idrbt-team-a@IECSIL-FIRE-2018: Named Entity Recognition of Indian languages using Bi-LSTM

S. Nagesh Bhattu[1], N. Satya Krishna[2,3], and D. V. L. N. Somayajulu[3]

[1] NIT Tadepalligudem, West Godavari District, Andhra Pradesh, India
`nageshbhattu@nitandhra.ac.in`
[2] IDRBT, Road No.1 Castle Hills, Masab Tank, Hyderabad, Telangana, India
`satya.krishna.nunna@gmail.com`
[3] NIT Warangal, Telangana, India
`{soma}@nitw.ac.in`

Abstract. Named entity recognition(NER) is a key task in NLP pipeline useful for various applications such as search engines, question answering systems, sentiment analysis in domains ranging from travel, bio-medical text, newswire text, financial text etc. NER is effectively solved using sequence labeling approaches like HMM and CRF. Though, CRF (being discriminative) shows better performance compared to HMM, it uses discrete features and do not naturally capture semantic features. LSTM based RNNs can address this through their ability to deal with continuous valued features such as Word2Vec, Glove, etc. Another advantage of using LSTM lies in its ability to capture the long and short range dependencies through its novel gating structure. This work presents the deep learning based NER using special type of Recurrent Neural Network(RNN) called Bi-directional Long Short-Term Memory(Bi-LSTM). We use a two stage LSTM based network, one acting at character level capturing the n-gram patterns related to NER. Such features are crucial in NER for Indian languages as suffixes used in Indian languages often carry syntactic information. The character based emebeddings, word2vec embeddings and sequence based bi-LSTM embeddings together carry all the requisite features necessary for the NER prediction problem. We present the experimental results on two test datasets from each Indian language such as *hindi, kannada, malayalam, tamil* and *telugu*. The accuracies on test-1 datasets of *hindi, kannada, malayalam, tamil* and *telugu* languages are 97.82%, 97.04%, 97.46% 97.41% and 97.54% respectively. These are highest accuracy results given by this model when compared with all other models presented by competitors in this shared task [2]. The accuracies on test-2 datasets of *hindi, kannada, malayalam, tamil* and *telugu* languages are 97.82%, 96.79%, 96.58% 96.18% and 97.68% respectively. On test-2 dataset this model stood in first position for *hindi* language and second position for the remaining four languages. The shared task organizers released F-Scores for test-2 datasets of all languages. This model got 94.0%, 84.55%, 84.78%, 89.55% and 91.44% F-Scores on *hindi, kannada, malayalam, tamil* and *telugu* languages respectively. All these F-Scores are in second position compared with other models. In overall average accuracy and F-Score of this model on all these five Indian languages is 97.01% and 86.99% which are in second position.

Keywords: Bi-LSTM· Named Entity Recognition· Word embedding· Information Extraction· Sequence labeling.

1 Introduction

Named Entity Recognition(NER) is one of the major subtasks in the field of Information Extraction(IE). The objective of NER is to identify and classify the entities into pre-confined class names *person, location, event, number, organization, occupation, datenum, name* and *other* from the given unstructured text. For example consider the sentences shown in table-1. In both sentences, each word is annotated with a named entity tag. Identifying the entities in the

Table 1. Example of annotated sentences

Sentence-1	john	working	as	an	assistant	professor	in	IITD
NE tags	Person	other	other	other	occupation	occupation	other	organization
Sentence-2	Columbus	discovered	America	in	1492			
NE tags	Person	event	location	other	datenum			

unstructured text resolve many problems in different applications. For example news publishers can manage their large data, created and generated on a daily basis, by classifying them based on major places, events, organizations and people. It can be used in customer support systems for processing the customer feedbacks and also to improve the performance of searching algorithms on text data.

There has been a lot of work done in NER since 3 decades [7]. In early days most of the work done for *english* language text using hand-crafted rule-based techniques. Later, algorithms are developed using machine learning techniques such as HMM, CRF. Compared to HMM, CRF uses a discriminative model and hence it is able to perform better due to its generalizability of log-linear models based on maximum entropy.

Initially in 1995, the system development competition for NER task was introduced by 6^{th} Message Understanding Conference(MUC-VI) on news article data. Later different shared task events were conducted for NER in different languages. The CoNLL-2002 [8] and CoNLL-2003 [9] also conducted two shared tasks with same name *Language-Independent Named Entity Recognition.* They provided the dataset with 4 different named entity annotations such as *person, location organization* and *name.* IJCNLP-2008 conducted a shared task on five south and south asian languages such as *hindi, bengali, oriya, telugu* and *urdu.*

CoNLL-2002 [8] shared task organizers provided the datasets in the form of train, development and test data for two languages *spanish* and *duch.* CoNLL-2003 [9] organized the competition task on NER for *english* and *german* languages. They provided the dataset in four different files such as training, development, test and unlabeled corpus files for each language. The *english* language dataset was prepared by collecting text from the Reuters Corpus[4] having news articles presented in the period of one year from mid-1996 to mid-1997. The *german* dataset was prepared by collecting text from the ECI Multilingual Text Corpus[5] containing the news articles from *german* news papers.

Black et al. [3] presented two different approaches. One is modified Transformation based Learning(TBL) integrated with Named Entity classifier and another is decision tree induction based approach. He presented the F1-score of 67.49 and 56.43 on *spanish* and *duch* language test datasets respectively. McNamee et al. [6] applied a one-vs-rest multiclass classifier for NER using eight linear kernel binary SVM classifiers. Cucerzan et al. [5] proposed a statistical based language independent named entity recognizer using word internal information(i.e current word, prefix and suffix of the current word) and contextual information(i.e previous and next words of the current word) extracted from the given annotated training data.

In this task we applied a deep learning based language independent NER system which is built by integrating the Convolutional Neural Network(CNN) followed by Bi-directional Long short-term Memory(Bi-LSTM). This system utilizes the character and word level informations which are extracted from unannotated corpus. We use this information in the form of two real valued vectors(aka word embeddings) as input features to NER to classify the entities in a given text. We experimented our model on five datasets of different Indian languages such as *hindi, kannada, malayalam, tamil* and *telugu*. In results section, we presented the performance results of our system on two test datasets from each language.

2 Approach

NER is a sequence labeling task, in which we predict the label for each word in a sequence of words. Applying traditional machine learning methods for text classification require more feature engineering. Though, the performance of rule-based NER is good, it requires more language dependent hand-crafted rules for classification. To reduce the feature engineering overhead, in our approach we applied the deep-learning based methods. As shown in the figure-1 first we build the feature vectors from two sources of information namely word level and character level. The pre-trained word feature vectors[6] are used in our approach. The word feature vectors for new words (which are not having pre-trained vectors) from the given dataset are built using the model. Character-to-word feature vectors are built using Bi-LSTM to capture the character level information from character sequences in each word of the dataset. We replaced each word in an input sentence of Bi-LSTM with a word embedding created by concatenation of the feature vectors corresponding to that word. Later, we predicted the label sequence for each sentence using these word embeddings.

[4] http://reuters.com/researchandstandards/

[5] http://www.idc.upenn.edu

[6] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

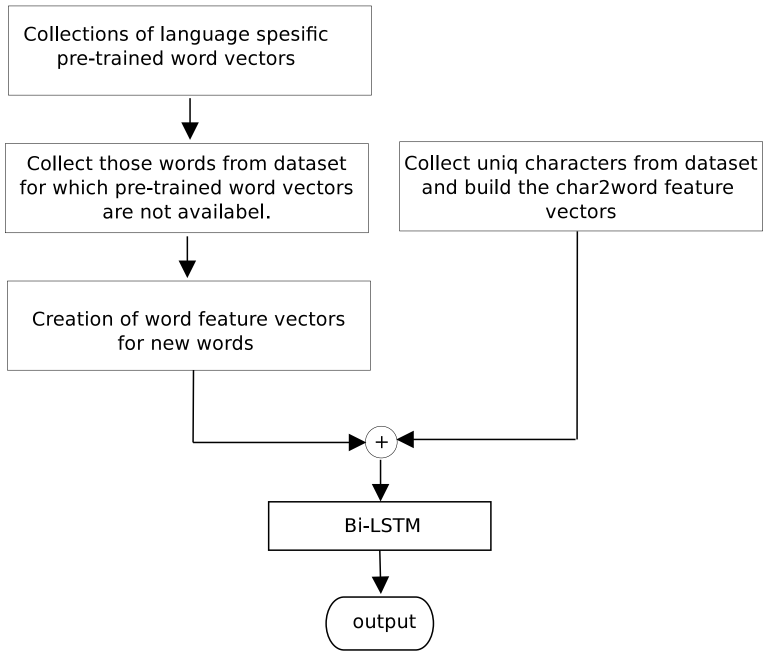
Collections of language spesific pre-trained word vectors
Collect those words from dataset for which pre-trained word vectors are not availabel.
Collect uniq characters from dataset and build the char2word feature vectors
Creation of word feature vectors for new words
+
Bi-LSTM
output

Fig. 1. Overview of approach

3 Experiments

In this section we described the computation of word embeddings, experimental details using an approach described in the previous section, and results on ten test sets(two test sets from each language dataseet).

3.1 Building word Feature Vectors

This model uses the word embedding as input feature in place of a word in the given input sentence. These word embeddings are created by combining the word feature vector and char to word feature vector. These word feature vectors are built using skip-gram model as defined in [4]. In skip-gram model, the word feature vector V_{w_i} is learned on large training corpus by computing the statistics of occurrence of its context words given the word w_i throughout all sentences in training data. For example a large corpus represented as a sequence of words $w_0, w_1, w_2, w_3, ...w_L$ in the training data, then it computes the log-likelihood using equation-1.

$$\frac{1}{\mathbf{L}} \sum_{l=1}^{L} \sum_{-c \leq j \leq c, j \neq i} \log \phi(w_{i+j}/w_i) \tag{1}$$

Here, c indicates the context window size. w_i is the current word in the sequence and w_{i+j} is a context word to w_i with j distance from its position in context window c. Here, ϕ is the probability denoting the occurrence of context word w_{i+j} given w_i. We compute the ϕ value using softmax function given in equation-2. Here, v_o is output feature vector corresponding to the w_o and v_i is feature vector corresponding to input word w_i. V is the size of vocabulary in the given corpus.

$$\phi(w_{i+j}/w_i) = \phi(w_o/w_i) = \frac{\exp(v_o^T v_i)}{\sum_{u=1}^{V} \exp(v_u^T v_{u_i})} \tag{2}$$

3.2 Problem statement and Bi-LSTM model

As our consideration, the NER is a sequence labeling task, we represented each input sentence as a sequence of words $w_0, w_1, w_2, ...w_n$ and its corresponding output label sequence is represented as $t_0, t_1, t_2, ...t_n$. Instead of using word sequence directly as input to the Bi-LSTM model we replaced each word with its corresponding word-embedding in the input sequence and then fed it to the Bi-LSTM model. The size of word-embedding in our experiment is 450 dimensions(i.e 300 word vector dimensions + 150 char to word vector dimensions). Here we implemented the Bi-LSTM cell using two Long Short-Term Memory(LSTM) cells. One LSTM cell scans the input vector sequence in forward direction and another LSTM cell scans the input vector sequence in backward direction. There are different versions of LSTM cells are defined. Among these we applied the following LSTM cell as shown in the following equation-3.

$$x_t = \sigma(X.[u_t, h_{t-1}] \oplus b_x) \tag{3a}$$
$$y_t = \sigma(Y.[u_t, h_{t-1}] \oplus b_y) \tag{3b}$$
$$o_t = \sigma(O.[u_t, h_{t-1}] \oplus b_o) \tag{3c}$$
$$\tilde{c_t} = \tanh(S.[u_t, h_{t-1}] \oplus b_s) \tag{3d}$$
$$S_t = x_t \otimes \tilde{c_t} \oplus S_{t-1} \otimes o_t \tag{3e}$$
$$h_t = y_t \otimes \tanh(s_t) \tag{3f}$$

Here, u_t denotes the input word embedding corresponding to the t^{th} position word w_t in the given input sequence. y_t denotes the predicted label corresponding to the word w_t. h_t denotes the hidden state vector. s_t represents the LSTM cell state vector. The weight matrices X and Y are used by LSTM cell in its input and output layers. Another weight matrices O and S are used by the LSTM in its forget layer and context layers respectively. In the above equations $\oplus$ denotes the element wise vector addition and $\otimes$ denotes the element wise vector product operation.

3.3 Dataset

Arnekt-IECSIL@FIRE2018 shared task[2] organizers provided five datasets[1] for competitors on five Indian languages such as *hindi, kannada, malayalam,* *tamil* and *telugu*. Each language dataset has three different files with text in its corresponding language script. Among these three files, one is training file having data in two column format with label in second column for each word in a sentence. The sentences are separated with a new line labeled as *newline*. Remaining two are test-1 and test-2 files having data same as training file without labels. Each test file has 20% of data among the overall dataset. Except *kannada* dataset remaining datasets are having sufficient number of sentences and words for learning word feature vectors using deep learning methods. The training file in each dataset has nine distinct labels such as *datenum, number, person, occupation, organization, location, name, things* and *other*. The table-2 describes the detail description regarding the number of sentences, number of words, number of unique words in each file of all datasets.

Table 2. Datasets description

Dataset	**File type**	**File Length**	**# Sentences**	**# words**	**# unique words**
Hindi	Train	1548570	76537	1472033	87842
	Test-1	519115	25513	493602	43797
	Test-2	517876	25513	492363	44642
Kannada	Train	318356	20536	297820	73712
	Test-1	107325	6846	100479	34200
	Test-2	107010	6846	100164	34040
Malayalam	Train	903521	65188	838333	143990
	Test-1	301860	21730	280130	67361
	Test-2	302232	21730	280502	67274
Tamil	Train	1626260	134030	1492230	185926
	Test-1	542225	44677	497548	89529
	Test-2	544183	44677	499506	89493
Telug	Train	840904	63223	777681	108059
	Test-1	280533	21075	259458	51555
	Test-2	279443	21075	258368	51294

3.4 Results

As per evaluation procedure given by the Shared task organizers, our model is evaluated using these metrics.

$$Accuracy = \frac{No.Of\ words\ are\ assigned\ with\ the\ correct\ label}{No.Of\ words\ in\ the\ dataset} \tag{4}$$

$$Precision(P_i) = \frac{No.Of\ words\ are\ correctly\ labeled\ with\ label_i}{No.Of\ words\ are\ labeled\ with\ label_i} \tag{5}$$

$$Recall(R_i) = \frac{No.Of\ words\ are\ correctly\ labeled\ with\ label_i}{Total\ No.Of\ words\ with\ label_i\ in\ test\ data} \tag{6}$$

$$fscore(F_i) = \frac{2 * P_i * R_i}{P_i + R_i} \tag{7}$$

$$Overall\ fscore(F) = \frac{1}{|L|} * \sum_{iinL} F_i \tag{8}$$

The table-3 summarizes the accuracy on testset-1 and 2 of five Indian language datasets. In Pre-Evaluation on all language datasets, this model shown the high performance compared to the all other models presented in competition. In Final-Evaluation this model got the first position in *hindi* language , secodn position in *malayalam, tamil* and *telugu* languages and third position in *kannada* language.

Table 3. Test accuracies in 5 languages

	Hindi	**Kannada**	**Malayalam**	**Tamil**	**Telugu**
Pre-Evaluation(Testset-1)	97.82	97.04	97.46	97.41	97.54
Final-Evaluation(Testset-2)	97.82	96.79	96.58	96.18	97.68

Table 4. F1-Scores in 5 languages

	Hindi	**Kannada**	**Malayalam**	**Tamil**	**Telugu**
Final-Evaluation(Testset-2)	85.9	84.55	84.78	89.55	91.44

The table-4 summarizes the F1-Scores of Final-Evaluation. This model given second highest performance in F-Score on *kannada, malayalam, tamil* and *telugu* language datasets. The actual F-score for *hindi* dataset is nearly 94.00%. But the F-score given in table-4 for *hindi* language is 85.9% according to the results given by the shared task organizers. The reason for less F-Score is typographical mistake done by us while converting the predicted label index with its corresponding label name *datenum* in the results file. Due to the zero F-Score for *datenum* label, the overall F-Score of this model is reduced to 85.9%. We figure-out this mistake after announcement of results by the shared task organizers.

4 Conclusion

As a part of competitive participation in Arnekt-IECSIL@FIRE2018 shared task, in this paper we have presented a NER system implemented using deep learning methods, in which we consider the pre-trained word vectors and character-to-word vectors as features. We have presented the experimental results on five Indian language datasets.

References

1. Barathi Ganesh, H.B., Soman, K.P., Reshma, U., Mandar, K., Prachi, M., Gouri, K., Anitha, K., Anand Kumar, M.: Information extraction for conversational systems in indian languages - arnekt iecsil. In: Forum for Information Retrieval Evaluation (2018)
2. Barathi Ganesh, H.B., Soman, K.P., Reshma, U., Mandar, K., Prachi, M., Gouri, K., Anitha, K., Anand Kumar, M.: Overview of arnekt iecsil at fire-2018 track on information extraction for conversational systems in indian languages. In: FIRE (Working Notes) (2018)
3. Black, W.J., Vasilakopoulos, A.: Language-independent named entity classification by modified transformation-based learning and by decision tree induction. In: Proceedings of CoNLL-2002. pp. 159–162. Taipei, Taiwan (2002)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
5. Cucerzan, S., Yarowsky, D.: Language independent ner using a unified model of internal and contextual evidence. In: Proceedings of CoNLL-2002. pp. 171–174. Taipei, Taiwan (2002)
6. McNamee, P., Mayfield, J.: Entity extraction without language-specific resources. In: Proceedings of CoNLL-2002. pp. 183–186. Taipei, Taiwan (2002)
7. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007)
8. Tjong Kim Sang, E.F.: Introduction to the conll-2002 shared task: Language-independent named entity recognition. In: Proceedings of CoNLL-2002. pp. 155–158. Taipei, Taiwan (2002)
9. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the conll-2003 shared task: Language-independent named entity recognition. In: Daelemans, W., Osborne, M. (eds.) Proceedings of CoNLL-2003. pp. 142–147. Edmonton, Canada (2003)