International Institute Of Information Technology

Hyderabad

PROJECT REPORT

Named Entity Recognition

IIIT HYDERABAD

June 26, 2021

Contributors

- Sandeep Kumar Gandrakota
- Vivek Reddy Bandi

Contents

1 Introduction

Named Entity Recognition (NER) also called entity identification or entity extraction is a Natural Language Processing (NLP) technique that seeks to locate and classify named entities mentioned in unstructured text into pre-defined categories such as a person, organization, location, medical codes, time expressions, quantities, monetary values, percentages, etc.
It has many applications in the field of Natural Language Processing such as text mining, machine translation, question answering, indexing for information retrieval, automatic summarising, etc.

1.1 Objectives

This project aims to classify the tokens in a Telugu data set into different categories like a person, organization, location, and miscellaneous.AS the Telugu language occupied 15th position in the world,2nd position in India and 3rd most spoken language.NER identification for this language is very much required for different applications in the field of Natural Language processing
In this project we develop and evaluate the Long Short-Term Memory-Conditional Random Field(LSTM-CRF) classifier model to perform NER classification for Telugu language data
The output and performance of this Long Short-Term Memory-Conditional Random Field(LSTM-CRF) model is compared with the already existing models

1.2 Challenges in NER for Telugu

Telugu is one of the most widely spoken Indian languages. It has highly inflectional and agglutinating nature and one of the richest and most complex sets of linguistic rules and complex word forms. The task of building a NER model for the Telugu language has some linguistic challenges like the unavailability of annotated corpora, gazetteer lists, capitalization feature, spelling variations, and free word order.
Named Entities in Telugu cannot be identified based on capitalization but it can be done easily for English and most European languages. Inflectional suffixes can be added either to the root or to the stem of Telugu words and common nouns can also be Named Entities in certain cases. The words are also more diverse in nature

2 Related Work

Our approach uses a language-independent LSTM-CRF based classifier that used pre-trained word embeddings, character-level embeddings, and contextual word representations. Then a CRF is used to perform classification. Many of the earlier proposed models on Telugu Language either used handcrafted rules and gazetteers or have obtained low accuracies. Along with Telugu, a considerable amount of work had been done in other Indian languages especially Bengali, Hindi, and Tamil.

A CRF-based NER system for Telugu is developed by Srikanth and Murthy (2008). They built a two-stage classifier. Firstly, they used a CRF to identify nouns. Then, they developed a rule-based model to identify the NEs among the nouns. Then a CRF-based NER model was developed using several extracted features and gazetteers. This model was developed using a part of the LERC-UoH Telugu corpus which includes books, articles, and news corpus of AndraPrabha and Eenadu. They have obtained F-measures ranging from 80% to 97% in various experiments. But the higher scores were obtained only upon using gazetteer lists and their model is capable of identifying NEs with one-word length.

A Hybrid approach that uses the Maximum Entropy model to identify the nouns and then identifies Named Entities from them using gazetteer lists and rules was presented by Khanum and Udayasri (2017). They have manually created a corpus from various resources like newspapers and blogs. Then the MaxEnt model was used to identify nouns by tagging the data into noun, verb, and others categories. Then Rule-based approach using several gazetteer lists, language-dependent features, and rules were used to identify the named entities from the nouns. Hence this approach assumes named entities as nouns and cannot identify several named entities that are not nouns.

Ekbal, A. and Bandyopadhyay, S. (2008) developed an SVM-based model for Named Entity Recognition using various features and gazetteer lists features as an input feature set. To properly denote the boundaries of the NEs to apply SVM, sixteen NE and one non-NE tag have been defined such as individual NE tag and beginning, internal, and ending NE tag. Then these 16 tags are replaced with 4 major tags using simple heuristics. A partially NE tagged Bengali news corpus developed from the archive of a widely read Bengali newspaper is used to develop SVM based NER system. The corpus contains around 34-million-word forms in ISCII and UTF-8 format and training set with 150K words have been developed by manually annotating the data with the sixteen NE tags. This model gave an F-measure of 91.8% on Bengali language and it can be applied to other Indian languages as well. But the gazetteer lists were used and SVM doesn't capture any contextual information from data.

A language-independent Hidden Markov Model (HMM) based classifier for NER in various Indian languages like Bengali, English, Hindi, Marathi, Punjabi, Tamil, and

Telugu was proposed by Vivekananda Gayen Kamal Sarkar at ICON 2013. This approach calculates the best sequence of tags to be assigned for the given input sequence. Since the calculation of probabilities is difficult, they used a dynamic programming-based model. The observation probability matrix and tag transition probability matrix are created in the training phase and the Viterbi algorithm is used for decoding. They used the corpus provided at the NLP tools contest i.e., ICON 2013 datasets. This approach gave a good F-measure of 85.9% for Bengali but performed purely for Telugu with an F-measure of 40.03%.

A Conditional Random Field Approach for Telugu NER was proposed by Shishtla P and Gali K (2008). This model predicts the NER by calculating the conditional probabilities of the values at the output nodes for the given values at the input nodes based on the Hammersley Clifford theorem. They used the development data released as a part of NER for South and southeast Asian Languages (NERSSEAL) Competition. The corpus in total consisted of 64026 tokens out of which 10894 were Named Entities. No gazetteer lists or rule-based methods were used in this model. But this approach gave very little accuracy and an F-measure of 44.89%

Named Entity Recognition in Kannada using Multinomial Naive Bayes (MNB) classifier was proposed by S Amarappa Dr. S V Sathyanarayana (2015). They used a simple classifier based on Bayes' theorem with strong and naive independence assumptions between every pair of features. The probability of a token belonging to each label is found using the Naive bias Algorithm. They have manually created a corpus with 100K Kannada words which includes Part of the EMILIA corpus, web articles, and Kannada books. This model takes less time compared to SVM and MaxEnt approaches and gives similar accuracy. They have obtained a Precision, Recall, and F-measure of 83%, 79%, and 81% respectively.

A bidirectional LSTM (Bi-LSTM) and a Convolution Neural Network hybrid model that automatically detects word and character-level features were proposed by Chiu, J. P., and Nichols, E. (2015). The CNN extracts character-level features from each word as feature vectors. The word-level feature vectors are obtained from multiple lookup tables. Both feature vectors are concatenated and fed into Bi-LSTM and tag scores are obtained for each tag. Lexicons and some additional features were used to improve performance. The 50- dimensional word embeddings released by Collobert et al. (2011b) trained on Wikipedia and the Reuters RCV-1 corpus were used. The model obtained F-measures of 91.62% on CoNLL-2003 and 86.28% on CoNLL-2012 datasets.

3 Corpus Description

We have used the corpus provided by Aniketh and monica (2019) which was accumulated by crawling Telugu newspaper websites and then manually tagged by with NEs belonging to four classes, namely, person (PER), location (LOC), organization (ORG) and other miscellaneous NEs (MISC) using the standard IOB scheme. It was described as Golden Telugu Data and it contains train and test sets with various number of entities. It was used to train model in Experiment 1.

Additionally, we have obtained data from web sources which was termed as NER Telugu data. Then it was added to Golden Telugu Data and model to train model with more data in Experiment 2. The details of number of entities in individual datasets and combined dataset is shown in tables below:

3.1 Statistics of Golden Telugu Data:

The details of NEs in Golden Telugu Data set are formulated in below table:

No.Training sets	Organization	Location	Person	Miscellaneous	Total
set-1	1180	1630	1977	2298	7085
set-2	1118	1624	1905	2250	6897
set-3	1165	1587	1959	2226	6937
set-4	1182	1608	1902	2275	6967
set-5	1161	1660	1961	2215	6997
set-6	1222	1661	1895	2234	7012
set-7	1134	1629	1930	2288	6981
set-8	1171	1625	1869	2300	6965
set-9	1168	1637	1892	2256	6953
TOTAL	10501	14661	17290	18342	62794

3.2 Statistics of NER Telugu Data

The details of NEs in NER Telugu Data set are formulated in below table:

No.Training sets	Organization	Location	Person	Miscellaneous	Total
set-1	1198	1592	1894	2255	6934

3.3 Statistics of Combined Data

The details of Combined Data set are formulated in below table:

Category	0rganization	Location	Person	Miscellaneous	Total
TOTAL	11699	16253	19184	20597	69728

4 Model and Implementation

4.1 LSTM-CRF classifier

We introduce LSTMs, CRFs and other relevant concepts before proceeding to explain the architecture of the classifier.

4.1.1 Long Short-Term Memory (LSTM)

RNNs are capable of learning long term dependencies on sequential data points. But they are prone to vanishing gradient problem leading to incapability of accurately learning long term dependencies which is essential for NLP tasks. LSTMs are a class of RNNs which overcomes this issue of by employing a 3-gated structure and selectively storing and forgetting data according to the context.

It takes sequential data of the form (x0, x1, x2...xn) as input and gives a sequential output of the form (y0, y1, y2...yn). It has a memory cell and 3 gates: Forget gate, Input gate and Output gate.

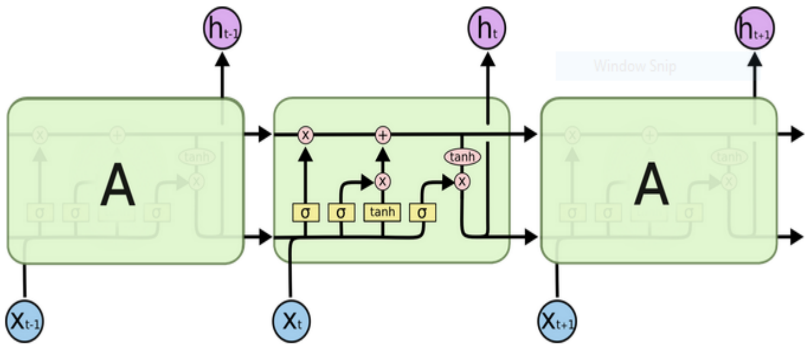

Figure 1: *Internal network in LSTM*

The forget gate determines how much of the previous state's information is to be retained and selectively forget the information when the context changes. It takes previous state output ht1 and present input Xt, and outputs a number between 0 and 1 for each number in the cell state Ct1.

$$f_t = \sigma(W_f \times [h_t - 1, x_t] + b_t$$

The input gate controls how much the state changes due to the new input. It selectively reads the input and decides about what new information to be stored in the new cell state. The sigmoid layer decides which values to update (it) and tanh layer creates a vector of new candidate values, C t. Then these are combined to give an update to the cell state Ct.

$$i_t = \sigma(W_i \times [h_t - 1, x_t] + b_i$$

$$\widetilde{C}_t = \tanh(W_c \times [h_t - 1, x_t] + b_c$$

$$C_t = f_t \times C_t - 1 + i_t \times \widetilde{C}_t$$

The LSTM finally gives an output which is determined by the output gate based on the internal state. It selectively writes a filtered version of the cell state to the output. The sigmoid layer decides which parts to output from the previous state Ot and then tanh layer bounds the cell state Ct values between –1 to +1. Both are multiplied and given as output ht

$$o_t = \sigma(W_o \times [h_t - 1, x_t] + b_o$$

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

4.1.2 Bi-directional LSTM

A Bidirectional LSTM, or biLSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. This structure allows the networks to have both backward and forward information about the sequence at every time step. Bi-LSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm. Hence in order to get both the left and right contexts of data, we use a Bi-LSTM and then concatenate the internal states of the forward and backward LSTMs.

Character-Level Word Embeddings:

The structure of a word is useful in determining if it is a named entity. This structure can be captured through character-level word embeddings. Initially each character is assigned an n-dimensional vector. Each token is broken up into its individual characters which are then mapped to their corresponding vectors. Thus, a token is converted to a sequence of vectors which is then fed to a Bi-LSTM and final character-level embedding of the token is obtained

Generating Contextual Representations of Words:

In the case of NER, the LSTM stores information about the tokens which occurred before the current token, thereby storing a contextual representation of the current token. By using Bi-LSTM, we can store the contextual representation of a token from left and right contexts of the data. After processing the given token, the token's final contextual representation is obtained by concatenating both the contexts.

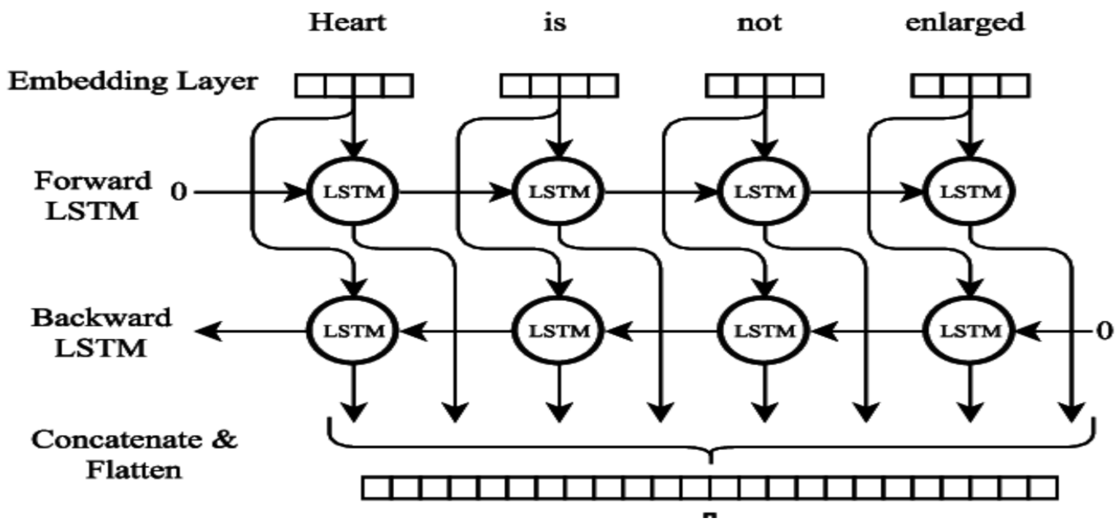

Figure 2: *Working of Bi-directional LSTM*

4.1.3 Linear Chain Conditional Random Fields(CRFs)

These CRFs are a class of statistical modeling techniques that predicts sequences of labels for sequences of input samples. The tagging is done for the entire sequence simultaneously and each tagging decision is dependent on the others. Hence it considers the linear dependencies in the data. For the NER task, the tagging decision must be made after taking into account the tagging decisions for all the other tokens in the sequence as they are highly dependent on each other. So using CRFs can improve the performance of the NER model by tagging Named Entities accurately.

Each token ti in a sequence of the form (t0, t1, t2...tn) has a corresponding m-dimensional vector si where m is the number of possible tags. Another m $\times$ m transition matrix A is used to capture the linear dependencies between the tagging decisions. Aij of the matrix is indicative of the probability of the ith tag being followed by the jth tag. s and e are two m-dimensional vectors whose values represent the confidence of a tagging sequence starting and ending with a given tag respectively. Each tagging sequence of the form (y0, y1, y2...yn) assigned to the token sequence is scored as

$$Score(y_0, y_1, y_2...y_n) = s[y_0] + \sum_{i=0}^{n} s_i[y_i] + \sum_{i=0}^{n-1} A[y_i, y_i + 1] + e[y_n]$$

The tagging sequence which has the maximum Score is output by the CRF-based classifier. While training, we minimize the negative log of the probability of the correct tagging sequence Y. The loss function is

$$Loss = -log(P(\widetilde{Y}))$$

$$P(\widetilde{Y}) = e^{Score(\widetilde{Y})} / \sum_{i=o}^{n} e^{Score(y_0, y_1, y_2 \ldots y_n)}$$

Then during backpropagation, the various weights and embeddings of the model are tuned to minimize the Loss.

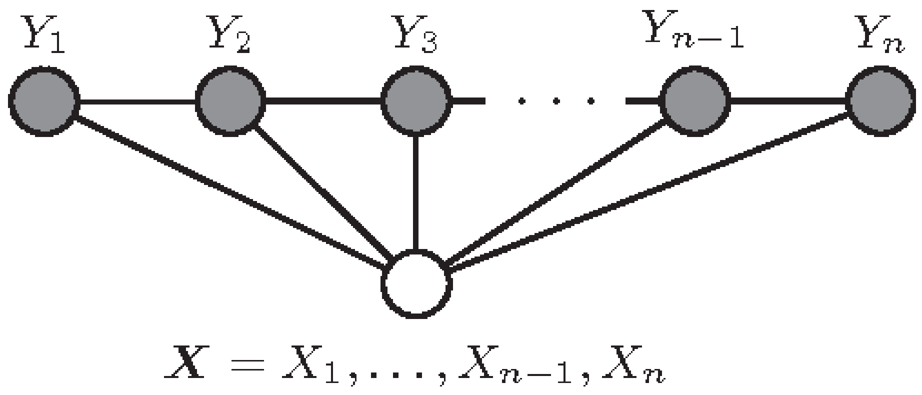

Figure 3: *Structure of linear chain CRF*

4.2 Neural Network Architexture and Implementation

Our classifier uses the global word embeddings and the character-level embeddings which are concatenated for each token to represent them as vectors. Using the training data now represented in the form of sequences of vectors, 600-dimensional contextual word embeddings are generated for each token. The contextual word embeddings are then used to compute the scores for a word using a 600 × m weight matrix W and an m dimensional bias vector b where m is the number of classes. The score for each word is an m dimensional vector given by:

$$s = W \times h + b$$

where h is the contextual embedding of that word. After getting the scores for an entire sequence, we use a linear chain CRF to make the tagging decisions for the whole sequence.

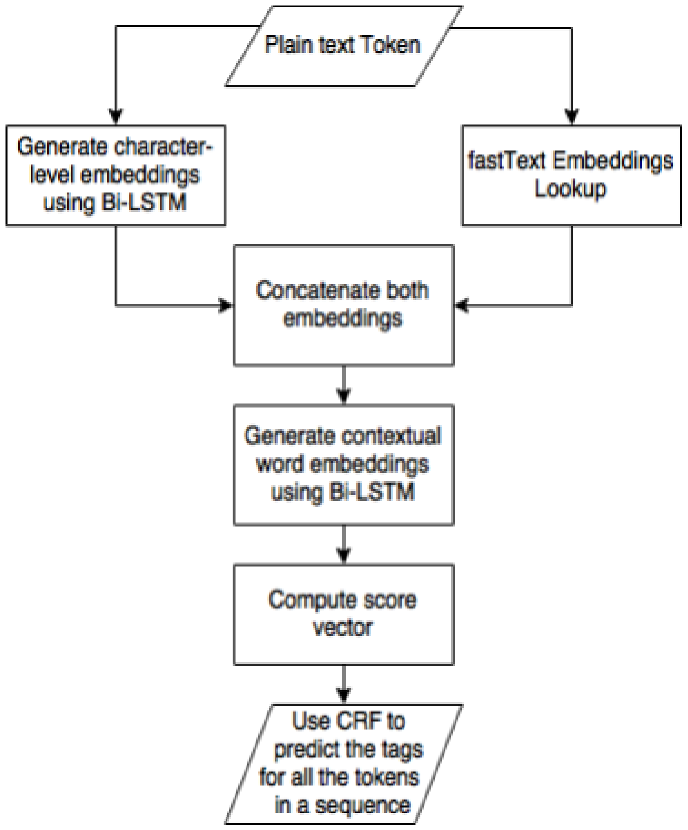
Plain text Token
Generate character-level embeddings using Bi-LSTM
fastText Embeddings Lookup
Concatenate both embeddings
Generate contextual word embeddings using Bi-LSTM
Compute score vector
Use CRF to predict the tags for all the tokens in a sequence

Figure 4: *Architecture of LSTM-CRF classifier*

5 Experiments and Results

Two experiments were performed one using Golden Dataset and the other by adding additional data. Then the performance of the model is evaluated and the results of these experiment are compared with the existing model results.

5.1 Experiment-1

In this Experiment first, we divided the Golden Telugu Data(Details of this data are mentioned in Corpus Description) into multiple trains and test sets. The model was trained with Train set for 20 Epochs. The results of the model were depicted in the tables below. Set to evaluate the performance of the model and compared the results with the existing model results.

5.1.1 Results

The vocabulary of 15747 tokens is developed and the model was trained on data. We have obtained an accuracy of 99.52% and an F-measure of 96.69% after training for 20 epochs.

3.Evaluation Table:

The model has processed 9812 tokens with 1293 phrases and found 1238 phrases and 1103 correct words.

Category	Precision	Recall	F1 score	Correct
LOCATION	88.47%	88.95%	88.71	373
Organisation	74.48%;	67.50%	70.82	145
Person	88.40%	79.66%	83.80	319
Miscellaneous	95.51%	93.87%	94.68	401
Overall	96.48%	89.10%	85.31%	1103

5.2 Experiment-2

We have evaluated the performance of the model by using the combined data set after adding the additional dataset. The model was once again trained and evaluated on this combined data and results are analyzed.

5.2.1 Results

The model was trained on combined data. We have obtained an accuracy of 99.62% and an F-measure of 98.03% after training for 20 epochs.

3.Evaluation Table:

Processed 9812 tokens with 1293 phrases and found 1223 phrases and 1080 correct words.

Category	Precision	Recall	FB1	Correct
LOCATION	90.17%	86.52%	88.31	356
Organisation	94.46%;	91.91%	93.17	397
Person	88.40%	77.12%	80.65	323
Miscellaneous	94.46%	91.91%	93.17	397
Overall	96.27%	88.31%	83.53%	1080

Then we have also created a test set of 50 sentences and compared the model performance.

6 Future Work and Conclusion

In this Project, we described the LSTM-CRF model-based approach for NER classification in Telugu. Named Entity Recognition(NER) have many applications in the field of Natural Language processing.LSTM-CRF model produces accurate results and has high accuracy and F-Measure.
We have used Golden Data corpus in Experiment 1 and added NER Telugu corpus in Experiment 2. The performance of the model was evaluated over two experiments and the results are evaluated.The accuracy and F-measure achieved by the LSTM-CRF model are 99.52% and 97.61% respectively.

The LSTM-CRF classifier clearly outperforms the other classifiers based on all the metrics. In fact, its performance is greater by approximately 7% based on overall F-measure. This is because of three main reasons. Firstly, the use of CRFs makes tagging more accurate because the classifier is able to discern the dependencies which exist between the individual tags. These dependencies can not be captured by an SVM and other classifiers thereby lowering its performance.Secondly, character-level embeddings allow the classifier to learn the general form of an NEs. Hence, our classifier is able to handle unknown NEs because it is able to identify them based on their structure. Finally, Bi-LSTMs are great at learning long term dependencies and further augment the classifier's ability to learn dependencies between tokens.

This project can be further extended to all Indian Languages. There is a large scope to increase the effectiveness of the model since NER for Telugu is a very tough task. We can add more data to the corpus and train the model to increase the accuracy and F-measure of the model and improve the performance.

7 References

1.Named Entity Recognition for Telugu using LSTM-CRF by Aniketh Janardhan Reddy, Monica Adusumilli, Sai Kiranmai Gorla, Lalita Bhanu Murthy Neti and Aruna Malapati
2.Srikanth, P. and Murthy, K. N. (2008). Named entity recognition for telugu. In IJCNLP, pages 41–50.
3.KANNADA NAMED ENTITY RECOGNITION AND CLASSIFICATION (NERC) BASED ON MULTINOMIAL NAÏVE BAYES (MNB) CLASSIFIER by S Amarappa 1 Dr. S V Sathyanarayana
4.Experiments in Telugu NER: A Conditional Random Field Approach by Praneeth M Shishtla, Karthik Gali, Prasad Pingali and Vasudeva Varma
5.Vijayakrishna, R. and Devi, S. L. (2008). Domain focused named entity recognizer for tamil using conditional random fields. In IJCNLP, pages 59–66.
6.Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. CoRR, abs/1603.01360.
7.Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L., and Luís, T. (2015). Finding function in form: Compositional character models foropen vocabulary word representation.
8.In EMNLP.shishtla, P. M., Gali, K., Pingali, P., and Varma, V. (2008). Experiments in telugu ner: A conditional random field approach. In Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages.

References

Appendices