120
Multi-word Named Entities	12421

Table 2: Manipuri Dataset details

[1] https://www.thesangaiexpress.com/

4.2 Hyper-parameter details

For Manipuri MNE classification model, different hyper-parameters settings were used for bi-directional LSTM training as described in Table 3. As deep neural network tends to overfit, spatial dropout and recurrent dropout are used as regularization in the model whose value is described in the table 3 below:

Hyper-parameters	Values
Bi-LSTM units	100
Bi-LSTM batch size	32(Bengio, 2012)
Spatial1D Dropout	30%
Recurrent Dropout	10%
Activation Function	Sigmoid
Loss Function	binary cross-entropy
Optimizer	RMSprop Learning Rate= 0.0001, $\rho = 0.9$

Table 3: Manipuri MNE Classification Hyper-parameters

Ruder (2016) suggests the use of the Gradient Descent optimizer:RMSprop, to overcome the radically diminishing learning rates during training. Graves (2013) published the first use of RMSprop optimizer in recurrent neural network and suggest a default value of learning rate = 0.0001 and decay constant $\rho = 0.9$.

4.3 The Skip-gram model

The Skip-gram model as shown in figure 5, takes in a pair of inputs words for each training example ([input word w_i, target word c_i]) having unique identifier which is then passed to an embedding layer initialized with random weights. It is then passed to a merge layer

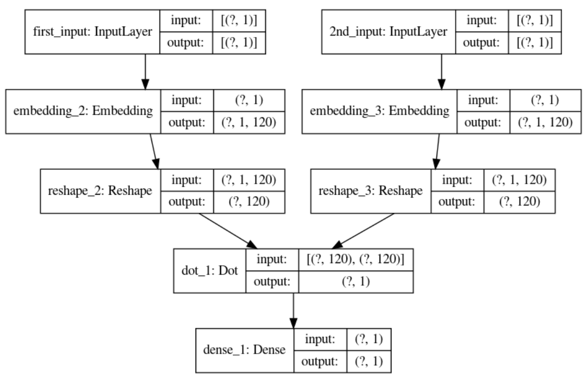

Figure 5: Skip-gram Model summary

to compute the dot product of these vectors where a sigmoid layer predicts the output

$$Y = \begin{cases} 1, if w_i \ and \ c_i \ is \ relevant \\ 0, if \ \ otherwise \end{cases} \tag{16}$$

The loss is leveraged by the mean squared error loss and performs back-propagation with each epoch to update the embedding layer.

5 Results and Evaluation

The main aim of this research is to focus on the addition of features and their effects on the final result of MNE classification as compared to the base models. We have used the Precision, Recall and F-Score evaluation metrics to measure the performance of the models.

5.1 Result of the Skip-gram model

Figure 6 shows the word similarity plot using Euclidean distance measure of the Skip-gram model. The circled areas contains some of

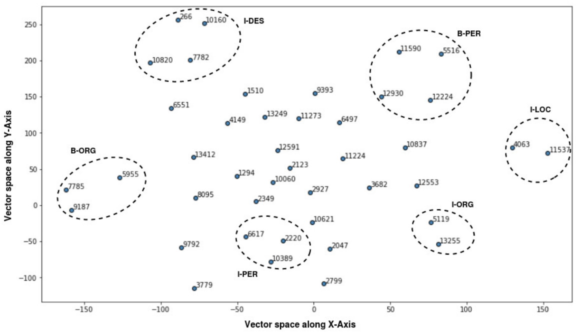

Figure 6: Euclidean Distance Similarity among words of Skip-Gram Model. The numbers in the figure represents index of the Manipuri MNE in the word vector

the MNE words identified by its index, that have similar euclidean distance, positioned near each other in the vector space.

Types of Relationship	Word	Similar word
	Word Pair	
Organization	কাউন্সিলগী Council-gi	পার্টিসু Party-su
Person Name	থৌনাউজম Thounaojam	লোকেন Loken
Location	টিডিম Tiddim	মোংবুং Moibung
Designation	মিনিষ্টরনা Minister-na	সেক্রেটরি Secretary

Table 4: Types of Word Relationship among word pairs

Table 4 shows the word-word similarity among the MNEs. The proposed Ski-gram model was found that the similarity of word representation go beyond the syntactic regularities with 89% of MNEs having found similarly distant in the vector space. The agglutinative nature of the language affects the Skip-gram model, as affixes define the inflection of the words (Bhat and Ningomba, 1997).

In the case of Time, Date and Currency MNEs, the Skip-gram model cannot find the similar word as these words are basically numbers and identifier words for Time পুং (poong), Currency লুপা (Lupa) and Date তাং (Tang) are unique and does not occur frequently in the dataset.

5.2 Optimal number of Clusters

The MNE classification models (as described in section 3) uses word cluster information generated from the Skip-gram model with K-means clustering algorithm. The clustering is to segregate the words into cluster and added as an additional feature to the LSTM model training. To select the optimal number of cluster for the k-means algorithm to perform clustering, the Silhouette analysis is performed for cluster number 3 to 10.

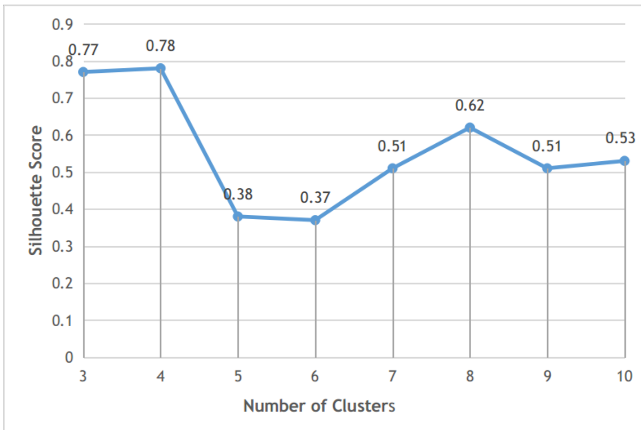

Figure 7: Silhouette Score for the clusters

From the Figure 7, it can be seen that the $s(4) = 0.78$ which is closer to 1, which led us to choose the number of clusters = 4.

5.3 MNE LSTM model Output

With the embedding weights generated from the Skip-gram model and the cluster information using the K-means algorithm, the MNE Bi-LSTM model is created. The result of the MNE classification using Bi-LSTM and Skip-gram embedding is shown in figure 8. The baseline model attained an average F-Score measure of 81.47% in classifying the MNE.

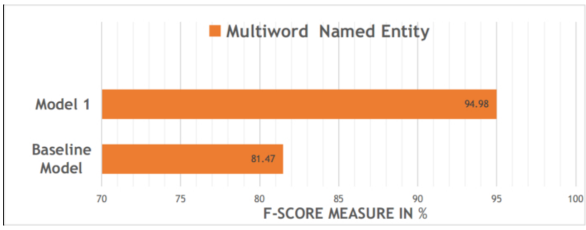

Figure 8: Average F-Score measure of Manipuri MNE models

With the addition of cluster information during the training, the MNE classification F-Score measures is increased to 94.98%. The fine grain classification report of the baseline model with cluster features is shown in Table 5.

MNE	P	R	F	Support
MW Person name	1.00	1.00	1.00	128
MW Location name	0.9408	0.9310	0.9358	52
MW Date/Time	0.9447	0.9212	0.9328	102
MW Organization Name	0.9213	0.9347	0.9306	43

Table 5: Fine grain Classification Report of the MWE Model using Skip-gram-LSTM where P= Precision, R= Recall and F= F-Score

The LSTM MNE model with cluster feature is able to recognize the multiword person name with 100% accuracy. The following are some of the errors encountered during testing of the model:

- The Manipuri transliterated English abbreviation were unable to be correctly recognized by the model as these words could not contribute enough to the semantic structure of the Skip-gram model from the context word.
- Most of the English multiword organization names were recognized incorrectly, which considerably decreased the accuracy, as the dataset, being a Manipuri News corpus contains fair amount of such type of words.

Overall, the model predicted the MNE tags of the words with state-of-the art accuracy considering the fact that no morphological rules or features were used in the model. Such model casts itself as break-through to Manipuri NLP computation where low resource is always a constraint.

5.4 Comparison with other MNE Classification Models

A comparative study is performed as described in Table 6. All the machine learning approaches for Manipuri MNE recognition such as SVM (Singh and Bandyopadhyay, 2010) and CRF Nongmeikapam and Bandyopadhyay (2010); Nongmeikapam et al. (2011a), use extensive morphological features such as surrounding words, word length, stemmed affixes, word counts, digit features, word frequency and NE tags.

Manipuri MWE Model	F-Score (%)
SVM (Singh and Bandyopadhyay, 2010)	MNE: 93.96%
CRF (Nongmeikapam and Bandyopadhyay, 2010)	72.24%
CRF (Nongmeikapam et al., 2011a)	92.40%
Bi-LSTM with Skip-gram Embedding	**MNE:94.98%**

Table 6: Comparison of Manipuri MWE classification Models

The F-Score measure of the proposed Manipuri MNE classification is calculated as the average of all the F-Score measures of the MNEs given in fine classification report table 5.

6 Conclusion

Finally, a deep neural model has been reported for Manipuri MNE recognition using Bi-LSTM and word embeddings in this paper. The training data consists of POS tagged words of 76526 with 12421 number of MNE. The Bi-LSTM model make use of embedding matrix generated using the Skip-gram for Manipuri Words. Word clusters information generated using the Skip-gram word vectors, is used as an additional feature. The use of K-means cluster information is to create a semantically meaningful MNE clusters. It has been reported that the cluster information generated from the Skip-gram embedding word vectors carries semantic values and has been able to supplement the MNE recognition task resulting in an increase in the average F-Score measure by 86% for MNE, as compared to the baseline model. The proposed Skip-gram model correctly represented the similarity of MNE words with about 89% accuracy. Overall the model achieved an average F-Score measure of 94.98%.

6.1 Impact on other research

As Manipuri is a low-resource language where properly tagged dataset is unavailable, it becomes crucial for algorithms that can function without the tagged dataset. This research provides a way on how Manipuri words can be partitioned using Skip-gram embedding and K-means clustering algorithm and can provide effective solution for researches such as Manipuri News clustering, Document clustering, sentiment analysis, hate speech detection.

References

SG Ajay, M Srikanth, M Anand Kumar, and KP Soman. 2016. Word embedding models for finding semantic relationship between words in tamil language. *Indian Journal of Science and Technology*, 9(45).

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. *CoRR*, abs/1206.5533.

Darbhe N Shankara Bhat and MS Ningomba. 1997. *Manipuri Grammar: XD-US....*, volume 4. Lincom Europa.

Paheli Bhattacharya, Pawan Goyal, and Sudeshna Sarkar. 2016. Using word embeddings for query translation for hindi to english cross language information retrieval. *Computación y Sistemas*, 20(3):435–447.

Sirajul Islam Choudhury, Leihaorambam Sarbajit Singh, Samir Borgohain, and Pradip Kumar Das. 2004. Morphological analyzer for manipuri: Design and implementation. In *Asian Applied Computing Conference*, pages 123–129. Springer.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Yoshiaki Kitagawa and Mamoru Komachi. 2017. Long short-term memory for japanese word segmentation. *arXiv preprint arXiv:1709.08011*.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Kishorjit Nongmeikapam and Sivaji Bandyopadhyay. 2010. Identification of mwes using crf in manipuri and improvement using reduplicated mwes. In *Proceedings of the International Conference on Natural Language Processing, India*.

Kishorjit Nongmeikapam and Sivaji Bandyopadhyay. 2011. Genetic algorithm (ga) in feature selection for crf based manipuri multiword expression (mwe) identification. *arXiv preprint arXiv:1111.2399*.

Kishorjit Nongmeikapam, Dhiraj Laishram, Naorem Bikramjit Singh, Ngariyanbam Mayekleima Chanu, and Sivaji Bandyopadhyay. 2011a. Identification of reduplicated multiword expressions using crf. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 41–51. Springer.

Kishorjit Nongmeikapam, Lairenlakpam Nonglenjaoba, Yumnam Nirmal, and Sivaji Bandhyopadhyay. 2011b. Improvement of crf based manipuri pos tagger by using reduplicated mwe (rmwe). *arXiv preprint arXiv:1111.2399*.

Kishorjit Nongmeikapam, Thiyam Ibungomacha Singh, Ngariyanbam Mayekleima Chanu, and Sivaji Bandyopadhyay. 2014. Manipuri chunking: An incremental model with pos and rmwe. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 277–286.

Gang Qian, Shamik Sural, Yuelong Gu, and Sakti Pramanik. 2004. Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237.

Aniketh Reddy, Monica Adusumilli, Saikiranmai Gorla, Lalita Neti, and Aruna Malapati. 2018. Named entity recognition for telugu using lstm-crf.

Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

Thoudam Doren Singh and Sivaji Bandyopadhyay. 2010. Web based manipuri corpus for multiword ner and reduplicated mwes identification using svm. In *Proceedings of the 1st Workshop on South and Southeast Asian Natural Language Processing*, pages 35–42.

Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. 2015. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *arXiv preprint arXiv:1510.06168*.

Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 981–986.