SSN_NLP@IECSIL-FIRE-2018: Deep Learning Approach to Named Entity Recognition and Relation Extraction for Conversational Systems in Indian Languages

D. Thenmozhi, B. Senthil Kumar, and Chandrabose Aravindan

Department of CSE, SSN College of Engineering, Chennai
{theni_d,senthil,aravindanc}@ssn.edu.in

Abstract. Named Entity Recognition (NER) focuses on the classification of proper nouns into the generic named entities (NE) such as person_names, organizations, locations, currency and dates. NER has several applications like conversation systems, machine translation, automatic summarization and question answering. Relation Extraction (RE) is an information extraction process used to identify the relationship between NEs. RE is very important in applications like short answer grading, conversation systems, question answering and ontology learning. NER and RE in Indian languages are difficult tasks due to their agglutinative nature and rich morphological structure. Further, developing language independent framework that supports all Indian Languages is a challenging task. In this paper, we present a deep learning methodology for both NER and RE in five Indian languages namely Hindi, Kannada, Malayalam, Tamil and Telugu. We proposed a common approach that works for both NER and RE tasks. We have used neural machine translation architecture to implement our methodology for these tasks. Our approach was evaluated using the data set given by IECSIL@FIRE2018 shared task. We have evaluated on two sets of data for NER task and obtained the accuracies as 94.41%, 95.23%, 95.97% and 96.02% for the four variations on pre-evaluation test set and 95.9%, 95.85% and 95.05% for the three runs on final-evaluation test set. Also, for RE task, we have obtained the accuracies as 56.19%, 60.74%, 60.7%, 75.43% and 79.11% for our five variations on pre-evaluation test set and 79.44%, 76.01% and 61.11% for Run 1, Run 2 and Run 3 respectively on final-evaluation test set.

Keywords: Named Entity Recognition (NER) · Relation Extraction · Information Extraction · Text mining · Deep Learning · Indian Languages.

1 Introduction

Named Entity Recognition (NER) is an Information Extraction (IE) task which identifies and classifies the proper names in the text into predefined classes such

as names of persons, organizations, dates, locations, numbers and currency. NER is very important for many NLP applications such as machine translation, text summarization, question answering and short answer grading. NER is very popular since 1970 in English and in other European languages. Several approaches have been reported for NER in these languages. Deep learning methods have also been employed for English, European and Chinese languages [5, 9, 7, 19]. However, NER for Indian languages is a very challenging task due to the characteristics namely no closed set vocabulary, no concept of capitalization, polysemy and ambiguity. Also, due to the complex morphology structure and agglutinative nature, IE in Indian languages is still an open challenge. Several methodologies including rule based, statistical and machine learning approaches have been reported for NER in Indian languages. However, the approaches are language dependent and no Indian language except Bengali reported above 90% F-score [18]. Moreover, developing language independent framework that supports all Indian Languages is a challenging task.

Relation Extraction (RE) is a process of extracting the relationships between NEs. It is also an IE task which extracts and classifies the relationship between entities. For example, "lives in" is the relationship present between person and location. RE is so important for applications such as ontology learning, conversational systems, question answering and short answer grading. Several methods are reported in literature to learn the relations automatically from English documents [22, 24, 14, 16, 8, 25]. Many of them are domain dependent [8, 25] and a few are domain independent approaches [22, 14]. They used approaches like rule-based, supervised and unsupervised to learn taxonomic [15, 23] or semantic relations [8, 17] between the entities. Only a very few approaches are presented that learn both taxonomic and semantic relations for any domain [22, 26]. Deep learning methods are also employed for RE in recent years [10, 13, 4]. However, RE for Indian languages [20] is still an open challenge. Further, developing language independent framework that supports all Indian Languages for extracting relations is a challenging task.

The shared task IECSIL@FIRE2018 [3] focuses on IE for conversational systems in Indian languages namely Hindi, Kannada, Malayalam, Tamil and Telugu. This shared task focuses on two sub-tasks namely NER task and RE tasks. The goal of IECSIL task is to research and develop techniques to extract information using language independent framework. IECSIL@FIRE2018 is a shared Task on information extractor for conversational systems in Indian languages collocated with FIRE-2018 (Forum for Information Retrieval Evaluation). This paper focuses on both sub tasks of IECSIL@FIRE2018 namely NER and RE. We propose a common approach that identifies and classifies the NEs to one of the generic classes namely name, occupation, location, things, organization, datenum, number and other, and also classifies the relations between NEs to one of the classes namely information_1, information_2, information_3, information_4, information_per, information_quant, information_closed, information_so, information_neg, information_cc, action_1, action_2, action_3, action_per, action_so, action_quant, action_neg, and other.

2 Proposed Methodology

We have used a deep learning approach based on Sequence to Sequence (Seq2Seq) model [21, 6]. We have utilized a common approach that addresses both NER and RE problems. We have adopted the Neural Machine Translation (NMT) framework [12, 11] based on Seq2Seq model for both NER and RE tasks. Figures 1 and 2 depict the flow of our approach for NER and RE tasks respectively.

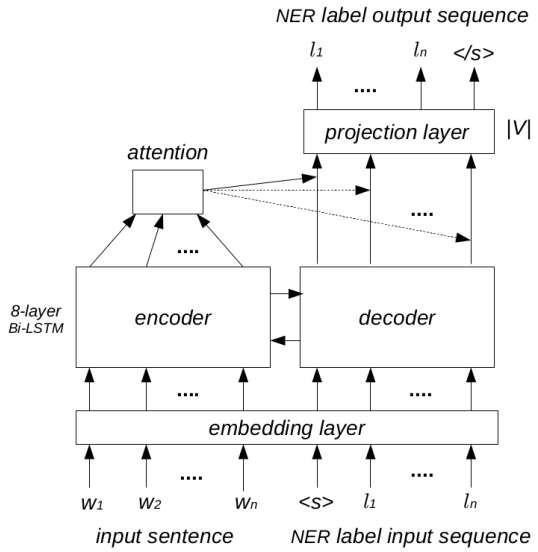

Fig. 1. System Architecture for NER.

The steps used in our approach are given below.

- Preprocess the given data
  - Convert set of tokens into sentences to obtain the input sentences and NER / RE label input sequences.
  - Split the given training sequences into training and development sets
  - Determine vocabulary from both input sentences and NER / RE label input sequences.
- Build a seq2seq deep neural network with the following layers.
  - embedding layer
  - encoding layer
  - decoding layer
  - projection layer
  - loss layer
- Predict the NER / RE label output sequences for the test sequences
- Tokenize the NER label output sequences to obtain the NE classes.

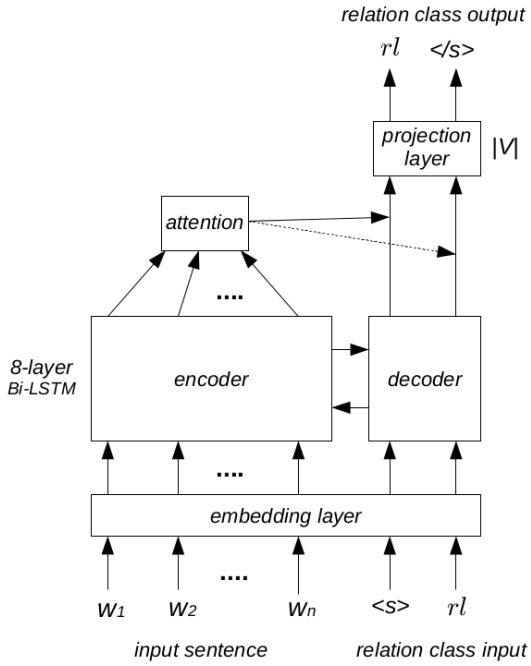

Fig. 2. System Architecture for Relation Extraction.

The steps are detailed below.

The given text consists of tokens of the sentences and their corresponding NEs for NER task and consists of the input sentences and their corresponding relation labels for RE task. Sample inputs for both tasks are given in Figures 3 and 4.

```
இவர்களில் other
பெண்கள் name
481 number
பேரும் location
ஆண்கள் other
457 number
பேரும் location
உள்ளனர் other
newline
```

Fig. 3. Input Data for NER.

We have prepared the data in such a way that Seq2Seq deep learning algorithm may be applied. The input sentences and NER / RE label input sequences are constructed separately based on the delimiter “newline”. For example, for the

बहमनी सल्तनत के अन्दर बीजापुर एक प्रान्त था information_1
विद्या चौधरी , भारत के उत्तर प्रदेश की पंद्रहवी विधानसभा सभा में विधायक रहे रहीं action_1

Fig. 4. Input Data for RE.

NER task, the above sequence of tokens are converted to input sentences and NER label input sequences as shown in Figure 5 and Figure 6 respectively.

இவர்களில் பெண்கள் 481 பேரும் ஆண்கள் 457 பேரும் உள்ளனர்

Fig. 5. Input Sentence for NER.

other name number location other number location other

Fig. 6. NER Label Input Sequence.

Similarly, for RE task, the input data is converted to input sentences and relation label output, as shown in Figure 7 and Figure 8.

Then the input sentences and NER / RE label input sequences are splitted into training sets and development sets. The vocabulary for both input sentences and NER / RE label input sequences are determined. For NER task, the input sentence with n words $w_1, w_2, ...w_n$ and NER label input sequence with n labels $l_1, l_2, ...l_n$ are given to the embedding layer. However, for RE task, the input sentence with n words $w_1, w_2, ...w_n$ and relation label input one label rl are given to the embedding layer.

To build a deep neural network model, a multi-layer RNN (Recurrent Neural Network) with LSTM (Long Short Term Memory) as a recurrent unit is used. This neural network consists of several layers namely, embedding layer, encoding layer, decoding layer, projection layer or softmax layer and loss layer. The embedding layer learns weight vectors from the input sentence and NER / RE label input sequence based on their vocabulary. These embeddings are fed into multi-layer LSTM where encoding and decoding are performed. The word embedding vector x_{wi} for each word w_i, where w_i constitutes a time step, is the input to LSTM network. The computation of the hidden layer at time t and the output can be represented as follows.

$$i_t = \sigma(w_x^{(i)} x + w_h^{(i)} h_{t-1} + b^{(i)}) \tag{1}$$

$$f_t = \sigma(w_x^{(f)} x + w_h^{(f)} h_{t-1} + b^{(f)} + 1) \tag{2}$$

बहमनी सल्तनत के अन्दर बीजापुर एक प्रान्त था
विद्या चौधरी , भारत के उत्तर प्रदेश की पंद्रहवी विधानसभा सभा में विधायक रहे रहीं

Fig. 7. Input Sentence for RE.

```
information_1
action_1
```

Fig. 8. Relation Label Input.

$$o_t = \sigma(w_x^{(o)}x + w_h^{(o)}h_{t-1} + b^{(o)}) \tag{3}$$

$$\widetilde{c}_t = tanh(w_x^{(c)}x + w_h^{(c)}h_{t-1} + b^{(c)}) \tag{4}$$

$$c_t = f_t \circ \widetilde{c}_{t-1} + i_t \circ \widetilde{c}_t \tag{5}$$

$$h_{b/f} = o_t \circ tanh(c_t) \tag{6}$$

where w_s are the weight matrices, h_{t-1} is the hidden layer state at time $t-1$, i_t, f_t, o_t are the input, forget, output gates respectively at time t, and $h_{b/f}$ is the hidden state of backward, forward LSTM cells. To have the better efficiency of LSTM, the bias value in the forget gate is set to a default value 1.

The attention mechanism [1, 11] is used to handle the longer sentences. Softmax layer or projection layer is a dense layer to obtain the NER / RE label output sequence. Loss layer is used to compute the training loss during model building. Once, the model is built, the NER / RE label output sequences are obtained by using the model for sequence mapping.

The target sequences we have obtained are the sequences of NER labels with respect to the given sentence. Thus, the NER label sequences are tokenized further to obtain the NE classes for each term. However, for RE task, the target sequence is itself a RE label and thus it not required to do any post process the output of our deep neural network.

3 Implementation

Our methodology was implemented using TensorFlow for IECSIL Shared Tasks namely NER and RE. The data set [2] used to evaluate the NER and RE tasks consists of a training set and two test sets namely pre-evaluation set and final-evaluation set for five Indian languages namely “Hindi”, “Kannada”, “Malayalam”, “Tamil” and “Telugu”. The details about the NER and RE data are given in Tables 1 and 2.

The input sentences and NER / RE label input sequences are constructed based on the delimiter “newline”. We have splitted these sequences into train set and development set to feed into the deep neural network. The details of the splits are given in Tables 3 and 4 for NER and RE tasks respectively.

Table 1. Data Set for IECSIL NER Task

Languages	No. of Tokens		
	Training	Pre-evaluation Test set 1	Final Evaluation Test set 2
Hindi	1548570	519115	517876
Kannada	318356	107325	107010
Malayalam	903521	301860	302232
Tamil	1626260	542225	544183
Telugu	840908	280533	279443

Table 2. Data Set for IECSIL RE Task

Languages	No. of Instances		
	Training	Pre-evaluation Test set	Final Evaluation Test set
Hindi	56775	18925	18926
Kannada	6637	2213	2213
Malayalam	28287	9429	9429
Tamil	64833	21611	21612
Telugu	37039	12347	12347

Table 3. Number of Sequences for NER Model Building

Languages	Training	Development
Hindi	57986	18532
Kannada	15500	5036
Malayalam	50000	15188
Tamil	100000	34030
Telugu	46000	17223

Table 4. Number of Sequences for RE Model Building

Languages	Training	Development
Hindi	40000	16775
Kannada	4500	2137
Malayalam	20000	8287
Tamil	45000	19833
Telugu	27000	10039

We have used TensorFlow code based on tutorial code released by Neural Machine Translation [1] [11] that was developed based on Sequence-to-Sequence (Seq2Seq) models [21, 1, 12] to implement our deep learning approach for NER and RE tasks. We have implemented several variations of the Seq2Seq model by varying the directionality, depth and number of training steps with a dropout of 0.2 and batch size of 128 to show the effectiveness of our proposed model of 8-layer, bi-LSTM with attention.

The implementation details of model building for both NER and RE tasks are explained below.

3.1 NER Models

We has used 4 variations of model building for NER task. The variations are given below.

- Model 1: 2 layer, uni-directional LSTM, without attention, 50,000 steps
- Model 2: 4 layer, uni-directional LSTM, with scaled-luong attention, 75,000 steps
- Model 3: 8 layer, bi-directional LSTM, with scaled-luong attention, 75,000 steps
- Model 4: 8 layer, bi-directional LSTM, with scaled-luong attention, 1,00,000 steps

The development bleu scores obtained for these variations are given in Table 5.

Table 5. Development Bleu Scores for NER

Languages	Model 1	Model 2	Model 3	Model 4
Hindi	93.5	94.0	94.3	94.6
Kannada	90.8	91.1	91.6	91.9
Malayalam	91.3	91.6	90.8	91.0
Tamil	91.0	91.1	91.2	91.6
Telugu	92.7	92.8	93.7	94.1

It is observed from Table 5 that except for the "Malayalam" language, bi-directional LSTM with attention having 8 layers depth and more number of training steps works well for all other languages.

3.2 RE Models

We have implemented a total of five variations for finding the relation labels for the given sentences to show the effectiveness of our proposed model. The first

[1] https://github.com/tensorflow/nmt

three variations use machine learning approaches and the last two used the deep learning approach.

The three variations using machine learning approach are given below.

- Model 1: Term frequency vectorizer
- Model 2: TF-IDF vectorizer by ignoring the terms with document frequency less than 1
- Model 3: TF-IDF vectorizer by ignoring the terms with document frequency less than 2

For these machine learning approaches, the bag of word features are extracted from the training instances. We have used Scikit–learn machine learning library to vectorize the training instances and to implement the classifier for the relation extraction and classification task.

In the first variation, CountVectorizer of sklearn is used for vectorization. Term Frequency - Inverse Document Frequency (TF-IDF) is used for vectorization with min_df as 1 and 2 in the second and third variations respectively. min_df is to build the vocabulary by ignoring terms that have a document frequency lower than the given value. TfidfVectorizer of sklearn is used for these two variations. We have used neural network classifier with Stochastic Gradient Descent (SGD) optimizer to classify the relation labels.

The two variations using deep learning approach for RE are given below.

- Model 4: 4 layer, uni-directional LSTM, with scaled-luong attention, 50,000 steps
- Model 5: 8 layer, bi-directional LSTM, with scaled-luong attention, 75,000 steps

The development accuracy scores obtained for these two variations are given in Table 6.

Table 6. Development Accuracy Scores for RE

Languages	Model 4	Model 5
Hindi	91.8	91.1
Kannada	34.0	53.3
Malayalam	77.3	81.1
Tamil	81.8	85.0
Telugu	85.9	84.2

It is observed from Table 6 that except for the "Telugu" language, bi-directional LSTM with attention having 8 layers depth performs well for all other languages in RE task.

4 Results

We have evaluated our models for the data set provided by IECSIL shared task. The results obtained for both NER and RE tasks are discussed in this section.

4.1 NER Results

Table 7 shows the accuracies we have obtained for the pre-evaluation test data using our models for NER task.

Table 7. Pre-Evaluation Test Data Accuracies for NER

Languages	Model 1	Model 2	Model 3	Model 4
Hindi	94.97	95.96	96.20	96.73
Kannada	93.30	94.00	95.49	95.63
Malayalam	95.21	95.66	95.87	95.44
Tamil	93.42	95.26	95.54	95.55
Telugu	95.14	95.28	96.77	96.75
Average	94.41	95.23	95.97	96.02

We have obtained the accuracies as 94.41%, 95.23%, 95.97% and 96.02% for Model 1, Model 2, Model 3 and Model 4 respectively on the pre-evaluation test data.

We have submitted three runs based on our three models namely Model 4, Model 3 and Model 2 as Run 1, Run 2 and Run 3 respectively for the task. Table 8 shows the accuracies we have obtained for the final-evaluation test data using our three models. We have obtained the accuracies as 95.9%, 95.85% and 95.05% for Run 1, Run 2 and Run 3 respectively.

Table 8. Final-Evaluation Test Data Accuracies for NER

Languages	Run 1	Run 2	Run 3
Hindi	96.68	96.51	95.95
Kannada	95.8	95.76	94.21
Malayalam	95.28	95.28	95.05
Tamil	94.91	94.9	94.66
Telugu	96.81	96.81	95.4
Average	95.9	95.85	95.05

It is observed from Table 8 that bi-directional LSTM with attention having 8 layers depth works well for five Indian languages. However, increase in just step size does not show significant improvement.

Table 9 shows the F1-scores we have obtained for the final-evaluation test data using our three runs. This table shows the F1-score for the individual classes of all five languages. It is observed from the table that we have obtained less F1-score for “datanum” class and high F1-score for “other” class. This may be due to low recall value for “datanum” class and low precision value for “other” class. Also, we have obtained a overall F1-score for all the classes is low for Kannada language. This is due to the size of the dataset which is lesser for Kannada when compared with all the other languages.

Table 9. Final-Evaluation Test Data F1-Scores for NER

Runs	Languages	datenum	event	loc.	name	number	occ.	org.	other	things
Run 1	Hindi	88	93	96	83	93	92	88	98	87
	Kannada	20	57	86	70	88	89	72	98	38
	Malayalam	20	60	86	76	94	86	78	97	77
	Tamil	77	89	92	79	96	91	82	97	89
	Telugu	64	94	97	85	97	93	78	98	94
	Average	53.8	78.6	78.6	78.6	93.6	90.2	79.6	97.6	77
Run 2	Hindi	86	92	96	82	93	91	87	98	86
	Kannada	22	49	86	70	87	88	77	98	40
	Malayalam	20	60	86	76	94	86	78	97	77
	Tamil	79	89	91	79	96	91	82	97	89
	Telugu	63	94	97	85	97	93	78	98	93
	Average	54	76.8	91.2	78.4	93.4	89.8	80.4	97.6	77
Run 3	Hindi	89	84	95	80	90	86	84	98	78
	Kannada	9	72	79	63	62	76	69	97	43
	Malayalam	30	71	85	76	92	84	78	97	80
	Tamil	78	84	93	78	94	87	78	97	83
	Telugu	55	73	96	81	92	85	62	97	66
	Average	52.2	76.8	89.6	75.6	86	83.6	74.2	97.2	70

4.2 RE Results

Table 10 shows the accuracies we have obtained for the pre-evaluation test data using our models.

Table 10. Pre-Evaluation Test Data Accuracies for RE

Languages	Model 1	Model 2	Model 3	Model 4	Model 5
Hindi	68.66	70.45	70.28	92.16	93.24
Kannada	36.15	46.26	46.54	43.0	51.20
Malayalam	44.71	52.17	51.98	77.18	81.89
Tamil	65.98	67.41	67.35	80.53	85.91
Telugu	65.43	67.41	67.35	84.29	83.29
Average	56.19	60.74	60.7	75.43	79.11

We have obtained the accuracies for the pre-evaluation test data as 56.19%, 60.74%, 60.7%, 75.43% and 79.11% for Model 1, Model 2, Model 3, Model 4 and Model 5 respectively.

We have submitted three runs based on our three models namely Model 5, Model 4 and Model 2 as Run 1, Run 2 and Run 3 respectively for the task. Table 11 shows the accuracies we have obtained for the final-evaluation test data using our three models. We have obtained the accuracies as 79.44%, 76.01% and 61.11% for Run 1, Run 2 and Run 3 respectively.

Tables 10 and 11 show that bi-directional LSTM with attention having 8 layers depth performs better for all the languages except “Telugu” language.

Table 11. Final-Evaluation Test Data Accuracies for RE

Languages	Run 1	Run 2	Run 3
Hindi	92.99	91.71	69.04
Kannada	51.87	45.14	49.43
Malayalam	81.99	75.25	51.8
Tamil	86.26	82.19	67.12
Telugu	84.11	85.78	68.17
Average	79.44	76.01	61.11

Table 12 and Table 13 show the F1-scores we have obtained for the final-evaluation test data using our three runs for relation extraction task. These tables show the F1-scores for the individual classes of all five languages in which "A_" and "I_" indicate "Action_" and "Information_" classes respectively. It is observed from the tables that we have obtained a overall F1-score for all the classes is very low for Kannada language when compared with all the other languages while we applied deep learning techniques. This is due to the size of the dataset which is lesser for Kannada language. However, Kannada language gives better F1-score than Hindi and Malayalam languages while we use machine learning algorithm.

Table 12. Final-Evaluation Test Data F1-Scores-1 for RE

Runs	Languages	A_1	A_2	A_3	A_neg	A_per	A_quant	A_so	I_1	I_2
Run 1	Hindi	97	93	0	92	NA	90	NA	96	11
	Kannada	NA	18	8	NA	40	19	NA	72	6
	Malayalam	NA	72	73	NA	67	33	86	90	87
	Tamil	67	72	66	84	62	20	NA	93	56
	Telugu	NA	65	73	NA	0	0	NA	83	98
Run 2	Hindi	96	87	0	90	NA	85	NA	96	14
	Kannada	NA	0	0	NA	0	0	NA	62	0
	Malayalam	NA	44	58	NA	54	9	53	87	88
	Tamil	55	64	55	58	44	17	NA	92	51
	Telugu	NA	70	56	NA	0	0	NA	85	98
Run 3	Hindi	67	0	0	20	NA	6	NA	81	0
	Kannada	NA	2	46	NA	0	0	NA	68	3
	Malayalam	NA	0	20	NA	7	9	2	68	40
	Tamil	2	17	28	55	0	0	NA	80	9
	Telugu	NA	15	40	NA	0	0	NA	62	94

5 Conclusions

We have presented a deep learning approach for NER and RE in Indian languages namely "Hindi", "Kannada", "Malayalam", "Tamil" and "Telugu". We

Table 13. Final-Evaluation Test Data F1-Scores-2 for RE

Runs	Languages	I_3	I_4	I_cc	I_closed	I_neg	I_per	I_quant	I_so	other	Overall
Run 1	Hindi	33	89	5	89	0	NA	69	NA	62	59
	Kannada	34	37	28	NA	NA	0	40	NA	16	26.5
	Malayalam	71	65	NA	64	NA	78	80	NA	65	71.62
	Tamil	76	72	89	92	45	73	91	91	45	70.24
	Telugu	70	88	84	90	NA	55	83	83	33	64.64
Run 2	Hindi	32	83	47	88	0	NA	30	NA	75	58.79
	Kannada	18	0	0	NA	NA	0	0	NA	0	6.67
	Malayalam	66	34	NA	30	NA	70	67	NA	46	54.31
	Tamil	71	48	72	85	15	68	89	91	30	59.18
	Telugu	74	87	91	91	NA	56	84	19	36	60.5
Run 3	Hindi	1	28	0	24	0	NA	32	NA	17	19.71
	Kannada	28	20	18	NA	NA	37	34	NA	10	22.17
	Malayalam	17	12	NA	0	NA	16	33	NA	5	17.62
	Tamil	13	6	32	3	4	48	75	29	1	23.65
	Telugu	35	53	19	3	NA	20	72	15	1	30.64

have used neural machine translation model to implement both NER and RE tasks. Our approach is a common approach that identifies and classifies the NEs into any of the generic classes namely name, occupation, location, things, organization, datenum, number and other, and also classifies the relations to any of the relation labels such as information_1, action_1, etc. We have evaluated four deep learning models for NER and two deep learning models for RE by varying the directionality, depth and number of training steps with and without attention mechanism called scaled-luong. To show the effectiveness of deep learning approach, we have also implemented three variations of traditional machine learning models for RE using bag of word features, term frequency / TF-IDF vectorizer and a neural network classifier with SGD optimizer having minimum difference as one and two to extract relations. We have evaluated these models using the data set given by IECSIL@FIRE2018 shared task for NER and RE. We have used the metric accuracy to measure the performance of different variations of our approach. For, NER task, we have obtained the accuracies as 94.41%, 95.23%, 95.97% and 96.02% for Model 1, Model 2, Model 3 and Model 4 respectively on pre-evaluation test data and 95.9%, 95.85% and 95.05% for Run 1, Run 2 and Run 3 respectively on final-evaluation test data. For RE task, we have obtained the accuracies as 56.19%, 60.74%, 60.7%, 75.43% and 79.11% for Model 1, Model 2, Model 3, Model 4 and Model 5 respectively on pre-evaluation test data, and 79.44%, 76.01% and 61.11% for Run 1, Run 2 and Run 3 respectively on final-evaluation test data. The performance may be improved further by incorporating different attention mechanisms, including more hidden layers, and increasing training steps.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. 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)
3. 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)
4. Chikka, V.R., Karlapalem, K.: A hybrid deep learning approach for medical relation extraction. arXiv preprint arXiv:1806.11189 (2018)
5. Chiu, J.P., Nichols, E.: Named entity recognition with bidirectional lstm-cnns. arXiv preprint arXiv:1511.08308 (2015)
6. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
7. Dugas, F., Nichols, E.: Deepnnner: Applying blstm-cnns and extended lexicons to named entity recognition in tweets. In: Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT). pp. 178–187 (2016)
8. Frunza, O., Inkpen, D., Tran, T.: A machine learning approach for identifying disease-treatment relations in short texts. Knowledge and Data Engineering, IEEE Transactions on **23**(6), 801–814 (2011). https://doi.org/10.1109/TKDE.2010.152
9. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)
10. Leng, J., Jiang, P.: A deep learning approach for relationship extraction from interaction context in social manufacturing paradigm. Knowledge-Based Systems **100**, 188–199 (2016)
11. Luong, M., Brevdo, E., Zhao, R.: Neural machine translation (seq2seq) tutorial. https://github.com/tensorflow/nmt (2017)
12. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025 (2015)
13. Lv, X., Guan, Y., Yang, J., Wu, J.: Clinical relation extraction with deep learning. IJHIT **9**(7), 237–248 (2016)
14. Paukkeri, M.S., García-Plaza, A.P., Fresno, V., Unanue, R.M., Honkela, T.: Learning a taxonomy from a set of text documents. Applied Soft Computing **12**(3), 1138–1148 (2012). https://doi.org/10.1016/j.asoc.2011.11.009
15. Peng, B., Wu, J., Yuan, H., Guo, Q., Tao, D.: Aneec: A quasi-automatic system for massive named entity extraction and categorization. The Computer Journal **56**(11), 1328–1346 (2013). https://doi.org/10.1093/comjnl/bxs114
16. Poon, H., Domingos, P.: Unsupervised ontology induction from text. In: Proceedings of the 48th annual meeting of the Association for Computational Linguistics. pp. 296–305. ACL, United States, Uppsala, Sweden (July 11-16 2010)
17. Punuru, J., Chen, J.: Learning non-taxonomical semantic relations from domain texts. Journal of Intelligent Information Systems **38**(1), 191–207 (2012). https://doi.org/10.1007/s10844-011-0149-4
18. Senthil Kumar, B., Thenmozhi, D.: Named entity recognition in dravidian languages – state of the art. International Journal of Applied Engineering Research **10**(34), 27295–72300 (2015)

19. Shen, Y., Yun, H., Lipton, Z.C., Kronrod, Y., Anandkumar, A.: Deep active learning for named entity recognition. arXiv preprint arXiv:1707.05928 (2017)
20. Sinha, B., Garg, M., Chandra, S.: Identification and classification of relations for indian languages using machine learning approaches for developing a domain specific ontology. In: Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on. pp. 415–420. IEEE (2016)
21. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. pp. 3104–3112 (2014)
22. Thenmozhi, D., Aravindan, C.: An automatic and clause based approach to learn relations for ontologies. The Computer Journal **59**(6), 889–907 (2016)
23. Velardi, P., Faralli, S., Navigli, R.: Ontolearn reloaded: A graph-based algorithm for taxonomy induction. Computational Linguistics **39**(3), 665–707 (2013)
24. Wang, W., Barnaghi, P., Bargiela, A.: Probabilistic topic models for learning terminological ontologies. Knowledge and Data Engineering, IEEE Transactions on **22**(7), 1028–1040 (2010). https://doi.org/10.1109/TKDE.2009.122
25. Weichselbraun, A., Wohlgenannt, G., Scharl, A.: Refining non-taxonomic relation labels with external structured data to support ontology learning. Data & Knowledge Engineering **69**(8), 763–778 (2010). https://doi.org/10.1016/j.datak.2010.02.010
26. Zouaq, A., Nkambou, R.: Evaluating the generation of domain ontologies in the knowledge puzzle project. Knowledge and Data Engineering, IEEE Transactions on **21**(11), 1559–1572 (2009). https://doi.org/10.1109/TKDE.2009.25