Journal of Intelligent & Fuzzy Systems xx (20xx) x–xx
DOI:10.3233/JIFS-179349
IOS Press

A step towards information extraction: Named entity recognition in Bangla using deep learning

Redwanul Karim, M.A. Muhiminul Islam, Sazid Rahman Simanto, Saif Ahmed Chowdhury, Kalyan Roy, Adnan Al Neon, Md. Sajid Hasan, Adnan Firoze and Rashedur M. Rahman*
Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh

Abstract. Information Extraction allows machines to decipher natural language through using two tasks: Named Entity Recognition and Relation Extraction. In order to build such a system for Bangla Language, in this work a Named Entity Recognition (NER) System is proposed, which requires a minimum information to deliver a decent performance having less dependency on handcrafted features. The proposed model is based on Deep Learning, which is accomplished through the use of a Densely Connected Network (DCN) in collaboration with a Bidirectional-LSTM (BiLSTM) and word embedding, i.e., DCN-BiLSTM. Such a system, specific to the Bangla language, has never been done before. Furthermore, a unique dataset was made since no Named Entity Recognition dataset exists for Bangla language till date. In the dataset, over 71 thousand Bangla sentences have been collected, annotated, and classified into four different groups using IOB tagging scheme. Those groups are person, location, organization, and object. Due to Bangla's morphological structure, character level feature extraction is also applied so that we can access more features to determine relational structure between different words. This is initially done with the use of a Convolutional Neural Network but is later outperformed by our second approach which is through the use of a Densely Connected Network (DCN). As for the training portion, it has been done for two variations of word embedding which are word2vec and glove, the outcome being the largest vocabulary size known to both models. A detailed discussion in regard to the methodology of the NER system is explained in a comprehensive manner followed by an examination of the various evaluation scores achieved. The proposed model in this work resulted in having a F1 score of 63.37, which is evaluated at Named Entity Level.

Keywords: Named entity recognition, information extraction, word embedding, sequence labelling, Bi-LSTM, densely connected network, Bangla, annotation, dataset, NLP, neural network, character level feature extraction, CNN

1. Introduction

Every day more than 2 million blog posts are written on the web. Along with blog posts, newspapers are publishing articles, books are getting printed, documents are being created, and so forth. In recent times, Bangladesh has moved towards digitized distribution and sharing of such sources of text. From the huge amount of text corpus, it is extremely tedious to extract the information through just reading it. To simplify such a tiresome task, deep learning is used to extract the information in a structured and logical manner in our work. However, a problem arises in which natural language is easily interpretable by humans but very difficult for machines seeing as how the data is in an unstructured format. Thus, a large amount of information may remain obscure if we are not able to retrieve structured information from the unorganized data. This is the point which motivates us to do our intended research. We are particularly interested in doing our research on retrieving structured information from an unstructured Bangla corpus as

*Corresponding author. Rashedur M. Rahman, Department of Electrical and Computer Engineering, North South University, Dhaka-1229, Bangladesh. E-mail: rashedur.rahman@northsouth.edu.

Bangla is the seventh most common language in terms of the number of native speakers across the world [32].

One of the most commonly researched topics of Natural Language Processing (NLP) is Information Extraction (IE). Ultimately, information extraction is the component which enables machines to translate, comprehend, and control natural language. The task of Information Extraction for an unstructured dataset is a combination of two tasks. The first task being Information Extraction (IE) which is done through Named Entity Recognition (NER) and the second task being to find the relationship between the recognized entities, which is known as Relation Extraction (RE).

Due to the fact that the Bangla language is a free word order language and does not have capitalization of any sort, this leads to the conclusion that the Named Entity Recognition (NER) for the language is not a trivial task even in a small domain. Another characteristic to consider about Bangla language is that it is both diverse in statistical and linguistic features. Thus, all of these characteristics about the language play a crucial role in information extraction.

Examples of implementations relating to Named Entity Recognition of Bangla language are as follows. A classifier combination of CRF, SVM, and two Maximum Entropy to classify the Named Entities in Bangla Language was introduced by Ekbal et al. [20] through a majority voting approach. In addition, a Margin Infused Algorithm was implemented in Bangla NER by Banerjee et al. in [21].

All the works mentioned above used handcrafted features which entails that the models need more information and intervention of actual human beings to deliver a good result. However, we can overcome this problem by using Deep Learning, which is one of the main objectives of this research. Deep Learning based models learn the hidden features from the data by itself, which results in less dependency on handcrafted features. This is the main motivation of research. The key contributions of our research include the following:

- There were no publicly available NER datasets for Bangla, we had to create our own with a size of around 71 thousand Bangla sentence through the use of IOB tagging scheme.
- We design DCN that can outperform CNN based feature extractors in terms of extracting morphological features from Bangla words. We use a much more simplified version of what Lee et al. proposed in [25].
- Within our work, we propose an NER System achieving an F1 score of 63.37 not requiring any information other than IOB tagged sentences, based on a Densely Connected Network (DCN) for character level feature extraction, word embedding from a pre-trained model (i.e. word2vec), and a Bi-LSTM for sequence labelling.

In Section 2, we thoroughly discuss the related works that predated ours. A detailed work flow of this whole research is given in the Section 3, which also includes Data Pre-processing, NER Dataset Preparation and details of Pre-trained word embedding. What follows after is the NER System, discussed in Section 4. Next we present the Training and Inference described in Section 5, in which the hyper parameter tuning, evaluation of the system are discussed. Accordingly, we discuss the results with analysis in Section 6. Finally, we briefly review the entirety of our research and how our thoughts have progressively changed over time in Section-7.

2. Literature review

Several supervised learning based models were proposed in the early stage of NER problem. Considerably the earliest one, proposed by Bikel et al. in [18], that used a Hidden Markov Model (HMM) to detect named entities. Maximum Entropy based approaches were used in [14, 19], proposed by Curran et al. and Borthwick respectively on NER. Conditional Random Field (CRF) was used in [15], which is based on feature induction, proposed by McCallum and Li for Named Entity Recognition. An F1 score of 90.80 on CoNLL-2003 dataset was achieved by Ratinov and Roth in [11] through the use of non-local feature based model, in which gazetteer and brown cluster were used as features. This model's performance was later surpassed by Lin and Wu [12], but without the use of gazetteer. They achieved this by using phrase clusters as features in a discriminative classifier, which were extracted by performing k-means clustering over a tens of millions of phrases. Another similar performance using public data by training phrase vectors in their lexicon infused skip gram model was proposed by Passos et al. in [7]. This was considered to be a revolutionary system, due to the fact that there was no use of external knowledge.

Durrett and Klein created a single CRF model in [8] through the combination of Entity Linking, Named Entity Recognition, and Co-reference Resolution. The results from the model brought forth new results on the OntoNotes dataset. Feature reduction by use of large scale unlabeled data to encounter sparse features was done by Suzuki et al. in [10], which managed to get an F1 score of 91.02 on CoNLL-2003. The distinguished results from CoNLL-2003 was attained through training a joint model over NER and an entity linking task which was proposed by Luo et al. in [5]. The improved results were due to the interconnections between the two models of Luo et al. [5] and Durrett and Klein [8].

All of the approaches mentioned previously rely on heavily handcrafted feature engineering to perform well. On the other hand, Neural Network (NN) based models need almost no handcrafted features, which is what makes NN models much more robust in nature. A multi-layered feed forward neural network was used for NER [16], in which the authors particularly focused on choosing appropriate data representation and used only POS tag and gazetteer tag rather than using word embedding. Collobert et al. in [9] proposed an amalgamated neural network architecture with a view to reduce the task-specific engineered features. This approach helped them to achieve good results for most of the tasks – POS tagging, NER, Chunking, semantic role labelling through the use of intermediate representations discovered on large unlabeled datasets. Self-organizing maps (SOM) for sequences - that deliberate on word morphology and principle component analysis was used to obtain word embedding and context vectors respectively along with a single directional LSTM for Named Entity Recognition, was introduced by Hammerton in [13]. The character level CNN that is used in our model to obtain character level features, was first used in CharWNN network depicted by Santos and Zadrozny in [6] for Spanish and Portuguese NER. Character level CNN's were also used in [2], proposed by Labeau et al. for German POS-tagging. Instead of using CNN to auto extract the character level features, Huang et al. in [3], proposed multiple models using different combinations of LSTM, BiLSTM, and CRF for POS tagging, NER and chunking task, in which they produced more or less state of the art results using BiLSTM-CRF model. Yang et al. in [1] proposed a neural re-ranking system for NER, which uses LSTM-CNN structure to exploit deep representation of sentences for re-ranking.

We are using almost the same configuration in our model as used in [4] by Chiu and Nichols. Differences between the models proposed by Chiu and Nichols [4] and ours is that we experimented with two feature extractors at character level. The first feature extractor is CNN based and the second one is a simple DCN. Rather than sticking with a Softmax layer which is only for predicting the labels, we experimented with a CRF layer. We took inspiration from the work depicted by Lee et al. in [25] in order to extract features at character level using a DCN, however ours is a much simpler version than what the authors proposed in their original work. Interestingly, this simpler version outperformed the CNN based approach, which we will show in the result analysis section. In [4] authors used casing information along with lexicons, whereas we are only using word embedding and character level features as inputs of our model, as the Bangla language does not have any feature like – casing (upper case and lower case). Though we have mentioned about few works [20, 21] which have been done for NER in Bangla Language, we are not in a position to compare their results with ours due to the fact that they are mostly based on traditional machine learning algorithms, which requires more hand-crafted features such as – Gazetteers, Clue words, POS tags that are not present in our dataset.

3. Workflow

We divided our whole workflow into five major segments – Data Pre-processing, NER Dataset Preparation, Pre-trained Word Embedding, Named Entity Recognition (NER) System and NER System Evaluation. A conceptual diagram of the whole workflow is given in Fig. 1.

At the initial stage of this work we started with collecting text documents from various Bangla online news sources and Wikipedia (see Appendix A). All these collected documents are then pre-processed to filter out garbage texts and turned into a corpus of tokenized sentences. A sub-set of the corpus we have prepared in the previous step is selected to prepare the NER dataset and the word embeddings are pre-trained on the whole corpus. After this step, we build an NER system with character level feature extractor (e.g. CNN, DCN) and BiLSTM. The predictions done by the NER System is then evaluated at named entity level using the SemEval'13 [39] standard.

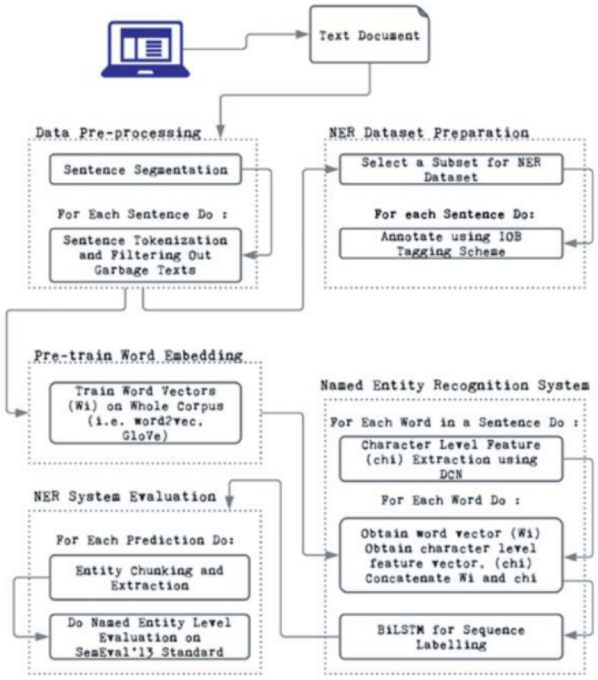

Fig. 1. Workflow diagram of the whole research.

3.1. Data pre-processing

As we are using Deep Neural Network for named entity recognition, we need a large volume of data to train our model. Thus, online documents are the best way to resolve this issue. Since online resources are full of different types of official and unofficial documents, we have used articles from Bangla Wikipedia and some Bangla newspapers (see Appendix A).

Since we need to scrape an enormous number of articles, we need a framework which requires less memory and CPU usage. To deal with this issue, we scrape the articles from online resources using open source python framework Scrapy [33], which is popular for web scraping and crawling.

To prepare our dataset, we scraped a total of 8,385,616 Bangla sentences from Bangla Wikipedia and three renowned Bangla newspapers; Ittefaq, Bangladesh Pratidin and Kaler Kantho.

Articles from different sections were scraped (e.g. politics, opinions, editorials, sports, health, and entertainment) from the newspapers and Wikipedia, ranging over a period of several years, all in an attempt to reduce any sort of bias due to the fact that some of the content may be the same.

After scraping the articles, we finalized our sentences by filtering out the garbage texts and texts written in other languages. In total, 8,154,503 sentences were left for our research. In order to identify the Person, Location, Organization, and Object named entity tags, we tokenized our scraped sentences into words using another open source python library named spacy [34] which is created for natural language processing.

Table 1
Dataset details

#	Frequency
Sentences	71,284
Tokens	983,663
Unique Tokens	96,154
Tokenized Sentence Length	[5–30]
Tagging Scheme	IOB

3.2. NER dataset preparation

Unlike other languages, there are very few available resources for tasks in Bangla e.g., complete text corpuses for OCR, data visualizations etc. Languages such as English, Spanish, and Russian have their very own standard datasets, from which benchmarking can be done using different tasks. To the best of our knowledge, we know of no publicly available Bangla datasets which can be used for NER task. Due to this issue, we created a dataset of our own which contains 71,284 tokenized sentences with proper annotation using a slightly modified version of the Annotation tool written by Adhikary [35]. Details of the dataset that we generated are given in Table 1.

We use the IOB tagging format which was first implemented by Ramshaw and Marcus in [22]. In an IOB tagging scheme, there is a set of chunk tag such as: {I, O, B}, from which tokens marked with 'I' are inside of the chunk, tokens marked with 'O' are outside, and tokens marked with 'B' are considered to be at the beginning of any base chunk. This dataset is annotated using four base tags, which are PER (Person Entity), LOC (Location Entity), ORG (Organization Entity), and OBJ (Object Entity).

By PER, we have considered all the nouns that are related to person and all the honorific and non-honorific forms of pronouns except the neutral form (e.g. ইহা (It)). Designations are also considered as PER in our dataset. For example, আমি (I), আমরা (We), তুমি (You), তোমরা (You), non-honorific - তুই (You), honorific - তিনি (He/She), তারা (They), person name সীমান্ত (Simanto), designation - ডাক্তার (Doctor), শিক্ষক (Teacher) etc. By LOC, we have grouped the words that deal with any place such as ঢাকা (Dhaka), বাংলাদেশ (Bangladesh), রাস্তা (Road), নদীতীর (Riverbank) etc. Any word that denotes an orga-

```
'sentence' : [ 'আমরা' , 'সর্ব' , 'সাউথ' , 'বিশ্ববিদ্যালয়' , 'এর' , 'ছাত্র' ],
'iob_tags' : [ 'B-PER' , 'B-ORG' , 'I-ORG' , 'I-ORG' , 'O' , 'B-PER']

'sentence' : [ 'আরব' , 'সাম্রাজ্যের' , 'খলিফা' , 'ওয়াপিদের' , 'পূর্বাঞ্চলীয়' , 'গভর্নর' , 'হাজ্জাজ' , 'বিন' , 'ইউসুফ' ]
'iob_tags' : [ 'B-LOC' , 'O' , 'B-PER' , 'B-PER' , 'B-LOC' , 'B-PER' , 'B-PER' , 'I-PER' , 'I-PER']
```

Fig. 2. Sample of NER dataset.

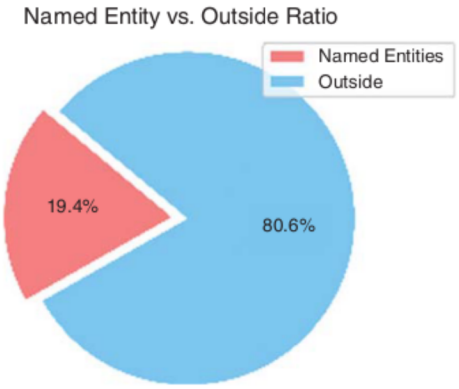

Fig. 3. Named entity vs outside ratio.

Table 2
Frequency per category

Category	Frequency
Person	101,166
Location	49,516
Organization	27,001
Object	11,669
Outside	794,311

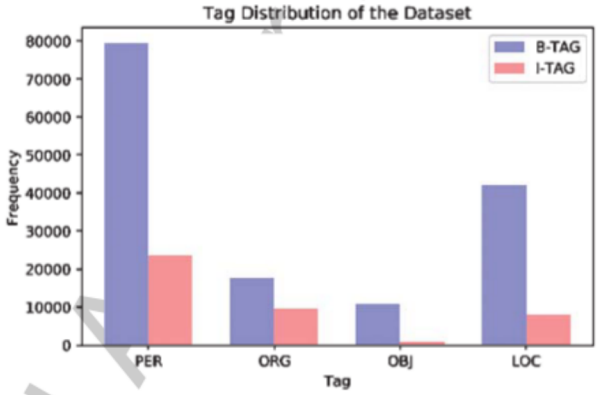

Fig. 4. Tag distribution of the dataset.

nized class or group of people with a purpose is annotated as ORG. For example, ওয়ার্ল্ড সায়েন্টিফিক (World Scientific), সরকার (Government), etc. As far as OBJ is concerned, any word that describes a lifeless but tangible thing, is taken into consideration. In other words, we are considering neuter gender such as, টেবিল (Table), তারা (Star), পাতা (Leaf), etc. For the task of annotation of any entity with more than just one token, we use base tags with appropriate IOB chunk tag. For example if a person's name contains three tokens, such as 'মোহাম্মাদ', 'রাশেদুর', 'রহমান' ('Mohammad', 'Rashedur', 'Rahman'), then this entity is tagged as ('B-PER', 'I-PER', 'I-PER'), which is then considered in this dataset as a Person mention, thus tagged with B-PER. Figure 2 illustrates a sample format of our tagged NER dataset.

In general, the frequency of Named Entities within a corpus is very low in comparison to the outside tokens. This statement holds for our dataset as well. A pie chart depicting the relation to the ratio of Named Entity and Outside mentions is given in Fig. 3.

As previously mentioned, we are using four different categories (i.e. PER, LOC, ORG, OBJ) to annotate the tokens in our dataset. Numbers of tokens of each of the categories can be seen in Table 2.

Analysis of tag distribution of our dataset shows that person mention has the highest frequency, location mention has the second highest frequency, organization mention has third highest frequency, and object mention has the least frequency. This can be seen in Fig. 4. As the annotation is done by more than one person, there might possibly be some differences in tagging due to the dissimilarity in human perception. Reason behind this could be that the same words can be used with different meaning and context. If we have a sentence like, সততার সাথে পুলিশ দায়িত্ব পালন করেছে। (The police have honestly served their duty), then in Bangla, the word পুলিশ (police) could be annotated both as a person and an organization. This could happen because of the fact that, in the above sentence it is not clear if the word পুলিশ (police) is used as a singular form, which would help the person by whom the annotation is done, to label the word as Person mention, or as a plural form, which would help to detect the word as Organization.

3.3. Pre-trained word embedding

Word embedding is a natural language modeling technique to represent words from the vocabulary in

a predefined vector space as vectors of real numbers. Based on syntactic and semantic relationship of one word with other words in a sentence, word embedding creates a unique vector representation for that word. Although every word has its unique vector representation, words that are similar to each other end up having values closer to each other.

As we have a limited annotated dataset, using word embedding in our model instead of a normal encoding system such as one hot encoding, we can improve the accuracy of our model dramatically. If we have a sentence in our training dataset like - "এই বছর আমের ভালো (This year farmers are benefited because of the mass production of mango) where "mango" is annotated as object, and we have a sentence in our test dataset like - ফলনের কারণে কৃষকরা লাভবান হয়েছে।" (This year farmers are benefited because of the mass production of jackfruit), then word embedding will help us to detect "jackfruit" as object even if we do not have the word jackfruit in our whole training dataset, because the vector representation of the words "mango" and "jackfruit" are closer to each other. Due to this fact we have trained our model using two different word embedding models Word2vec and GloVe and analyzed their performances; the detailed results of which are discussed in the result analysis section.

Mikolov et al. (2013) [23] proposed a predictive word embedding model Word2vec, which has revolutionized Natural Language Processing by providing much better vector representations for words than past approaches. Then a count based word embedding model GloVe was proposed by Pennington et al. (2014) [26], which is also based on almost the same approach as word2vec. Hence, any of these two models can perform equally as impressively. Due to that, we used both Word2vec and GloVe, and experimented with 50, 100, 150 and 200 dimensions of word embedding.

We trained our word2vec model using open-source Python library Gensim [36] word2vec module which specializes in vector space and topic modeling. We trained our word2vec model for 50, 100, 150 and 200 dimensions, each of which has a vocabulary size of 446,087, trained over 112 million words. Previously, Alam et al. trained word2vec as part of their research [31], which had 84.25 million words and with a vocabulary size of 436,126. To train our Glove model, we used the python implementation which was implemented by Kula et al. [37], and is also trained for the same dimensions, each of which has a vocabulary size of 1,501,767. As of now, to the extent of our knowledge, our word2vec model is the largest word2vec word embedding for Bangla language and our glove model is first glove word embedding for Bangla.

Table 3
Word embedding details

#	Frequency
Dimensions	[50, 100, 150, 200]
Sentences	8,154,503
Unique Word	1,501,767
Vocabulary Size (Word2Vec)	446,087
Vocabulary Size (GloVe)	1,501,767

4. Named entity recognition system

To realize our primary goal, we use a very similar model which was previously used by Chiu et al. in [4]. The core feature of this model is that it uses character level feature extraction. Another feature is that BiLSTM is used for the labelling of the sequence of words. The diagram of the entire model can be seen in Fig. 5.

4.1. Sequence labeling with Bi-LSTM

Named Entity Recognition is a sequence labelling task, thus it is very important to remember the information both from the past and future time steps. In a recurrent neural network (RNN) for the vanishing gradient problem, it is not possible for the learning algorithm to remember the long-term dependencies. However, for correctly identifying the named entity, it is very crucial to remember this information and long short-term memory (LSTM) based RNN solves this matter e.g. "এই বছর কাঁঠালের ভালো ফলনের কারণে কৃষকরা লাভবান হয়েছে।" (Sabic likes to eat pomegranate). Here Sabic "সাবিক ডালিম খেতে পছন্দ করে।" is a person entity and pomegranate (সাবিক) is a fruit which is an object entity. RNN will correctly identify these two entities but in this sentence: "(ডালিম)" (According to the declaration of the court, the owner of that shop is Dalim), Dalim ("আদালত এর ঘোষণা মতে ঐ) is a person entity and the learning algorithm cannot correctly identify it because it has to remember long dependency. However, as we are using LSTM based RNN, it produces the correct output because LSTM units can easily store the previous long-term dependencies. In NER problem, LSTM based RNNs also

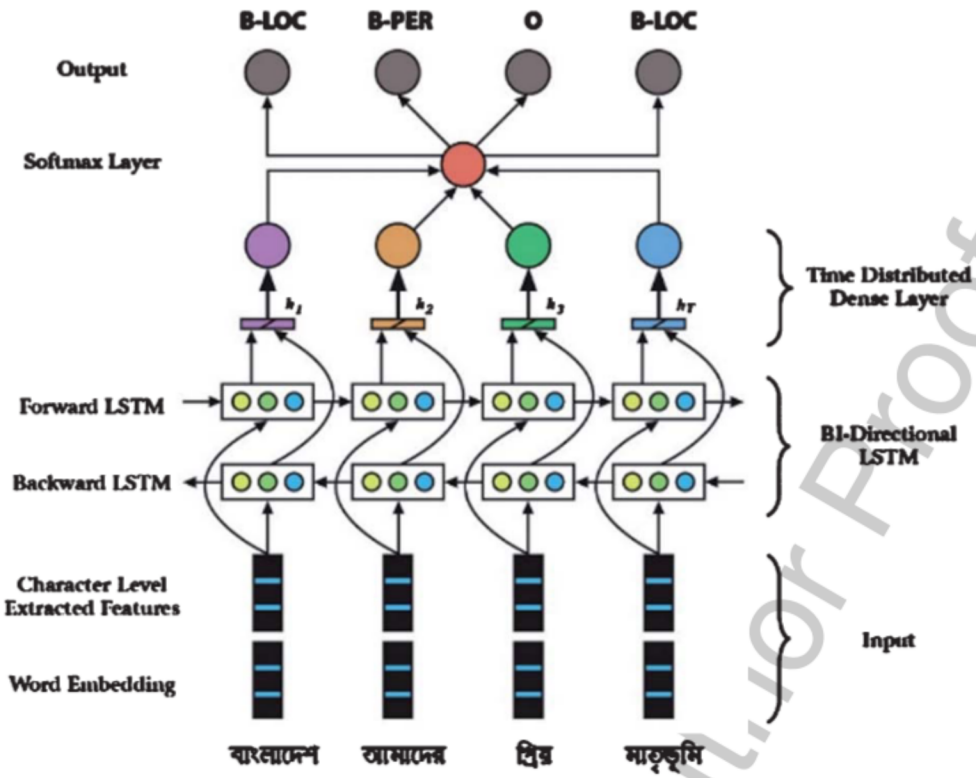

Fig. 5. A Bi-LSTM-CRF model with the concatenated input of character level extracted features and word embedding. (ref. [4]).

have limitations in some cases where the same word can be classified into different entities based on the context e.g. "দোকানের মালিকানা ডালিমের।"" (Yesterday the price of the pomegranate was very high in the market). Within this sentence pomegranate ((ডালিম)) indicates an object entity but in the previous example it was person entity. So if we only use LSTM based RNN it may predict it as a person entity due to the fact that the learning algorithm faces that word as a person entity most of the time. The future time step features are also very momentous since we have to feed them into the neural network in order to predict the entity correctly. For example, the words after "গতকাল বাজারে ডালিমের দাম (pomegranate) are very influential in order to correctly predict the output, so the bidirectional LSTM (Bi-LSTM) does this job perfectly. Bi-LSTM RNN is the combination of a forward LSTM RNN layer and a backward LSTM RNN layer. We feed the sentence twice in Bi-LSTM, first from left to right and then from right to left. For a given time step, the Bi-LSTM RNN has the access of both past and future input features and using those features the learning algorithm can correctly analyze the context of the sentence.

4.2. Character level feature extraction

Morphologically rich languages like Bangla have special relational structure among its words, which is considered to be very useful information for tasks such as NER and POS tagging. While learning word embedding, we focused only on the semantics of a word, but the morphological structure was not taken into account in the previous approach. As a result, a lot of the useful information still remains obscure. These types of morphological structures are also referred to as inflectional system, in which a root word combined with an affix forms a new word, which is used for new context. By adding different affixes with the same root word, we can have different words with different contextual meaning. Another interesting characteristic in Bangla words is that some affixes can be found that are used only for a certain type of Named Entities. In Table – 4, we present root words (mostly Named Entities) with their affixes.

All the affixes mentioned above are mostly used with Named Entities. Thus, Character Level Feature extraction can enable us to access these types of hidden features in morphological structure. As for the

Table 4
Some examples of inflected words with their root word and affix

Entity Type	Root Word	Affix	Combined Form (Root + Affix)
Person	কল্যাণ	এর	কল্যাণের
Person	এজেন্ট	কে	এজেন্টকে
Person	সহযোগী	দের	সহযোগীদের
Location	সাতক্ষীরা	আর	সাতক্ষীরার
Location	যশোর	এর	যশোরের
Location	সিলেট	এর	সিলেটের
Organization	মন্ত্রানালয়	এ	মন্ত্রানালয়ে
Organization	হাসপাতাল	এ	হাসপাতালে
Object	চেয়ার	টি	চেয়ারটি
Object	কাঁঠাল	এর	কাঁঠালের

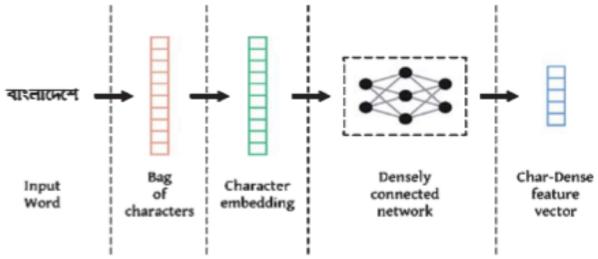

Fig. 6. A densely connected network to extract character level features. (ref [25]).

character level feature extractor, we experimented with two different approaches. The first one was presented by Santos and and Zadrozny in [6] as well as Chiu and Nichols in [4], and the second one was presented in [25] by Lee et al.

As proposed by Santos and and Zadrozny in [6], the first approach we have used here creates local features around each character using a convolutional layer. After doing so, a max operation is used to both combine and create fixed size character level feature vector of the word.

Character level features are commonly extracted through the use of CNN or RNNs, but in [25] Lee et al. showed that fully connected dense layer can surpass the results obtained by CNN or RNNs. So, as for our second approach towards character level feature extraction, we are using a Densely Connected Network (DCN). The diagram of character level feature extraction using DCN is shown in Fig. 6.

One thing that needs to be mentioned here is that we are not using some of the features that was used in the original paper [25]. For example, in the original paper the authors proposed an algorithm to split the word into k pieces. After splitting, they concatenated their corresponding one-hot encodings with the character order and word length feature. However, in our case we experimented with a much simpler version of this model. We only used character embedding to extract character level features via a DCN.

Table 5
Hyper-parameter Tuning Details

Hyper parameter	Range	Final
Hidden State Size (LSTM)	[200–250]	200
Kernel Size (CNN)	[3–9]	7
Dropout Probability (LSTM)	[0.25–0.5]	0.50

5. Training and system evaluation

5.1. Hyper parameter tuning

The NER model we are using has 3 hyper parameters which includes Hidden State Size of LSTM, Kernel Size of CNN, and the Dropout probability for Bi-LSTM layer. During the training period a range of hyper parameters were tested in which parameters with the best accuracy were selected. A table showing those details of hypermeter selection is given in Table 5.

We have done our training using mini-batch Stochastic Gradient Descent (SGD) as learning algorithm with a learning rate of 0.01 and categorical cross-entropy as objective function. As for regularization we use dropout in our hidden layers. We also tried other optimization algorithms such as RMSprop, momentum, and Adam, however they did not perform as well as SGD.

5.2. Evaluation

To test the performance of the NER model, the most common prototypical evaluation procedure is token level precision, recall, and F1 score. To evaluate with a full named entity level, these metrics play a predominant role. In our work, we use the procedure that was first proposed at SemEval'13 [39].

The SemEval (Semantic Evaluation) introduces four different types of approaches for evaluation - strict, exact, partial and type. Each of them helps to measure the performance by considering correct, incorrect, missed, partial, and spurious response.

This procedure is based on the metrics defined by Message Understanding Conference (MUC) [38]. These metric compares the response of a system with the golden annotation. While the straight forward

evaluation at token level does not consider the case of partial match or the case where the systems prediction does not exist in the golden annotation, the evaluation procedure by SemEval'13 on top of MUC defined metrics takes all of these special cases into account. Cases that SemEval takes into account are given below –

1. Partial (PAR): If the predicted labels contain chunk with partial match.
2. Missing (MIS): If a chunk is totally ignored by the system.
3. Spurious (SPU): If the prediction of the NER system is not in the golden annotation.
4. Correct (COR): If the prediction and the true labels are exactly same.
5. Incorrect (INC): If the NER system predicts a chunk into a different category other than the golden annotation.

We use the "Partial Match" evaluation approach in our case. In partial match the boundary on string surface is more flexible, so, in this approach we give partial credit for detecting a chunk of entity with a match of partial boundary or some overlapping boundary between the detected and true label. According to the SemEval'13 standard, the number of gold-standard annotations (Final Score):

$$Possible(POS) = COR + INC + PAR + MIS = TP + FN$$

And the number of annotations produced (NER system):

$$Actual(ACT) = COR + INC + PAR + SPU = TP + FP$$

Partial Match Score:

$$Precision = \frac{COR + 0.5 \times PAR}{ACT} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{COR + 0.5 \times PAR}{POS} = \frac{TP}{TP + FN}$$

Table 6
Frequency count of each of the cases (partial, spurious, correct, incorrect, missing) define by MUC

Entities	PAR	MIS	SPU	COR	INC
PER	1	0	1	2	1
LOC	0	0	1	1	0
ORG	0	0	0	0	0
OBJ	0	0	0	0	0

To illustrate the evaluation process of the model, let us go through an example. Let us assume a tagged sentence along with its true and predicted labels is as follows in Fig. 7.

Figure 7 shows an example sentence for which we will be demonstrating the evaluation process. The first line containing the IOB tags are the true labels and the second line containing IOB tags are the predicted labels. From the above example, by comparing the predicted labels with the true labels, we obtain the Table 6 of different cases that is defined by MUC for each category.

So, the calculation is as follows –

$$Possible(POS) = 3 + 1 + 1 + 0 = 5$$

$$Actual(ACT) = 3 + 1 + 1 + 2 = 7$$

Calculation of partial match score:

$$Precision = \frac{COR + 0.5 \times PAR}{ACT} = \frac{3 + 0.5 \times 1}{7} = 0.50 = 50\%$$

$$\text{Recall} = \frac{COR + 0.5 \times PAR}{POS} = \frac{3 + 0.5 \times 1}{5} = 0.70 = 70\%$$

F1 score of partial match score:

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times 0.5 \times 0.7}{0.5 + 0.7} = 0.58 = 58\%$$

Thus, the Precision, Recall and the F1 score for the sample above is 50%, 70%, and 58% respectively.

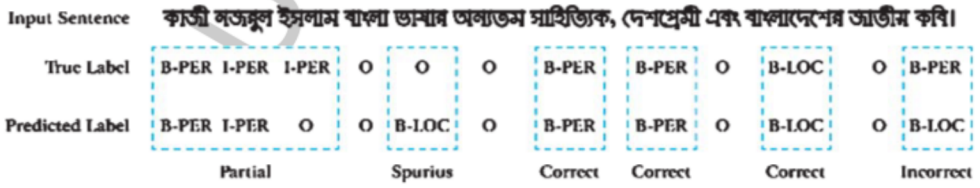

Fig. 7. An example of an annotated sentence with predicted labels for sample evaluation.

Table 7
Models which we will be experimenting with

Model	Description		
	Word embedding	Character level feature extractor	Output layer
BiLSTM	word2vec/glove	–	Softmax
CNN-BiLSTM	word2vec/glove	CNN	Softmax
DCN-BiLSTM	word2vec/glove	DCN	Softmax
CNN-BiLSTM-CRF	word2vec/glove	CNN	CRF
DCN-BiLSTM-CRF	word2vec/glove	DCN	CRF

6. Result analysis

A thorough experiment is conducted to compare the word embedding models (word2vec and glove) as well as for feature extractors at character level (CNN and DCN). Additionally, we have to choose hyper parameters pertaining to hidden state size, kernel size, number of filters, dropout probability, and output layer type (Softmax or CRF) for the entirety of this experiment.

Before beginning, in the analysis portion, we would like to mention different variations of the BiLSTM model that we experimented with in Table 7.

Initially we compare two word embedding models, word2vec and glove, as mentioned previously in Section 3.3 (Pre-trained Word Embedding).

It is quite clear from Table 8 that both word embedding – word2vec and GloVe performs at its best when we use 150 dimensional vector representation of the words. Comparing the data presented in Table – 8 we can conclude that almost all of the models performed much better when word2vec embedding is used as an input rather than the GloVe embedding. For word2vec's case the dimension 150 ended up having its highest F1 score of 62.50, and for GloVe the dimension 150 has the highest F1 score of 49.76.

As the above analysis depicts, word2vec outperforms glove embedding in the case for Bangla Language, thus any further analysis will be based on word2vec embedding only.

In regards to character level feature extraction, we experimented with two models. The first model being a CNN and the second one being a simple DCN-based extractor. We will compare these two models by showing that the addition of character level morphological features can actually boost the performance of a sequence labelling task, which in our case is Named Entity Recognition. Table 9 shows the mean (average) performance of each of the models for the following three categories - No character level feature extractor, character level feature extractor using CNN, and character level feature extractor using DCN.

(a) BiLSTM = only word embedding is used,
(b) CNN-BiLSTM=word embedding is used along with the CNN extracted character level features,
(c) DCN-BiLSTM=word embedding is used along with the DCN extracted character level features.

Table 9 portrays one of the major finding of our work, that is, character level feature extraction using a simple DCN empowers our NER system which in return boosts a significant amount of performance and outperforms the CNN based feature extractor.

The quartiles of the F1 scores of different scenario such as when the morphological feature is used and not used, are shown in the Fig. 8 as a violin plot. The regions colored with blue, light brown and green represents the F1 score distribution of the variations of the BiLSTM model when we use word embedding only, word embedding with CNN extracted character

Table 8
Performance (average) of the models for different dimensions of Word2Vec and GloVe embedding

Model\Dim	Word2Vec embedding											
	50			100			150			200		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
BiLSTM	71.69	50.29	59.12	**72.13**	52.82	60.99	**72.75**	52.59	61.05	**73.67**	50.96	60.25
CNN-BiLSTM	**72.05**	50.03	59.05	71.76	52.42	60.59	71.57	52.41	60.51	72.97	52.14	60.83
DCN-BiLSTM	67.95	55.82	61.27	69.41	**57.20**	**62.70**	69.39	**56.83**	**62.50**	73.13	51.48	60.43
CNN-BiLSTM-CRF	70.21	52.10	59.81	70.37	53.20	60.60	70.55	52.73	60.35	69.46	54.23	60.91
DCN-BiLSTM-CRF	67.15	**56.66**	**61.46**	68.59	55.50	61.34	69.30	56.27	62.11	70.09	**55.05**	**61.67**
	GloVe Embedding											
BiLSTM	70.36	23.66	35.42	68.38	24.58	36.17	**68.59**	26.78	38.53	67.91	26.20	37.76
CNN-BiLSTM	**75.06**	4.83	9.08	**79.48**	11.41	19.96	65.73	27.00	38.28	**69.46**	19.52	30.27
DCN-BiLSTM	64.22	34.84	45.18	61.69	37.08	46.32	65.84	33.84	44.71	63.29	**35.14**	**45.00**
CNN-BiLSTM-CRF	69.91	**36.41**	**47.89**	65.71	**39.90**	**49.66**	62.58	**41.29**	**49.76**	68.67	22.98	33.66
DCN-BiLSTM-CRF	65.37	30.77	41.85	63.47	36.52	46.37	67.42	29.50	41.05	63.61	32.84	43.03

Table 9
Average Performance for each of the Character Level Feature Extractor

Model	Precision (avg.)	Recall (avg.)	F1 (avg.)
BiLSTM	72.34	51.80	60.37
CNN-BiLSTM	71.11	52.10	60.13
DCN-BiLSTM	68.88	56.06	61.78

features, DCN extracted character features respectively. Each of the regions is marked with three dashed lines, showing the lower (25%), middle (50%) and upper (75%) quartiles of the F1 score distribution of all experiments. So, from the figure we can interpret that when word embedding with DCN based char-feature extractor is used, all the quartiles are higher compared to the use of CNN base char-feature extractor, or no use of char-feature extractor.

So far, selection of the word embedding model and dimensions for its word vector, the type of character level feature extractor have been discussed. Though the hidden state size of 250 has a higher average F1 score, it has been observed that during the experimental session that the hidden state size of 200 gave the overall highest F1 score. So, we are reporting the scores that we get while using hidden state size of 200, given that the difference between the averages of F1 score of hidden state size 200 and 250 is approximately ≈ 0.02, in Table 10.

The ridgeline plot in Fig. 9 shows the overall distribution of the performance along the x-axis. Three distributions of Recall (Blue shade), F1 (Green shade) and Precision (Red shade) are shown respectively for

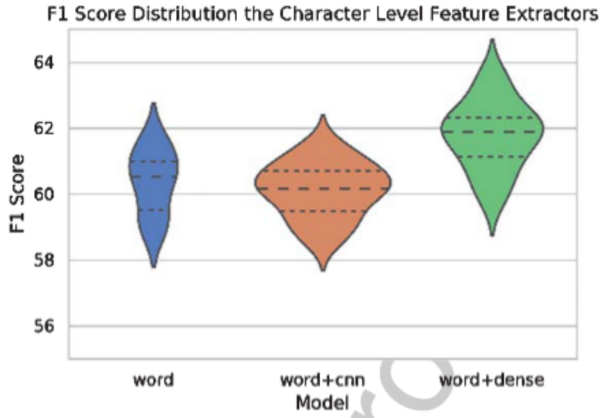

Fig. 8. F1 Score distribution the character level feature extractors.

Table 10
Best performance for each of the model

Model	Precision	Recall	F1
BiLSTM	**73.16**	53.10	61.54
CNN-BiLSTM	72.80	53.10	61.41
CNN-BiLSTM-CRF	71.22	53.45	61.07
DCN-BiLSTM	68.95	**58.62**	**63.37**
DCN-BiLSTM-CRF	69.80	57.87	63.28

each variation of the model. Comparing the score distributions shown in the Fig. 9, we can conclude that a DCN based character-feature extractor in combination with word2vec word embedding and BiLSTM performs best in the context of Bangla language.

Through examination of the result analysis, we are able to see what components and parameters perform better independently along with which combination

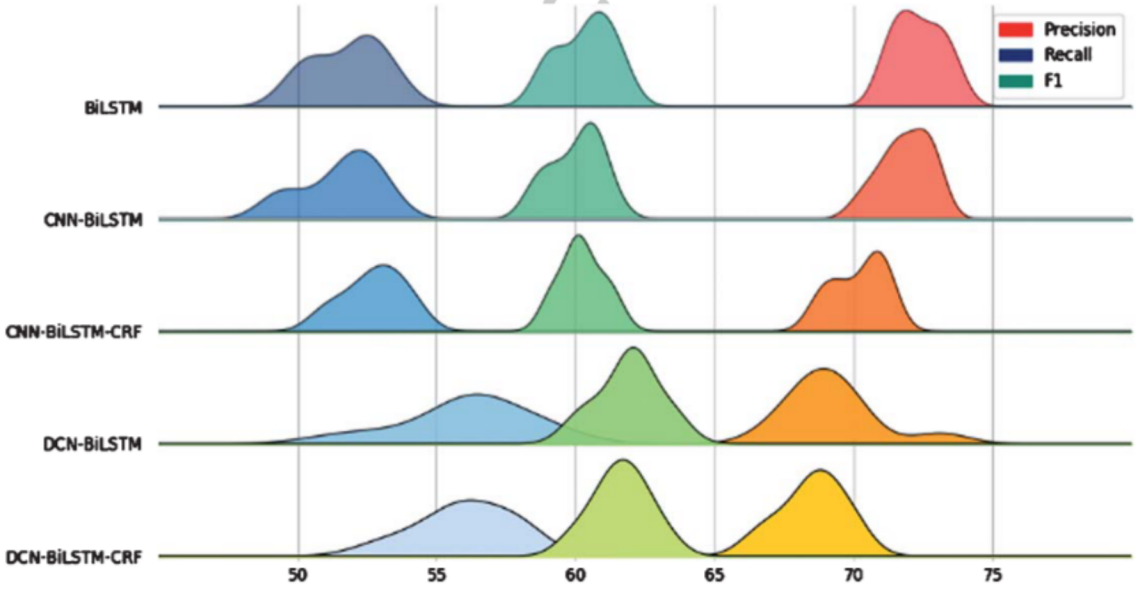

Fig. 9. Distribution of precision, recall and F1 scores of all experimental results for different variation of models.

of models, extractors, and embedding to maximize the efficiency within our system.

7. Conclusion

As the existing works on NER specific to Bangla Language are mostly dependent on feature engineering, it is important to build a system which is superior in terms of having less dependency on handcrafted features and requires less information while being robust in nature. In this work, through the use of Bi-LSTM along the word embedding and character level morphological features, we are able to detect NER within the Bangla language. Character level features of the words in any language that are morphologically rich (e.g. Bangla) can enrich any systems related to NLP, mostly for sequence labelling, which are not accessible by the use of word embedding only. This statement is also proven to be correct for Bangla Language through our work. Moreover, to extract the morphological features at character level, we have experimented with two different approaches, one is CNN based and another one is a densely connected network (DCN) with much simpler configuration than the CNN based architecture. Interestingly, the densely connected network outperformed the CNN based ones. After all the experiment and analysis, we ended up having an End-to-End architecture DCN-BiLSTM for NER which uses word embedding, character level features which are extracted by the densely connected network as its input and a Bi-LSM network for sequence labelling. Our proposed DCN-BiLSTM model has achieved a promising performance with a precision of 68.95, recall of 58.62 and an F1 score of 63.37.

8. Future work

As for the future work, our goal is to extend our current dataset of NER. In parallel to the improvement of our NER system, we would like to accomplish the next step of Information extraction, which is Relation Extraction between the recognized Named Entities and present a complete system that extracts information within a Bangla Corpus.

Acknowledgment

We want to acknowledge the support that was provided by Dr. Nabeel Mohammed, Assistant Professor at North South University during this research.

References

[1] J. Yang, Y. Zhang and F. Dong, Neural reranking for Named Entity Recognition, *CoRR*, vol. abs/1707.05127, 2017.

[2] M. Labeau, K. Löser and A. Allauzen, Non-lexical neural architecture for finegrained pos tagging, in it EMNLP, 2015.

[3] Z. Huang, W. Xu and K. Yu, Bidirectional LSTM-CRF models for sequence tagging, *CoRR*, vol. abs/1508.01991, 2015.

[4] J.P.C. Chiu and E. Nichols, Named entity recognition with bidirectional lstm-cnns, *CoRR*, vol. abs/1511.08308, 2015.

[5] G. Luo, X. Huang, C-Y. Lin and Z. Nie, Joint Named Entity Recognition and Disambiguation, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 879–888, September 2015.

[6] C.N. Dos Santos and B. Zadrozny, Learning character-level representations for part-of-speech tagging, in *Proceedings of the 31st International Conference on Interna-tional Conference on Machine Learning - Volume 32, ICML'14*, pp. II–1818–II–1826, *JMLR.org* (2014).

[7] A. Passos, V. Kumar and A. McCallum, Lexicon infused phrase embeddings for named entity resolution, *CoRR*, vol. abs/1404.5367, 2014.

[8] G. Durrett and D. Klein, A joint model for entity analysis: Coreference, typing, and linking, *Transactions of the Association for Computational Linguistics* **2** (2014), 477–490.

[9] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P.P. Kuksa, Natural language processing (almost) from scratch, *CoRR*, vol. abs/1103.0398, 2011.

[10] J. Suzuki, H. Isozaki and M. Nagata, Learning condensed feature representations from large unsupervised data sets for supervised learning, in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 636–641, Association for Computational Linguistics 2011.

[11] L. Ratinov and D. Roth, Design challenges and misconceptions in named entity recognition, in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning, CoNLL '09, (Stroudsburg, PA, USA)*, pp. 147–155, Association for Compu-tational Linguistics 2009.

[12] D. Lin and X. Wu, Phrase clustering for discriminative learning, in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume2, ACL '09, (Stroudsburg, PA, USA)*, pp. 1030–1038, Association for ComputationalLinguistics 2009.

[13] J. Hammerton, Named entity recognition with long short-term memory, in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 -Volume 4, CONLL '03, (Stroudsburg, PA, USA)*, pp. 172–175, Association for Compu-tational Linguistics 2003.

[14] J.R. Curran and S. Clark, Language independent NER using a maximum entropy tagger, in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4, CONLL '03, (Stroudsburg, PA, USA)*, pp. 164–167, Association for Computational Linguistics 2003.

[15] A. McCallum and W. Li, Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4,CONLL '03, (Stroudsburg, PA, USA)*, pp. 188–191, Association for Computational Linguistics 2003.

[16] G. Petasis, S. Petridis, G. Paliouras, V. Karkaletsis, S.J. Perantonis and C.D. Spy-ropoulos, *Symbolic and Neural Learning of Named-Entity Recognition and ClassificationSystems in Two Languages*, pp. 193–210, Springer Netherlands, Dordrecht, 2002.

[17] J.D. Lafferty, A. McCallum and F.C.N. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, (San Francisco, CA,USA)*, pp. 282–289, Morgan Kaufmann Publishers Inc., 2001.

[18] D.M. Bikel, R. Schwartz and R.M. Weischedel, An algorithm that learns what's in a name, *Machine Learning* **34** (1999), 211–231.

[19] A.E. Borthwick, A Maximum Entropy Approach to Named Entity Recognition. PhD thesis, New York, NY, USA, 1999. AAI9945252.

[20] A. Ekbal and S. Bandyopadhyay, Bengali named entity recognition using classifier combination, in *2009 Seventh International Conference on Advances in Pattern Recognition* (2009), 259–262.

[21] S. Banerjee, S.K. Naskar and S. Bandyopadhyay, Bengali named entity recognition using margin infused relaxed algorithm, in *Text, Speech and Dialogue (P. Sojka, A. Horák, I. Kopečcek, and K. Pala, eds.), (Cham)*, pp. 125–132, Springer International Publishing, 2014.

[22] L.A. Ramshaw and M.P. Marcus, Text Chunking Using Transformation Based Learning, pp. 157–176, Springer Netherlands, Dordrecht, 1999.

[23] T. Mikolov, K. Chen, G. Corrado and J. Dean, Efficient estimation of word representations in vector space, *CoRR*, vol. abs/1301.3781, 2013.

[24] A. Graves, A. Mohamed and G.E. Hinton, Speech recognition with deep recurrent neural networks, *CoRR*, vol. abs/1303.5778, 2013.

[25] C. Lee, Y.-B. Kim, D. Lee and H. Lim, Character-level feature extraction with densely connected networks, in *COLING*, 2018.

[26] J. Pennington, R. Socher and C.D. Manning, Glove: Global vectors for word representation, in *Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 1532–1543.

[27] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer, Neural architectures for named entity recognition, *CoRR*, vol. abs/1603.01360, 2016.

[28] X. Ma and E.H. Hovy, End-to-end sequence labeling via bi-directional lstm-cnns-crf, *CoRR*, vol. abs/1603.01354, 2016.

[29] C. Dong, J. Zhang, C. Zong, M. Hattori and H. Di, Character-based lstm-crf with radical-level features for chinese named entity recognition, in *Natural Language Understanding and Intelligent Applications (C.-Y. Lin, N. Xue, D. Zhao, X. Huang, and Y. Feng, eds.), (Cham)*, pp. 239–250, Springer International Publishing 2016.

[30] H. Poostchi, E.Z. Borzeshi and M. Piccardi, Bilstm-crf for persian named-entity recognition armanpersonercorpus: The first entity-annotated persian dataset, in *LREC*, 2018.

[31] F. Alam, S.A. Chowdhury and S.R. Haider Noori, Bidirectional LSTMs - CRFs Networks for Bangla POS Tagging, *ICCIT*, 2016.

[32] Wikipedia, the free encyclopedia, List of languages by number of native speakers, https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

[33] Scrapy, An open source and collaborative framework for extracting the data, https://scrapy.org/

[34] Spacy, A library for advanced Natural Language Processing in Python and Cython, https://spacy.io/

[35] Aniruddha Adhikary, Anitator. - A free tool for annotating text, https://github.com/aniruddha-adhikary/anitator

[36] Gensim, A python library for topic modelling, https://radimrehurek.com/gensim/

[37] Maciej Kula, A toy python implementation of GloVe, https://github.com/maciejkula/glove-python.

[38] Grishman, Ralph, Sundheim, Beth, 'Message Understanding Conference-6: A Brief History', In *Proceedings of the 16th International Conference on Computational Linguistics (COLING), I, Kopenhagen* (1996), 466–471.

[39] SemEval-2013 : Semantic Evaluation Exercises, International Workshop on Semantic Evaluation, June 14–15, Atlanta, Georgia, https://www.cs.york.ac.uk/semeval-2013.

Appendix A. List of Data Sources

A list of the source of online documents from where we prepared the Bangla corpus used in this work is given below –

- The Daily Ittefaq, http://www.ittefaq.com.bd/
- Bangladesh Pratidin, http://www.bd-pratidin.com/
- Kaler Kantho, http://www.kalerkantho.com/
- Bangla Wikipedia, https://bn.wikipedia.org/wiki/