Received February 26, 2020, accepted March 17, 2020, date of publication March 23, 2020, date of current version April 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982427

BANNER: A Cost-Sensitive Contextualized Model for Bangla Named Entity Recognition

IMRANUL ASHRAFI, MUNTASIR MOHAMMAD, ARANI SHAWKAT MAUREE, GALIB MD. AZRAF NIJHUM, REDWANUL KARIM, NABEEL MOHAMMED, AND SIFAT MOMEN
Department of Electrical and Computer Engineering, North South University, Dhaka 1229, Bangladesh
Corresponding author: Nabeel Mohammed (nabeel.mohammed@northsouth.edu)

This work was supported by the Special Grant of the Ministry of ICT, Bangladesh, under Grant 56.00.0000.028.20.004.19-248.

ABSTRACT Named Entity Recognition (NER) is a task in Natural Language Processing (NLP) that aims to classify words into a predetermined list of Named Entities (NE). Many architectures have produced good results on high resourced languages like English and Chinese. However, the NER task has not yet achieved much progress for Bangla, a low resource Language. In this paper, we perform the NER task on Bangla Language using Word2Vec and contextual Bidirectional Encoder Representations from Transformers (BERT) embeddings. We propose multiple BERT-based deep learning models that use the contextualized embedding from BERT as inputs and a simple statistical approach for class weight cost sensitive learning. The modified cost-sensitive loss function was used to address the class imbalance of the data. In our modified cost-sensitive loss function, we penalize the dominant classes by taking the ratio concerning the maximum sample in a class instead of the whole dataset. This penalty is made so that the learner learns slowly for the dominant class. In addition, we experiment by adding a Conditional Random Field (CRF) layer and incorporating Focal Loss to the training process. We found the best F1 Macro score to be 65.96%, F1 Micro score of 90.64%, and F1 Message Understanding Coreference (MUC) score of 72.04%, which were calculated at Named Entity level. Our experimental results demonstrate that one of the proposed models, which jointly optimizes for the CRF loss and class weighted cost-sensitive loss according to our proposed statistical approach, achieve an improvement of over 8% F1 MUC score on a recently introduced Bangla NER dataset when compared to previously published work.

INDEX TERMS Bangla NER, Bengali NER, BERT, CRF, NLP, focal loss, cost-sensitive learning, contextual embeddings.

I. INTRODUCTION

Named Entity Recognition (NER) is a task in Natural Language Processing (NLP) which seeks to classify words in an unstructured text into particular categories of Named Entities(NE), such as - person, organization, location etc. [18]. NER models can be used in multiple downstream tasks like identifying names of genes and gene products [43], recognition of chemical entities [50] and chatbots [40].

Bangla is a globally spoken language. Hence it is important that globally spoken languages like Bangla are enriched in linguistic knowledge and vocabulary. But current state of Bangla NLP has not achieved much progress in tasks like NER. This is because, unlike English, Bangla is more complex regarding both usability and vocabulary. Since Bangla is a rich language, many linguistic challenges occur while training models on it.

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Tucci.

Figure 1 provides examples which illustrate the real challenges of Bangla language more precisely. Unlike English, there are minimal indicators for tags like capitalization in Bangla as stated by [17], [42]. For instance, under *Capitalization* in the figure, the words *'chottogram'* and *'kushtia'* refer to *Chittagong* and *Kushtia*, which are names of places. From the English words, it is clearly understood that *Chittagong* and *Kushtia* are capitalized and names (of locations). Hence, it is possible that they can be tagged as *Location* through NER models trained in English Language. On the other hand, observing the Bangla sentences, it is not clear as to which parts of speech do each word signify simply by looking at the words appearance without prior knowledge of Bangla.

Challenges	Sentences
Capitalization	আমার জন্মস্থান চট্টগ্রাম (amar jonmosthan chottogram) My birthplace is Chittagong
	আমি কুষ্টিয়া থাকি (ami kushtia thaki) I live in Kushtia
Multiple Meaning	তার মায়ের নাম কবিতা (tar mayer nam kobita) His mother's name is Kabita
	তিনি কবিতা লিখেন (tini kobita likhen) She writes poem
Sentence Structure	আমি দোকান থেকে ফুলটা কিনেছি (ami dokan theke fulta kinechi) I bought the flower from the shop
	আমি ফুলটা দোকান থেকে কিনেছি (ami fulta dokan theke kinechi) The flower was bought from the shop by me
Inflection	আমি ঢাকায় থাকি (ami dhakay thaki) I live in Dhaka
	ঢাকা একটি শহরের নাম (dhaka ekti shohorer nam) Dhaka is the name of a town
Multiword Expressions	করিম এখন ননীর পুতুল হয়ে গেছে (karim ekhon nonir putul hoe geche) Karim is not a working person now
	রহিমের ভাই একজন রুই–কাতলা (rahimer vai ekjon rui katla) Rahim's brother is a famous man.

FIGURE 1. Examples of challenges in Bangla Language.

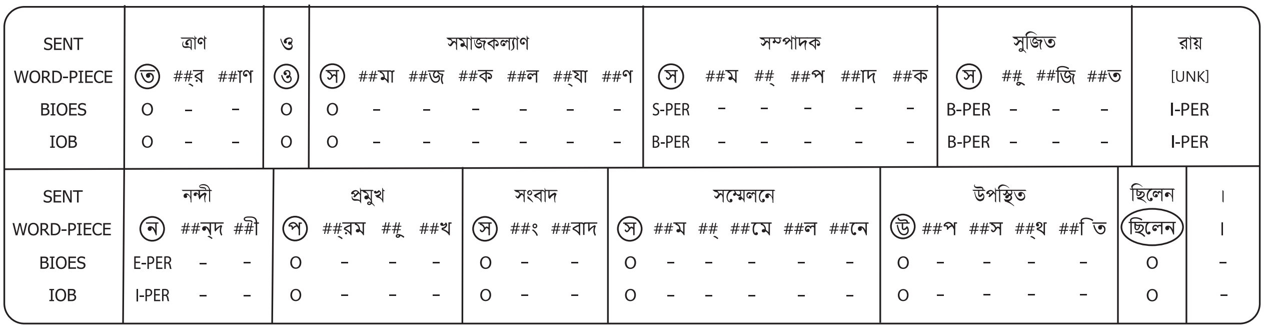
Though modern Neural Networks do not rely on capitalization anymore, we still find this problem valid for Bangla as there is no indicator. One of the major characteristics of Bangla is that same word may refer to multiple meaning based on the position of the word [26], [42]. In the figure under *Multiple Meaning*, both sentences have the same highlighted Bangla word *'kobita'* which refers to *Kobita* as *Person* in the first sentence and *Poem* as *Object* in the second sentence. It is possible to detect both entities in English as they are different words, but in Bangla it becomes more complex since the particular word is used differently in the sentences. Furthermore, as Bangla is a relatively free word order language, entities can appear in any position without changing the meaning of a sentence [17], [42]. In the figure under *Sentence Structure*, the highlighted word *'fulta'*, which refers to *flower* is in different positions. But the meaning of the sentences is identical. Moreover, Bangla has highly inflected words. Based on the context of sentences, various prepositions are added with proper nouns [17], [26], [42]. For instance, sentences under *Inflection* in the figure, the word *Dhaka* is referred to different word forms in Bangla where both of the words signify *Location*. In Bangla, there are many words with different affixes which add to the root word causing complex inflections. Finally in Bangla, there are many multiword expressions which indicate completely different meanings rather than their actual meaning. In *Multiword Expressions* in the figure, the words *'nonir putul'* indicates *soft toy* and *'rui-katla'* indicates *fish* but they actually mean *anti work person* and *well known person* respectively which are completely different. These types of scenarios make the NER task more challenging for Bangla.

In this paper, we introduce a contextualized BERT [15] based model which contributes greatly to address the above mentioned problems and also achieves state-of-the-art result in Bangla NER task in the recently introduced Bangla NER dataset [26]. The effect of contextualized embedding for the portrayed examples in Figure 1 are described in Section VI-A.

For NER task, the use of contextualized embeddings is necessary to classify the entities based on their contexts. Hence, we perform our experiments using BERT embeddings. We also perform experiments with non-contextual Word2Vec [37] embeddings in order to compare the effects of using contextual embeddings. Our empirical results yield that, using BERT along with our modified cost-sensitive loss function ameliorates the imbalanced dataset problem and produces decent accuracy. We also experimented with Focal Loss, but could not produce better results than the above mentioned model.

In recent years, neural models have gained much popularity in tasks like NER [11], [28], [35], as they can learn distributed representations from large scale unlabelled texts. NER tasks can be formulated as *sequence labeling* problems where a text is treated as a series of linguistic tags. As such, current state-of-the-art approaches typically include use of unidirectional (left-to-right) Language Models [39] or bidirectional (both left-to-right and right-to-left) Language Models [23]. There have also been use of Bidirectional Long Short Term Memory networks (BiLSTMs) [22] and a subsequent conditional random fields (CRF) decoding layer [35].

A neural model trained in a supervised setting from scratch may seem appealing, but the amount of annotated data required to carry out this approach is enormous. For low resource languages like Bangla, for which there is a dearth of large annotated datasets, this naive approach is limited in its applicability. Moreover, many effective models rely heavily on word and character embeddings for word representation [19], [41], thus increasing the importance of a large varied dataset, particularly for capturing words found in different contexts. For instance, a series of texts may have the token *George* appear many times. Ordinarily, the token refers to a person's name, but in many contexts, the token may be used as a location like *George Street*. A recent approach which can be used to ameliorate these problems is to use the embeddings found from language models trained on very large unlabelled



SENT	ত্রাণ			ও	সমাজকল্যাণ							সম্পাদক						সুজিত				রায়
WORD-PIECE	ত	##্র	##াণ	ও	স	##মা	##জ	##ক	##ল	##্যা	##ণ	স	##ম	##্	##প	##াদ	##ক	স	##ু	##জি	##ত	[UNK]
BIOES	O	-	-	O	O	-	-	-	-	-	-	S-PER	-	-	-	-	-	B-PER	-	-	-	I-PER
IOB	O	-	-	O	O	-	-	-	-	-	-	B-PER	-	-	-	-	-	B-PER	-	-	-	I-PER

SENT	নন্দী			প্রমুখ				সংবাদ			সম্মেলনে						উপস্থিত					ছিলেন	।
WORD-PIECE	ন	##ন্দ	##ী	প	##্রম	##ু	##খ	স	##ং	##বাদ	স	##ম	##্	##মে	##ল	##নে	উ	##প	##স	##্থ	##ি ত	ছিলেন	।
BIOES	E-PER	-	-	O	-	-	-	O	-	-	O	-	-	-	-	-	O	-	-	-	-	O	-
IOB	I-PER	-	-	O	-	-	-	O	-	-	O	-	-	-	-	-	O	-	-	-	-	O	-

FIGURE 2. Single sentence annotation tags from dataset and BERT wordpiece embeddings of the words.

corpus. In particular, many recent architectures use contextualized embeddings from Language Models like BERT [7], [51], ELMO [6] in order to grasp the contextual meaning of words as well as to receive embeddings rich with information which can be used in tasks with smaller datasets. In particular for NER, the use of contextual embeddings may help a model in understanding the context of each word based on its neighbors in the sentence, hence helping in understanding the difference in NER tag classes. It is also worth mentioning that other approaches rely on neural models trained in an unsupervised setting [20], [25], [44] and these may also be applicable for low resource languages.

To address the above mentioned problems we chose to use a pre-trained language model which we then used to train the extended layer and fine-tuned as necessary. We chose to use a BERT language model mainly for two reasons. Firstly because it has been shown that fine-tuning BERT models for specific language tasks result in good performance. Secondly, a pre-trained BERT multilingual model is available which also uses a limited Bangla wordpiece vocabulary. Due to constraints with computational resources, we could not train a BERT model on a large corpus from scratch. Therefore, although the pre-trained multilingual model uses a very limited set of Bangla wordpiece vocabulary resulting in many common words being divided into smaller word pieces (Fig 2), due to the inherent architecture of BERT, it is able to produce embeddings which may be interpreted as word-level embeddings. Our results indicate this to be the case.

The contributions of our paper are as follows:

- We propose a BERT based fine-tuning architecture for Named Entity Recognition for Bangla, a low resource language. Our best performing model achieves state-of-the-art results on an imbalanced dataset of 72000 sentences, outperforming previously published results by 8% as measured by macro MUC.
- As far as we could ascertain, this is the first work on Bangla NER which leverages contextualized embeddings such as BERT. We experimented on 10 different BERT based models which analyzes several cost functions and their effectiveness regarding the BERT model on our imbalanced dataset.
- We perform multiple experiments to address the issue of the imbalance present in the dataset used. To address the data imbalance, a simple statistical approach has been proposed to penalize the Cross Entropy loss function and it performs better than using Focal loss in this context. Extensive experiments have been performed to verify the effectiveness of the proposed approach for penalizing the Cross Entropy Loss function.
- Intriguingly we found that, jointly optimizing for a CRF loss and our proposed cost sensitive Cross Entropy loss perform better than optimizing for either one individually or optimizing for techniques like focal loss. We then further verify the effectiveness of this joint optimization on a Bangla POS Tagging dataset and an English NER dataset. Details of the latter two experiments are described in the Appendix.

II. RELATED WORK

Many architectures for Named Entity Recognition have been proposed in recent years. Most of the approaches are language-specific, and there have only been a few works for low resource languages. In the following sections, we review the existing models relevant to our experiments and Bangla language.

A. GAZETTEER BASED MODELS

A gazetteer is a dictionary or lexicon consisting of lists of entity names. Liu *et al.* [34] applied gazetteers to neural models. Using gazetteers, they built a model based on Hybrid Semi-Markov Conditional Random Fields (HSCRFs), where word-level scores were attained from token-label scores. In addition to that, they introduced a module for scoring candidate words by the soft match with the gazetteer. On the other hand, Mikheev *et al.* [36] showed that the use of small gazetteers of well-known names produces sufficiently good accuracy.

Architectures using Hidden Markov Model were also introduced. Zhou and Su *et al.* [55] proposed a Hidden Markov Model (HMM), which took deterministic internal feature, internal semantic feature, internal gazetteer feature and external macro context feature of words and produced a chunk of words. The chunk was then passed through a NER system to recognize and classify names. On the other hand, Chieu and Ng [10] introduced a maximum entropy-based NER. This is different from most NER models as it uses the entire document information as global information to classify the tags for each word.

B. LSTM-CRF BASED MODELS

Without using hand-crafted features like gazetteers and domain-specific knowledge like semantic features of words, Lample *et al.* [28] introduced a model with two neural architectures. The first one is based on a bidirectional LSTM and conditional random fields(CRF), and the other constructs and labels segments using a transition-based approach using shift-reduce parsers. The models were trained on the joint form of character-based representation of words from supervised setting and unsupervised word representation. On the other hand, Ma and Hovy [35] used bidirectional LSTM, Convolutional Neural Netowk (CNN), and CRF in order to acquire both the word and character representations. Similar to Ma *et al.*, Chiu and Nichols [11] used bidirectional LSTM and CNN to incorporate into a single word representation in order to perform the NER task.

Unlike using Recurrent Neural Networks (RNNs) for NER task, only CNN embeddings were used in some models. Baevski *et al.* [3] proposed a bi-directional transformer architecture which used character CNN embeddings. The architecture predicted every token in the training data through the removal of random words during training. The tokens were predicted based on the left-to-right and right-to-left context representations. The model was trained separately to produce forward, and backward representations of the words and then joined together. In case of downstream tasks like NER, they removed the masks and produced the contextual representations of each word to predict their label. Akbik *et al.* [2] proposed to leverage the internal states of a pretrained character language model in order to produce word embeddings, which they referred to as contextual string embeddings. For NER, the input sentence was turned into a sequence of characters and passed through the pretrained LM. For each word, contextual embeddings were produced, which were then sent to a BiLSTM-CRF sequence labeler to predict the label of the word.

C. CONTEXTUALIZED EMBEDDINGS BASED MODELS

Contextualized architecture works with the contextual meaning present inside of a sentence. Many models were built based on this characteristic. Beltagy *et al.* [7] proposed SCIBERT which used BERT architecture on scientific data with an unsupervised setting. For NER, the output of the SCIBERT model was fed into a linear classification layer with softmax output.

Biomedical Named Entity Recognition is a challenging task in the field of biomedical information processing. Li *et al.* [31] proposed the use of BERT embeddings and Conditional Random Fields (CRF) layer for finding the best tag sequence for a given sentence. Yuan [51] proposed a BERT-based question answering model, where they used a BERT-CRF model in order to detect mentions of the entities in sentences. In [45], Souza *et al.* employed the BERT model with CRF layer for Portuguese NER task. They also compared feature-based and fine-tuning based strategies. Xue *et al.* in [49] fine-tuned the BERT model to focus on the NER and Relation Extraction task words in medical texts. For both the task, a shared parameter layer was employed. And for NER task, CRF layer was used for sequence labeling.

On the other hand, Sharma and Daniel, Jr. [43] proposed BioFLAIR, which performed NER on benchmark tasks like Gene or Protein detection using the FLAIR [1] architecture on pretrained PubMed embeddings.

Without using one specific model, some architectures were found to compress multiple models together so that the combined model can produce a better result. Yoon *et al.* [50] proposed CollaboNet which utilized an ensemble of NER models. Each model was trained on different datasets. Each time a target model was fixed, and the input sequence was merged with the output of other collaborator models and then sent to the target model. Each time the target and collaborator models were switched during training in order to reduce the number of false positives for all cases.

D. NAMED ENTITY RECOGNITION IN BANGLA

Not much progress has been made in Bangla NER compared to other languages. There are limited remarkable works in Bangla language. Chaudhuri and Bhattacharya [9] inspired by [18] suggested a three-stage named entity detection, where Named Entity(NE) dictionary, rules for named-entity and left-right co-occurrence statistics have been used for respective steps.

Banerjee *et al.* [4], applied the Margin Infused Relaxed Algorithm proposed by Crammer and Singer [13] to learn named entities. Margin Infused Relaxed Algorithm is a learning algorithm for multiclass classification problems. It iteratively takes training sentences one by one, and updates parameters based on correct classification with a margin against incorrect ones where the loss must be as large as the incorrect ones. Since NER is a sequence labeling problem, a multiclass classification algorithm was used to classify individual words correctly.

Ibtehaz and Satter [24] proposed a partial string matching approach to identify a named entity from an unstructured text corpus in Bangla. The algorithm is based on Breadth First Search (BFS) search on a Trie data structure. The model has a closed domain dictionary, and is capable of three operations: *insert, search, and delete*. After that, they have used sub-string from the next corpus to identify the named entity and check for the named entity sub-string, which is done by BFS search. It checks all the nodes and indices of the sentence, and after matching, it performs either of the above operations. Hence, this model takes a Bangla paragraph as an input and gives entities of almost every word as an output.

Ekbal and Bandyopadhyay [16] suggested a combined output methodology of Maximum Entropy (ME), Conditional Random Field (CRF), and Support Vector Machine (SVM). They separately tested these three algorithms and finally combined and achieved a satisfying result.

Banik and Rahman *et al.* [5] proposed a Gated Recurrent Unit (GRU), a variation of RNN, based named entity recognition for Bangla online newspapers with a manually annotated dataset. In this model, the sequence of RNN is represented by a fixed-sized vector which is fed to the recurrent unit one by one.

Chowdhury *et al.* [12] proposed a Bangla NER model which uses Word level features, Parts of Speech tagger and List lookup features. The prepared custom gazetteers for Bangla and used them as list lookup features along with CRF for sequence classification.

One of the notable recent works is done by Karim *et al.* [26].They suggested a Deep Learning model which included Densely Connected Network (DCN) in collaboration with a Bidirectional-LSTM (BiLSTM) and word embedding. This approach is relatively new to Bangla NER task. Initially, they showed that sequence labeling using Word2Vec and Glove embeddings with BiLSTM produces good results. Since Bangla has inflected words which can be separated by root word and affix, they explored character level feature extraction to capture the hidden features. For feature extraction, they used two methods proposed by dos Santos and Guimaraes [41] and Lee *et al.* [29]. The authors presented that the use of DCN instead of CNN while extracting character features produced better results. They also introduced Conditional Random Field (CRF) to their architectures. In case of the use of Word2Vec and Glove embeddings, they trained the models using over 112 million words. Since Bangla lacks resources in terms of Datasets, they introduced a new dataset for the task of Bangla NER which includes four types of named entities with a large number of examples. All the experiments illustrated in this paper uses the mentioned Dataset and the results are compared with their proposed architectures. Detailed results and comparisons are illustrated in SectionVII.

III. DATASET

We performed all of our experiments on 72000 annotated Bangla sentences introduced by Karim *et al.* in [26]. All sentences were tagged in two ways: BIOES (Beginning-Inside-Outside-End-Single) tagging system and IOB (Inside-Outside-Beginning) tagging system. *Beginning* tag refers to the beginning of a name, *Inside* tag refers to the continuation of a name, *End* tag refers to the ending of a name, *Outside* tag refers to a word not under tagging vocabulary (otherwise known as *Other* tag), and *Single* tag refers to a single token.

In case of IOB tagging, *Inside* tag may refer to both continuation and also the ending of a name. For both cases, *Outside*, *Inside*, *Beginning*, *End* and *Single* tags are denoted with *O*, *I*, *B*, *E*, *S*. The *B*, *I*, *E*, *S* tags may differ based on the token, such as - for a person's name, the tags become *B-PER*, *I-PER*, *E-PER*, *S-PER* while for location they become, *B-LOC*, *I-LOC*, *E-LOC*, *S-LOC*.

We divided our dataset into three parts: training set, validation set, and test set. The training set consisted of 90% of the dataset, the validation set contained 5%, and the test set contained the rest 5% of the dataset. This was the exact split performed by Karim *et al.* in their experiments [26]. Figure 3 shows the distribution of the dataset for training our model. The tag 'O' is not shown in the figure so that the distribution of other class samples can be understood. Table 1 shows the number of samples in every tag class.

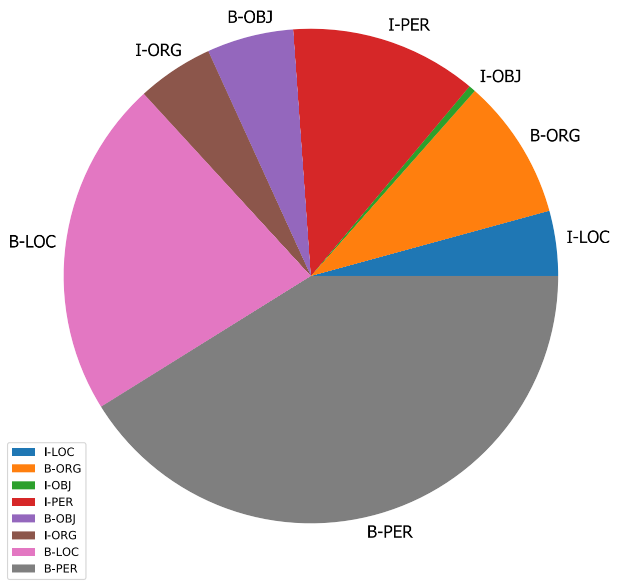


FIGURE 3. Distribution of our dataset.

TABLE 1. Per class number of samples in the dataset.

Tag Class	Number of Samples
I-LOC	7,237
B-ORG	15,684
O	7,14,867
I-OBJ	759
I-PER	20,815
B-OBJ	9662
I-ORG	8512
B-LOC	37,529
B-PER	70,025
Total Number of Samples	**8,85,090**

It is clearly visible that some of the labels, for instance, B-PER, B-LOC contain high number of samples in the dataset. However, some of the labels like I-OBJ, I-LOC have very few samples which complicate the learning process of the model.

Figure 2 shows a sample annotation from the dataset. The figure portrays the state of the input to our architecture after tokenization which is performed using BERT tokenizer. From the figure, it is clearly visible that most of the words are split in wordpieces as the BERT vocabulary for Bangla in the pretrained model is very few. Therefore, the BERT model

TABLE 2. **Layer by layer parameters and shapes for experimentation on Word2Vec model. Hidden State for BiLSTM layer = 75 (150/2) dimensions.**

Layers	Input Shape	Output Shape	Parameters
Word2Vec Layer	[batch_size, 50]	[batch_size, 50, 150]	66,913,050
BiLSTM Layer	[batch_size, 50, 150]	[batch_size, 50, 150]	272,400
Linear Layer	[batch_size, 50, 150]	[batch_size, 50, 100]	15,100
Dropout Layer	[batch_size, 50, 100]	[batch_size, 50, 100]	0
Linear Layer	[batch_size, 50, 100]	[batch_size, 50, 10]	1010
Total Parameters			**67,201,560**

TABLE 3. **Layer by layer parameters and shapes for experimentation on BERT model. Hidden State for BiLSTM layer = 384 (786/2) dimensions.**

Layers	Experiments	Input Shape	Output Shape	Parameters
BERT Layer	All Experiments	[batch_size, 50]	[batch_size, 50, 768]	177,853,440
BiLSTM Layer	BERT + BiLSTM	[batch_size, 50, 768]	[batch_size, 50, 768]	7,090,176
Linear Layer	All Experiments	[batch_size, 50, 768]	[batch_size, 50, 512]	393,728
Dropout Layer	All Experiments	[batch_size, 50, 512]	[batch_size, 50, 512]	0
Linear Layer	All Experiments	[batch_size, 50, 512]	[batch_size, 50, 10]	5130
CRF Layer	BERT + BiLSTM + CRF	[batch_size, 50, 10]	[batch_size, 50]	120
Total Parameters				**185,342,594**

yields less word-level embeddings which is a downside for the model as it cannot learn word-level entities.

IV. PROBLEM DEFINITION

Given a sentence $s = (w_1, w_2, \ldots, w_n)$, where each of $w_1, w_2, w_3, \ldots w_n$ are tokens, the required output is a sequence of labels such that $y = (y_1, y_2, \ldots, y_n)$. For this specific task, we use the standard IOB annotation for named entities which are classified into 9 categories.

V. METHODOLOGY

In this section, we introduce an overview of our proposed architectures in subsection V-A. We then explain each component of our architecture in subsections according to the following maanner: Word2Vec (V-B), BERT (V-C), BiLSTM (V-D), Linear Layer (V-E), CRF (V-F), Focal Loss (V-G). If the reader is familiar with the methodologies, these sections can be skipped. Our proposed statistical approach and training methods are described in subsections V-H and V-I. Tables 3 and 2 denote the layer by layer parameter count of our proposed model along with our experimental Word2Vec model. It also shows architecture specific layers used throughout our experiments.

A. OVERVIEW

Figure 4 shows the overall architecture of our model. Sentences taken from the dataset are first inserted into the BERT model to extract feature representations. In case of Word2Vec experiments, the representations of each sentence are taken based on pretrained embeddings. These representations are then sent to a Bidirectional LSTM network. The outputs of BiLSTM network are then sent to a Fully Connected (FC) layer to finally produce tag sequences. For experiments with CRF layer, the hidden states produced from

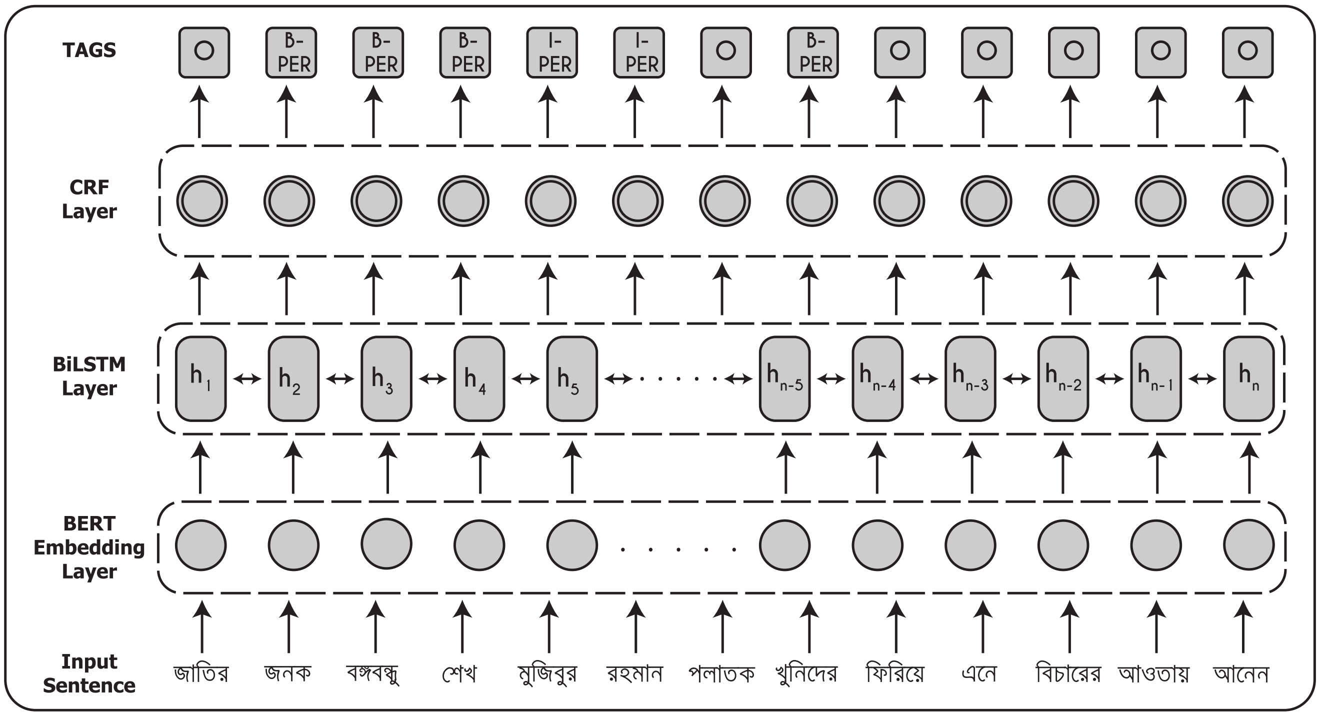


FIGURE 4. **Architecture of the model.**

the BiLSTM network are then used as feature vector inputs to the CRF (Conditional Random Fields) layer followed by a Fully Connected (FC) layer to finally produce the probability distribution of the token level tags. In the following sections, we describe the architecture components separately. We also describe our proposed cost-sensitive learning methodology in section V-H.

B. Word2Vec

Word2Vec [38] is a neural language model that creates distributed representations of words based on dependence on one another in sentences. The word embeddings produced by Word2Vec represent close spatial representations for similar words. Word2Vec embeddings are typically gained through either of the two variants: Continuous Bag of Words (CBOW) model and Skip-grams model. The Word2Vec embeddings used for our experiments were produced using CBOW model. Hence we focus only on the CBOW model here.

In the CBOW model, based on the neighboring words, the target word is predicted. Let $w_1, w_2, w_3, \ldots, w_n$ be the input sequence of the model and W be a weight matrix for the input. Then -

$$v_{wI} = \frac{1}{C} W \sum_{i=1}^{C} w_i \tag{1}$$

Here v_{wI} is the average of the input vectors which are weighted by W and C represents the context window. v_{wI} contains the average of the vectors of the words which are within the context window C. Again let w_O be the corresponding output target word and its vector form be v_{wO}. Then -

$$p(w_O|w_1, w_2, \ldots, w_C) = \frac{exp(v_{wI}^T v_{wO})}{\sum_{j=1}^{C} exp(v_j^T v_{wO})} \tag{2}$$

The objective function of the model then becomes -

$$\frac{1}{N} \sum_{n=1}^{N} \sum_{-C \leq k \leq C} log(p(w_n|w_{n+k})) \tag{3}$$

For the named entity task, we used Word2Vec embeddings pretrained on a Bangla Corpus, in order to compare the effects of contextualized and non-contextualized embeddings. Word2Vec embeddings also have the added advantage of producing word-level embeddings unlike the Multi-lingual BERT, where words are divided into word-pieces. For the words which were not in the Word2Vec vocabulary, we used the *unknown* token.

C. BIDIRECTIONAL ENCODER REPRESENTATION FROM TRANSFORMERS (BERT)

Bidirectional Encoder Representations from Transformers (BERT) [15] is a recently proposed architecture which is used for pre-training a Transformer [46]. It produces deep bidirectional representations of words in unlabelled text based on contextual relations of the words to its surroundings. It produces word-piece embeddings of words, based on its vocabulary. BERT pretraining is done using a masked language model (MLM), which randomly masks words that the model will predict and calculate the loss, and next sentence prediction (NSP) task where it has to predict the next sentence from a current sentence.

Let W_1, W_2, …., W_5 are the words of a sentence. W_4 is randomly masked using the [MASK] token. Then the output of the words in the sentence becomes O_1, O_2, …, O_5. The outputs are then fed through a Block containing two Fully Connected (FC) Layers, a GELU [21] layer, and a normalization layer. The output of the Block is the sentence along with the predicted value of the masked token.

For our named entity task, we extract the representation of our input sentences using a pretrained BERT model. Since pretraining a BERT model is computationally expensive, which is one of our main constraints, we use the *bert-base-multilingual-cased* pretrained model [48] which was trained on 104 languages including Bangla. The model has 12 layers with 768 hidden layers, 12 headed attention layers, and 110M parameters. Given sentence $s = (w_1, w2, \ldots, w_n)$, the tokenized input becomes -

$$T_i = wordpiece(w_{i1}, w_{i2}, \ldots, w_{iN}) \tag{4}$$

where *wordpiece* denotes the tokenizer and T_i represents the input vector containing individual tokenized tokens of i^{th} sentence. If a word is not present in BERT vocabulary, *wordpiece* tokenizer divides the word into word-pieces. The word then takes the following form -

$$wordpiece(w_{ij}) = m_{ij1}, m_{ij2}, \ldots, m_{ijM}; \quad j = 1, 2, \ldots \tag{5}$$

Here, w_ij is the j^{th} word of the i^{th} sentence and m_{ijk} is the k^{th} word-piece token of w_{ij} where $1 \leq k \leq M$ and M being the length of word-piece tokens. m_{ij1} denotes the head word-piece token of w_{ij}.

BERT accepts up to 512 input tokens. For our model, we take the maximum length of tokens in a particular batch and zero-pad the inputs to the maximum length. The output from BERT can be denoted as -

$$R_i = BERT(w_{i1}, w_{i2}, \ldots, w_{iN}) \tag{6}$$

where R_i is the word-piece embeddings of the i^{th} input sentence.

The Figure 2 represents the outputs of a sentence after BERT tokenization. The circled word-pieces denote the head word-piece for each word. We assign the tags of the individual words to the head word-piece token after tokenization. As the BERT model is essentially a set of stacked Transformers [46], the calculation of the embedding of each wordpiece is influenced by the embeddings of the other wordpieces of the same word due to use of self attention. Therefore, any wordpiece within a word may be used as a representative embedding of the word. However, to simplify our choice we have chosen to use the first embedding of the wordpiece of a word as representative of the entire word. Nonetheless, more experimentation can be done regarding this approach in future work.

D. BIDIRECTIONAL LSTM LAYER

LSTM (Long Short Term Memory) [22] is an effective network architecture that models sequential information for specific tasks. Bidirectional LSTM (BiLSTM) networks are variants of LSTM networks that performs computation on both direction of a sequence, unlike LSTM networks which compute either of the two. Since the BiLSTM networks have information about backward and forward hidden states, it is advantageous in sequence tagging tasks like NER [11], [28], [35].

The BiLSTM Layer takes a sequence of the input sentence. In our case, for the experiments with BERT embeddings, the inputs are representations produced from BERT, $R = (R_1, R_2, \ldots, R_n)$. On the other hand, for the Word2Vec embeddings experiments, pretrained embeddings of the sequence were taken. The output of the network is a set of hidden states for each input vectors, $(h_1, h_2, \ldots, h_n)$. The final hidden state is the concatenation of the backward and forward hidden states. Here, r_i denotes the i^{th} token of input R_i. Hence, the output becomes,

$$h_i^b = LSTM(r_i, h_{i-1}^b), h_i^f = LSTM(r_i, h_{i+1}^f) \tag{7}$$

$$h_i = concat(h_i^b, h_i^f) \tag{8}$$

where h_i^b and h_i^f denotes the backward and forward hidden states of the network respectively.

E. LINEAR LAYER

The BERT architecture does not require any complex or deep layers to fine-tune by a novel dataset [15]. After feeding an input sequence to the BERT model, contextual embeddings of each token is produced. In some experiments, we forwarded these embeddings to a BiLSTM layer for further sequence modeling. The output of the BiLSTM is then projected to 9 logits using fully connected layers, where the weights of the fully connected layers are shared over the sequence dimension of the BiLSTM outputs. The output of the fully connected layer is then passed through a softmax activation function to get the probability distribution over the labels. A similar setting was applied for Word2Vec experiments.

Moreover, since BERT model produces contextually meaningful sequences, and Transformers [46] are sequence learning models themselves, we performed experiments where the BERT embeddings were projected down to 9 logits using a fully connected layer without using a BiLSTM layer in between. This experiment was done to ensure that the BiLSTM actually provides utility. Details of the fully connected layer are shown in the Table-2.

F. CONDITIONAL RANDOM FIELD

Almost all natural language sentences have words which depend on the word positions and their frequency for its semantic meaning. It is always a good idea to consider the current and neighboring words when extracting sentence features. Linear-chain Conditional Random Field [27] is an effective way to control the structure prediction by using a series of potential functions to produce the conditional probability of the output label sequence given the input representations. However, instead of trying to maximize the probability of each individual output in a sequence, CRFs allow us to attempt to maximize the joint probability of the output sequence.

Many state-of-the-art architecture employ a CRF layer after a BiLSTM network for sequence labelling tasks [2], [28], [35]. Lample *et al.* [28] introduced a model with two neural architectures for the NER task. One of the architectures were based on a BiLSTM and CRF layer. The CRF layer was used for predicting tag sequences. Similar to Lample *et al.*, Ma and Hovy [35] proposed the use of BiLSTM and CNN and CRF in order to acquire both word and character level representations, which is different from the architecture proposed by Lample *et al.* where CRF layer was used for sequence labelling. Again, different from Lample *et al.* and Ma *et al.* where BiLSTM, and BiLSTM, CNN and CRF networks were used to gain word representations, Akbik *et al.* [2] proposed to leverage pretrained Language Models in order to find contextual embeddings. These embeddings were then sent to a BiLSTM and CRF layer for sequence labeling.

On the other hand, many recent state-of-the-art architectures used CRF layer after a contextual Language Model [31], [45], [49], [51]. Both Li [31] and Yuan *et al.* [51] used BERT-CRF model for sequence labelling tasks. In [45], Souza *et al.* employed the BERT model with CRF layer using feature-based and fine-tuning based strategies for Portuguese NER task. Different from Li *et al.*, Yuan *et al.* and Souza *et al.*, Xue *et al.* in [49] fine-tuned the BERT model to focus on the NER and Relation Extraction task specific words in medical texts. For both the tasks, a shared parameter layer was employed. And for NER task, CRF layer was used for sequence labelling.

In all of the above architectures, CRF layer was used for sequence labeling task. CRF layer produces graphical models that can successfully encode relations between tokens based on neighboring context. Hence, using CRF layer in order to produce contextual sequences improves the ability of a model for tag prediction. Ma *et al.* used the CRF layer along with BiLSTM and CNN layers in order to produce word and character level representations. These representations hold the contextual meaning of the words based on the neighbours.

Following the previous section, we take the output results from the linear layer $y = (y_1, y_2, \ldots, y_n)$, where y_i denotes the corresponding tags of the tokens. After that, we calculate the conditional log probability of the produced sequence given the input sequence using CRF layer.

CRF layer relies on the first-order Markov assumption over labels. Let $1 \leq t \leq T$ where T is the total number of observations for a particular sequence and t is an observation at any particular time index, x is the whole input observation sequence and y is a possible output sequence, then -

$$p(y|x) = \frac{exp(\sum_{t=1}^{T} U(x_t, y_t) + \sum_{t=1}^{T-1} T(y_t, y_{t+1}))}{Z(X)} \tag{9}$$

$$Z(X) = \sum_{y} exp(\sum_{t=1}^{T} U(x_t, y_t) + \sum_{t=1}^{T-1} T(y_t, y_{t+1})) \tag{10}$$

Here, U denotes a function which calculates the unary or emission score, that is, how likely y_t is to occur based on input, x_t and function T calculates the transition score of y_{t+1} based on input, y_t i.e. the observation on previous time index t. The denominator $Z(X)$ is a normalization factor found by summation of emission and transmission scores over all possible y sequences. Calculating Z naively is difficult and is found efficiently using the forward-backward algorithm. Phrased thus, a CRF layer attempts to find the best prediction sequence, in terms of the joint probability of the output labels, given a particular input. This helps the CRF layer to understand the contextual meaning of each word in a sentence and also what is likely to come after based on the current word.

The loss function of CRF layer is calculated as negative likelihood loss as shown in equation 9.

$$\begin{aligned} L &= -log(p(y|x)) \\ &= -log(\frac{exp(\sum_{t=1}^{T} U(x_t, y_t) + \sum_{t=1}^{T-1} T(y_t, y_{t+1}))}{Z(X)}) \\ &= log(Z(X)) - log(exp(\sum_{t=1}^{T} U(x_t, y_t) + \sum_{t=1}^{T-1} T(y_t, y_{t+1}))) \\ &= log(Z(X)) - (\sum_{t=1}^{T} U(x_t, y_t) + \sum_{t=1}^{T-1} T(y_t, y_{t+1})) \end{aligned} \tag{11}$$

During inference, in order to find the best tag sequence for a given sequence, the Viterbi Algorithm is used. The algorithm is implemented by simply keeping track of the maximum score of the above equation 9 in each time index. The best sequence is then found by traversing through the maximum score from the last time index to the first time index.

G. FOCAL LOSS

Focal loss is used to address the training bias when using an imbalanced dataset. It focuses on false negatives produced from each class by down-weighting the correctly predicted training samples of each class. Focal Loss was introduced in [32], where the authors proposed to reshape the Cross Entropy loss function to down-weight the correct training examples and therefore focusing on false negatives. They proposed adding the modulating factor $(1-\rho)^\gamma$ with the cross entropy loss. They described the focal loss as:

$$FL(\rho_t) = -(1 - \rho_t)^\gamma log(\rho_t) \tag{12}$$

Here, γ is the focusing parameter ($\gamma \geq 0$). Loss is unaffected and near to 1 when ρ_t is small and example is wrongly classified. γ is a hyper-parameter which is adjusted based on the down-weight to be applied for each correctly classified example. The value of γ was set as 2 according to the findings provided in [32].

H. COST SENSITIVE LEARNING

Categorical cross entropy is the loss function of choice when training a neural network on a classification task with more than two labels. If for a sample i of the training data y_i is the ground truth probability and p_i is the predicted probability (usually from a softmax layer), then the Categorical Cross Entropy can be calculated for N different training samples by equation 13.

$$Loss = -\sum_{i=0}^{N} y_i * log(p_i) \tag{13}$$

In an imbalanced data-set, using this approach poses a problem. The classification model, when being trained, will be biased towards the classes with large number of training samples and will perform poorly for the subservient classes. Since Cross Entropy assigns equal weights to all classes irrespective of the number of training examples, the overall loss helps in faster feature learning for the dominant classes. As such, the loss becomes low for individual subservient classes, and hence, fewer features can be learned.

In order to address this problem, many cost-sensitive learning approaches have been used. Zhang *et al.* in [52] proposed an adaptive differential evolution method in order to find the misclassification cost of each class. The adaptive differential evolution method of [54] was used to find the misclassification costs of each class based on the training data. They argued that this method did not require prior domain knowledge. Li *et al.* in [30] proposed a method of penalizing misclassification through the use of Gini index [8] and information gain measure. A composite attribute measure was introduced based on the Gini splitting measure and Kullback-Leibler (KL) divergence of the input samples in the dataset. This hybrid attribute measure incorporating the total misclassification cost of each attribute split dataset produced the misclassification cost for each attribute. Zhang *et al.* in [53] used a differential evolution algorithm in order to optimize the cost matrix, where the misclassification cost of each class is kept. Liu *et al.* in [33] proposed cost-sensitive feature learning through optimizing F-measure through rigorous theory guidelines for each class.

We use a simple but effective statistical approach to cost-sensitive class feature learning. By using this approach, higher cost (higher loss value) is assigned to the subservient classes for misclassification and lower cost (lower loss value) for the dominant classes. Unlike Focal Loss, where the modulating factor assigns the down-weighting cost only on hard-negatives irrespective of class, here, each class is assigned a cost to emphasize more on the misclassification made on subservient classes.

Let **Class** $= N_1, N_2, N_3, \ldots, N_n$ be the set of number of training examples in each class, $\mathbf{C_{class_i}}$ be the cost assigned to class i and γ be an adjusting factor used for adjusting the class weights. For all experiments in this paper, the value of γ was set as 10.

$$C_{class_i} = \frac{N_i}{max(\mathbf{Class})} * \gamma; \quad i = 1, 2, 3, \ldots \tag{14}$$

Let C^{max} and C^{min} be the maximum and minimum values from the list of C_{class_i} calculated, where $class_i \neq class_{max(Class)}$. Again, let $C_{\hat{class_i}}$ be the new cost within the range **a** and **b**. For this experiment, a, b was set as 0.5 and 0.9 respectively.

$$C_{\hat{class_i}} = \frac{(b-a) * (C^{max} - C_{class_i})}{C^{max} - C^{min}} + a \tag{15}$$

The class with the most training examples was assigned a cost of 0.5. These costs were then introduced to Cross-Entropy loss. The cost-sensitive Cross Entropy loss then becomes -

$$Loss = -\frac{1}{N}\left(\sum_{i=0}^{N}\sum_{j=0}^{J} C_{\hat{classof(y_{i,j})}} * y_{i,j} * log(p_{i,j})\right) \tag{16}$$

where $J =$ *total no. of words in the sentence,*

$N =$ *total no. of sentences in the dataset* $classof =$ *Returns the class of* j^{th} *label of the* i^{th} *sample*

The cost-sensitive Cross Entropy loss, emphasizes the misclassification made primarily on the subservient classes by use of cost-sensitive class weight. A comparison between other methods of acquiring misclassification costs is not in the scope of this paper and is kept for future works.

I. TRAINING PROTOCOLS

In total results of 11 different models will be presented which have been trained using two protocols. Models are differentiated based on layers, training steps, and fine-tuning steps.

In *Protocol-I* models were trained for 20 epochs. Two different types of embedding layers were used. When using Word2vec embeddings, the embedding layer was initialized with pre-trained word embeddings. When using BERT embeddings, the pre-trained multi-lingual BERT model was used. For all experiments, the embedding layers were frozen for the first 10 epochs, training only the latter added layers. For the next 10 epochs, the embedding layers were trained along with the latter layers. For first 10 epochs, the added layers had a learning rate of 0.001. And for the remaining epochs, the learning rate for the embedding layers was set to 5×10^{-5} and the added layers to 0.0005. For all experiments, the batch size was set to 32.

Protocol-II is used to fine-tune a model obtained by training on *Protocol-I*. However, unlike the fine-tuning done before, in this protocol the model is fine-tuned on a modified loss function by training for a further 3 (three) epochs. All layers are trained using the learning rates of *Protocol-I*. The modification to the loss functions is different in each case, which will be discussed in detail in later sections. It is worth noting that in *Protocol-I*, the training runs for 20 epochs. We choose the model parameters for the epoch on which the model had the best validation Macro FI to use in *Protocol-II*.

J. EVALUATION METRICS

For evaluating our model, we took 3,564 sentences as a test set. We calculated precision, recall, and F1 scores as the evaluation metrics. As for evaluation with the F1 score, we took both the micro average and macro average to further analyze the results. We also took the Message Understanding Coreference (MUC) [47] score to understand the partial match accuracy of the tags better at individual token level. The MUC metric gives partial credit for correctly recognizing part of an entity boundary. The MUC metric was taken in order to compare the results of the models with the scores published by Karim *et al.* in [26].

The MUC metric generally evaluates the target and the predicted sequences on 5 aspects: Correct (COR) - if both the sequences are exactly the same; Incorrect (INC) - if the sequences don't match; Partial (PAR) - if the sequences are somewhat similar but not the same; Missing (MIS) - if the predicted sequence is missing tags that are present in the target sequence; Spurius (SPU): if the predicted sequence produces tags that are missing in the target sequence. The metric calculation, according to SemEval'13 standard, is as follows:

$$\begin{aligned} Possible(POS) &= COR + INC + PAR + MIS \\ &= TP + FN \\ Actual(ACT) &= COR + INC + PAR + SPU \\ &= TP + FP \\ Precision &= \frac{COR + 0.5 * PAR}{ACT} \\ Recall &= \frac{COR + 0.5 * PAR}{POS} \\ F1 &= \frac{2 * Precision * Recall}{Precision + Recall} \end{aligned} \tag{17}$$

K. MODEL SELECTION

All model configurations were trained on the training data following one of the two protocols mentioned above. Model performance, based on Macro F1, was calculated for each epoch on the validation set. The epoch for which the model parameters gave the best validation set Macro F1 was chosen and evaluated on the test set. These test set results are reported in the following sections.

VI. EXPERIMENTS

Table 4 demonstrates the summary of all the experiments done.

Table 7 demonstrates the F1 scores for all the experiments along with the above mentioned MUC scores for the test set. Table 8 demonstrates the precision, recall scores for all the experiments. We further explain and analyze the results for each experiment below.

TABLE 4. Summary of experiments carried out.

Architecture	Summary
Word2Vec + BiLSTM	Using Word2Vec pretrained embedding as input to the BiLSTM layer and predicting the sequence by using Protocol-I
BERT	Using pretrained BERT embeddings as input to a fully connected layer and predicting the sequence by using Protocol-I
BERT + BiLSTM	Using pretrained BERT embeddings as well as training a BiLSTM layer followed by the fully connected layer and predicting the sequence by using Protocol-I
BERT + BiLSTM + Focal	Using BERT BiLSTM along with Focal Loss in addition to Cross Entropy and predicting the sequence by using Protocol-I
BERT + BiLSTM + CW	Using BERT BiLSTM along with Cost Sensitive learning in addition to Cross Entropy and predicting the sequence by using Protocol-I
BERT + BiLSTM + CW (Protocol-II)	Using BERT BiLSTM along with Cost Sensitive learning in addition to Cross Entropy and predicting the sequence by using Protocol-II
BERT + BiLSTM + CRF	Using BERT BiLSTM along with CRF instead of Cross Entropy and predicting the sequence with Viterbi Algorithm by using Protocol-I
BERT + BiLSTM + CRF + CE	Using BERT BiLSTM along with CRF in addition to Cross Entropy and predicting the sequence with Viterbi Algorithm by using Protocol-I
BERT + BiLSTM + CRF + CE	Using BERT BiLSTM along with CRF in addition to Cross Entropy and predicting the sequence with Viterbi Algorithm by using Protocol-II
BERT + BiLSTM + CRF + CW	Using BERT BiLSTM along with CRF in addition to Cross Entropy with Cost Sensitive learning and predicting the sequence with Viterbi Algorithm by using Protocol-I
BERT + BiLSTM + CRF + CW (Protocol-II)	Using BERT BiLSTM along with CRF in addition to Cross Entropy with Cost Sensitive learning and predicting the sequence with Viterbi Algorithm by using Protocol-II

A. EFFECTS OF CONTEXTUAL EMBEDDING

We first experimented with Word2Vec and BERT embeddings in order to compare the effects of non-contextual and contextual embeddings. The Word2Vec embeddings were initially trained on a corpus of Bangla text containing 8,154,503 sentences collected from Wikipedia and various online news sources such as - Ittefaq, Bangladesh Pratidin, Kaler Kantho. The learnt vectors were of 150 dimensions and covered a vocabulary of $4, 46, 087$ unique words. This vocabulary covered almost all the words in our NER training set, with very few unknown words. For these edge cases we introduced an *unknown* token. The word embeddings were of 150 dimensions. Since, Word2Vec produces non-contextual embeddings, we used BiLSTM in order to produce sequentially and contextually meaningful sequences.

The Word2vec embeddings were compared with the performance from BERT Embeddings. While the multilingual model has a combined vocabulary of over $110, 000$ wordpieces, for our Bangla NER training dataset we see only about 950 Bangla wordpieces being used. The output embeddings of the last BERT layer were used in two different architectures. In the first architecture, the embeddings were used directly to make label prediction by passing them through fully connected layers. The second architecture added a BiLSTM to process the BERT embeddings. The BiLSTM layer was followed by a shared fully connected layer. The softmax activation function was used to derive probabilities from logit scores.

All the models were trained using the Protocol-I described in the Section V-I and learning rates mentioned in Section VI. For all three models, Cross Entropy was used as the loss function. If $\mathbf{y_i}$ is the ground truth probability of sample $\mathbf{i}$ and $\mathbf{p_i}$ is the predicted value, then the Cross Entropy loss function for N samples becomes -

$$Loss = -\sum_{i=1}^{N} y_i * log(p_i) \tag{18}$$

From Table 5, we can see that Word2Vec embeddings produced decent results on test dataset. The model scored 63.19% on Macro F1 and 71.75% on MUC F1. This result is 8.38% better than the results reported in Karim *et al.* [26]. Also, we can see that BERT model outperformed Word2Vec + BiLSTM model in terms of Macro F1, whereas the Word2vec based models achieve slightly better MUC F1. The model scored 64.54% on Macro F1 and 71.35% on MUC F1. Considering that BERT model was trained with small vocabulary compared Word2Vec embeddings, and Word2Vec embeddings produce word-level embeddings unlike the Multi-lingual BERT model, where word-piece embeddings are produced, the Multi-lingual BERT model still outperformed Word2Vec + BiLSTM model.

TABLE 5. Comparison of Results between Contextual and Non-Contextual Embeddings.

Architecture	Result		
	Micro F1	Macro F1	MUC
Word2Vec + BiLSTM	90.69	63.19	71.75
BERT	90.35	64.54	71.35
BERT + BiLSTM	**90.65**	**65.42**	**71.04**

From Table 5, we can see that BERT + BiLSTM model clearly does better than the BERT model. The model

Challenges	Sentences	Word2Vec Prediction	BERT Prediction
Capitalization	আমার জন্মস্থান চট্টগ্রাম (amar jonmosthan chottogram) My birthplace is Chittagong	B-PER O **B-LOC**	B-PER O **B-LOC**
	আমি কুষ্টিয়া থাকি (ami kushtia thaki) I live in Kushtia	B-PER **B-LOC** O	B-PER **B-LOC** O
Multiple Meaning	তার মায়ের নাম কবিতা (tar mayer nam kobita) His mother's name is Kabita	B-PER O O **O**	B-PER O O **B-PER**
	তিনি কবিতা লিখেন (tini kobita likhen) She writes poem	B-PER **O** O	B-PER **O** O
Sentence Structure	আমি দোকান থেকে ফুলটা কিনেছি (ami dokan theke fulta kinechi) I bought the flower from the shop	B-PER O O **O** O	B-PER B-LOC O **B-OBJ** O
	আমি ফুলটা দোকান থেকে কিনেছি (ami fulta dokan theke kinechi) The flower was bought from the shop by me	B-PER **O** I-LOC O O	B-PER **B-OBJ** B-LOC O O
Inflection	আমি ঢাকায় থাকি (ami dhakay thaki) I live in Dhaka	B-PER **B-LOC** O	B-PER **B-LOC** O
	ঢাকা একটি শহরের নাম (dhaka ekti shohorer nam) Dhaka is the name of a town	B-LOC O B-LOC O	B-LOC O **B-LOC** O
Multiword Expressions	করিম এখন ননীর পুতুল হয়ে গেছে (karim ekhon nonir putul hoe geche) Karim is not a working person now	B-PER O O **B-OBJ** **O** O	B-PER O **B-PER** **B-PER** O O
	রহিমের ভাই একজন রুই–কাতলা (rahimer vai ekjon rui katla) Rahim's brother is a famous man	B-PER B-PER O **B-LOC**	B-PER B-PER O **B-PER**

FIGURE 5. Handling of different challenges of Bangla using Word2Vec and BERT based model.

produced a score of 65.42% in Macro F1 and 71.04% on MUC F1. We are reporting both Macro F1 and MUC F1 scores, but in terms of model choice we have decided to treat the Macro F1 as our guiding metric, while reporting MUC F1 to compare the results with Karim *et al.* [26]. Based on this approach, the next set of experiment will be conducted by taking the BERT + BiLSTM model forward.

The use of contextualized embedding also addresses the problems demonstrated in Section I. Figure 5 illustrates the results on sentences for both Word2Vec and BERT based models. The predictions of the underlined Bangla words are highlighted using *red* and *green*, indicating incorrect and correct prediction respectively. For the *Capitalization* problem, both Word2Vec and BERT based model detected the highlighted words correctly as *Location*. In the *Multiple Meaning* section, the Word2Vec based model predicts the word *'kobita'* as *Other* entity which is wrong because the word refers to *Person* in the first sentence and *Other* in the second. However, the BERT based model was successfully able to predict the correct entity types. For the problem of *Sentence Structure*, the Word2Vec based model failed to predict the highlighted word *'fulta'* correctly in both of the sentences. Whereas, the BERT based model was able to predict the word correctly as *Object* even though it appears in different locations in the sentences but with the same meaning. As for the *Inflection* problem, both Word2Vec and BERT based models were able to identify the highlighted inflected word *'dhakay'* which represents *Location*. Furthermore for

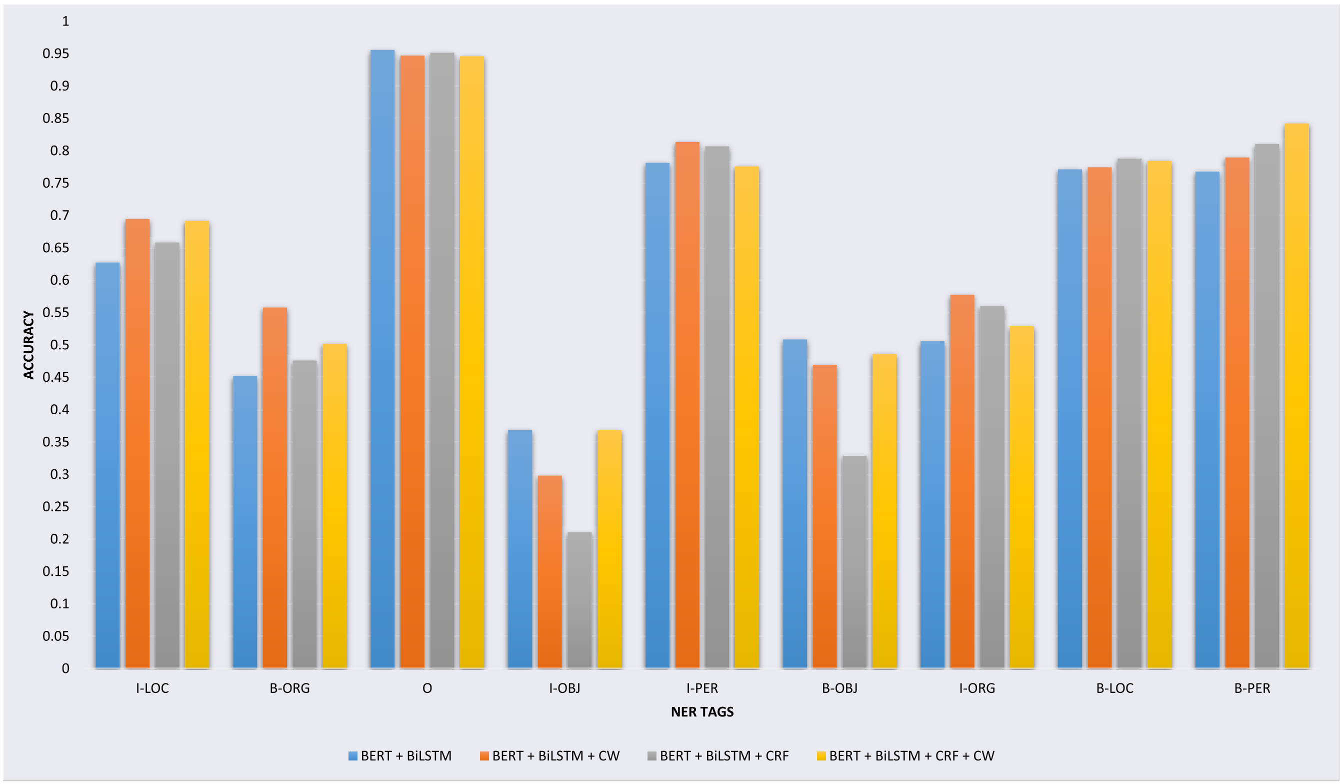


FIGURE 6. **Per class prediction comparison for different models. BERT + BiLSTM + CW(orange) produced the overall better results than BERT + BiLSTM(blue) especially in tags like B-ORG, I-LOC, I-ORG. On the other hand, BERT + BiLSTM + CRF + CW(yellow) produced the overall better results comparing against all other models especially in tag classes like I-OBJ, B-PER. Best viewed in color.**

the *Multiword Expression* problem, the first example shows *'nonir putul'* as an expression consisting of two separated words and the second example shows *'rui-katla'* as an expression consisting of two words joined with a hyphen. The Word2Vec model entirely misclassified the two expressions. But the BERT model was able to predict the highlighted expressions *'nonir putul'* and *'rui-katla'* correctly as *Person* whereas their original meaning indicates *soft toy* and *fish* which were not used here because of their position and context as a multiword expression. However it considered the separate word *'putul'* as a beginning of entity (B-PER) tag whereas it should be an inside of entity (I-PER) tag. Therefore, the BERT based model was partially correct while predicting the multiword expressions. While investigating the reason we inspected that the dataset (provided by [26]) by which we trained our models contained very few examples of multiword expressions. Though the BERT based model did not make an entirely accurate prediction, the fact that it gave a partially correct prediction engender confidence about using this approach in future follow up work to address this multiword issue.

B. INTRODUCTION OF COST SENSITIVE LEARNING

In our previous experiments, BERT + BiLSTM model produced the best results. But the results were found not to be well distributed among all classes. From Figure 6, we can see that BERT + BiLSTM model (blue) had the highest accuracy for the *Other* tag. For other tag classes, the model did not gain a well distribution. In order to address this issue, we experimented with Focal Loss. Focal Loss is a widely used objective function in Computer Vision tasks. It helps in reducing the false negatives of each class irrespective of the number of examples in it.

Since BERT + BiLSTM model produced the best results in our previous experiments, we performed our experiments on that model using Focal Loss as the objective function instead of Cross Entropy. If $\mathbf{p_t}$ is the probability of t^{th} sample, following equation 18, the loss function for N samples then becomes -

$$Loss = -\sum_{t=0}^{N}(1-p_t)^{\gamma}\,log(p_t) \tag{19}$$

In order to indicate the use of Focal Loss in a model, '+ *Focal*' was used in the name of the model.

The feature vectors of the last layer of BERT model were sent to the BiLSTM network followed by a fully connected layer. Similar to our previous experiments, the BERT + BiLSTM model was trained using the protocol mentioned in Protocol-I. Softmax function was applied on the outputs of the model.

From Tables 5 and 6, we can see that, addition of Focal Loss produced worse results than the BERT model in terms

TABLE 6. Comparison of Results upon introducing Cost Sensitive Learning.

Architecture	Result		
	Micro F1	Macro F1	MUC
BERT + BiLSTM + Focal	90.66	62.88	70.55
BERT + BiLSTM + CW (Protocol-I)	90.47	65.49	70.06
BERT + BiLSTM + CW (Protocol-II)	**90.51**	**65.51**	**70.70**

of Macro F1 and MUC F1 measure. The Macro F1 score was found to be 62.88% and MUC F1 score to be 70.55%. It was also not able to outperform the BERT + BiLSTM model. The Macro F1 score decreased to 62.88% and MUC F1 score to 70.55% from 65.42% and 71.04% respectively. We hypothesize that BERT + BiLSTM with Focal Loss as objective function was not able to outperform the BERT + BiLSTM model because, although it might be able to able to improve in terms of false negatives for subservient classes, it was not able to do so for dominant classes at the same ratio. Also the modulating factor intrinsic to Focal Loss plays a vital role in this matter. The modulating factor focuses on the false negatives of all the classes. Hence it assigns equal costs to all the classes. This approach could not tackle our initial imbalanced dataset problem. As such, the loss was still dominated by the dominant classes during backpropagation. Hence, the features of the dominant classses were learnt faster and better than the subservient classes. Therefore, the loss particularly to subservient classes were low. This in turn was reflected on the lower Macro F1 score.

In order to address the issue, we introduced an extremely simple approach of cost sensitive learning. For Cost Sensitive Learning, the class weights were generated for each class

TABLE 7. F1 Results of different experiments.

Architecture	Loss Function	Protocol	Validation			Test		
			Micro F1	Macro F1	MUC	Micro F1	Macro F1	MUC
Word2Vec + BiLSTM	Cross Entropy	Protocol-I	90.64	63.83	69.92	90.69	63.19	71.75
BERT	Cross Entropy	Protocol-I	89.99	62.02	69.66	90.35	64.54	71.35
BERT + BiLSTM	Cross Entropy	Protocol-I	90.56	63.99	70.31	90.65	65.42	71.04
BERT + BiLSTM + Focal	Focal Loss	Protocol-I	90.28	61.57	69.12	90.66	62.88	70.55
BERT + BiLSTM + CW	Cross Entropy + Class Weights (Multiplicative)	Protocol-I	90.19	63.78	69.21	90.47	65.49	70.06
BERT + BiLSTM + CW	Cross Entropy + Class Weights (Multiplicative)	Protocol-II	90.33	63.29	70.04	90.51	65.51	70.70
BERT + BiLSTM + CRF	CRF	Protocol-I	90.76	62.87	71.25	90.69	63.63	71.44
BERT + BiLSTM + CRF + CE	CRF + Cross Entropy (Additive)	Protocol-I	90.13	63.48	70.12	90.35	65.39	71.14
BERT + BiLSTM + CRF + CE	CRF + Cross Entropy (Additive)	Protocol-II	90.42	62.94	70.68	90.51	64.69	71.78
BERT + BiLSTM + CRF + CW	CRF + Class Weights (Additive)	Protocol-I	90.23	63.32	70.41	90.64	65.96	72.04
BERT + BiLSTM + CRF + CW	CRF + Class Weights (Additive)	Protocol-II	90.47	63.99	70.89	90.66	65.60	71.77

TABLE 8. Precision and Recall results of different experiments.

Architecture	Loss function	Protocol	Validation				Test			
			Micro		Macro		Micro		Macro	
			Pre	Rec	Pre	Rec	Pre	Rec	Pre	Rec
Word2Vec + BiLSTM	Cross Entropy	Protocol-I	90.65	90.64	66.40	62.53	90.70	90.69	64.76	62.28
BERT	Cross Entropy	Protocol-I	90.02	89.95	60.84	63.47	90.39	90.31	63.49	65.96
BERT + BiLSTM	Cross Entropy	Protocol-I	90.60	90.53	67.00	61.86	90.68	90.61	67.65	63.73
BERT + BiLSTM + Focal	Focal Loss	Protocol-I	90.30	90.26	65.18	60.10	90.68	90.64	66.52	61.27
BERT + BiLSTM + CW	Cross Entropy + Class Weights (Multiplicative)	Protocol-I	90.20	90.19	62.98	65.13	90.48	90.47	64.26	67.14
BERT + BiLSTM + CW	Cross Entropy + Class Weights (Multiplicative)	Protocol-II	90.34	90.33	63.74	63.98	90.52	90.51	66.25	64.81
BERT + BiLSTM + CRF	CRF	Protocol-I	90.78	90.74	67.21	60.73	90.72	90.66	66.94	62.10
BERT + BiLSTM + CRF + CE	CRF + Cross Entropy (Additive)	Protocol-I	90.15	90.10	62.17	65.39	90.37	90.32	64.14	67.00
BERT + BiLSTM + CRF + CE	CRF + Cross Entropy (Additive)	Protocol-II	90.44	90.40	62.36	63.73	90.83	90.49	63.84	65.81
BERT + BiLSTM + CRF + CW	CRF + Class Weights (Additive)	Protocol-I	90.26	90.20	63.47	63.76	90.68	90.61	65.60	66.78
BERT + BiLSTM + CRF + CW	CRF + Class Weights (Additive)	Protocol-II	90.49	90.45	63.47	65.16	90.68	90.64	65.67	65.83

based on the number of examples in each class, from the equation 13 and 14. The class weights were then applied to Cross Entropy loss function in a multiplicative manner shown in equation 15. By using the class weights, it was possible to give less contributing loss factor for the dominant classes. Therefore, the subservient classes should get more focus while training.

For this experiment, the feature vectors of the last layer of BERT were taken and sent to the BiLSTM network similar to our previous experiments. The BERT + BiLSTM model was trained according to Protocol-I followed by fully connected layers. Softmax function was applied on the outputs of the fully connected layer. As for the objective function, if $\mathbf{y_{ij}}$ is the ground truth probability of j^{th} word in i^{th} sample, $\mathbf{p_{ij}}$ is the probability found and $C_{class_{ij}}$ is the class weight of the class in which the word belongs, then following equation 18, for N samples and M words in each sample the loss function becomes -

$$Loss = -\sum_{i=0}^{N}\sum_{j=0}^{M} C_{class_{ij}} * y_{ij} * log(p_{ij}) \tag{20}$$

In order to indicate the use of Class Weighted Cross Entropy in a model, '+ *CW*' was used in the name of the model.

From Table 6, we can see that Class Weighted BERT + BiLSTM outperformed all of our previous experiments. The Macro F1 score was found to be 65.49% and MUC F1 score to be 70.06%. Since, class weights were introduced to each class for false prediction, our initial imbalanced dataset problem could be tackled. This helped in outperforming models trained on Focal Loss and Cross Entropy Loss without class weights, as both of the losses assign equal weights to all classes. In Figure 6, we can also see that BERT + BiLSTM + CW overall performed better than BERT + BiLSTM. Also in the ROC Curves of BERT + BiLSTM + CW and BERT + BiLSTM in Figure 8, all the other tags, except for I-OBJ and B-OBJ, had experienced an improvement.

We also experimented with fine-tuning the BERT + BiLSTM model with Cost Sensitive Learning following Protocol-II. Keeping our previous experimental setup, we took the best validation epoch weights of BERT + BiLSTM model. Then we fine-tuned the model with Weighted Cross Entropy for 3 epochs. Table 6 shows slight improvement in Macro F1 score and MUC F1 score, from 65.49% and 70.06% to 65.51% and 70.70%. Since BERT + BiLSTM model was fine-tuned on Class Weighted Cross Entropy, the training did not start by randomly initializing weights. Therefore, this performed slightly better than BERT + BiLSTM + CW (Protocol-I) model.

C. INTEGRATION OF CRF LAYER

Our previous experiments depicted that, training BERT BiLSTM with Class Weighted Cross Entropy produced the best contextually meaningful sequences for tag prediction. In many state-of-the-art architectures, Conditional Random Field (CRF) was widely used for sequence labelling tasks. [2], [28], [31], [35], [45], [51].

In order to compare with the state-of-the-art architectures, we experimented by adding a CRF layer in our architecture. Since, BERT + BiLSTM model performed best in all of our previous experiments, we integrated the CRF layer replacing Cross Entropy loss. The feature vectors of the last layer of BERT model were at first passed through the BiLSTM layer. Then, the outputs from the fully connected layer were used as input for the CRF layer. Similar to our previous experiments, the BERT model was trained using Protocol-I. Our objective function for this experiment was the CRF loss described in equation 11. In order to indicate the use of CRF Loss in a model, '+ *CRF*' was used in the name of the model. For predicting the best tags sequence, Viterbi algorithm was used.

From Tables 6 and 9, we can see that addition of CRF layer decreased the accuracy for BERT + BiLSTM model. The Macro F1 score decreased from 65.49% to 63.63%. Despite having a lower Macro F1 score, when inspecting the results we found that the CRF-based model makes correct predictions in some cases where the BERT + BiLSTM model fails. Figure 7 shows such an example.

TABLE 9. Comparison of Results on integrating CRF layer.

Architecture	Result		
	Micro F1	Macro F1	MUC
BERT + BiLSTM + CRF	90.69	63.63	71.44
BERT + BiLSTM + CRF + CE (Protocol-I)	90.35	65.39	71.14
BERT + BiLSTM + CRF + CE (Protocol-II)	90.51	64.69	71.78
BERT + BiLSTM + CRF + CW (Protocol-I)	90.64	**65.96**	**72.04**
BERT + BiLSTM + CRF + CW (Protocol-II)	90.66	65.60	71.77

Following this observation, we performed a set of experiments where a single model is trained to optimize a loss function which is a sum of the CRF loss (Equation 11) and the class weighted cost sensitive cross entropy loss (Equation 20), e.g. summation of the loss functions mentioned in equations 11 and 20. This joint optimization of both loss functions was done using Protocol I, where the model was trained for 20 epochs using this loss function and also using Protocol II, where the model was trained using CRF loss for the first 20 epochs and then fine-tuned on the combined loss function. These are the BERT + BiLSTM + CRF + CW experiments listed in Table 4.

From Table 9, we can see that when trained using Protocol I, the BERT + BiLSTM + CRF + CW model performed better than the previous models. The Macro F1 and MUC F1 score significantly increased from 63.63% and 71.44% to 65.96% and 72.04% respectively. From the Figure 7, we see an example where the BERT + BiLSTM + CRF + CW model was able to predict correctly the tags that BERT + BiLSTM + CRF and BERT + BiLSTM + CW individually were unable to predict. Also in Figure 6, we can see that

		তবে	বর্তমানে	পয়েন্ট	বলতে	ডেস্কটপ	কম্পিউটার	প্রকাশনা	শিল্পে	সংজ্ঞায়িত	জন্য	ডেস্কটপ	পাবলিশিং	পয়েন্ট	-কেই	বোঝায়
Architecture	True Label	O	O	O	O	B-OBJ	I-OBJ	O	O	O	O	O	O	O	O	O
BERT + BiLSTM + CW (Multiplicative)	Predicted Label	**O**	**O**	**O**	**O**	**B-OBJ**	**B-OBJ**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**
BERT + BiLSTM + CRF	Predicted Label	**O**	**O**	**O**	**O**	**O**	**I-OBJ**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**
BERT + BiLSTM + CRF + CW (Additive)	Predicted Label	**O**	**O**	**O**	**O**	**B-OBJ**	**I-OBJ**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**

		উদাহরণস্বরূপ	এবার	বোরো	ধান	সংগ্রহের	জন্য	মূল্য	নির্ধারণ	করা	হয়	৯৬০	টাকা	প্রতি	মণ
Architecture	True Label	O	O	B-OBJ	I-OBJ	O	O	O	O	O	O	O	B-OBJ	O	O
BERT + BiLSTM + CW (Multiplicative)	Predicted Label	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**B-OBJ**	**O**	**O**
BERT + BiLSTM + CRF	Predicted Label	**O**	**O**	**O**	**I-OBJ**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**B-OBJ**	**O**	**O**
BERT + BiLSTM + CRF + CW (Additive)	Predicted Label	**O**	**O**	**B-OBJ**	**I-OBJ**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**B-OBJ**	**O**	**O**

FIGURE 7. **Visualization of tag prediction for model BERT + BiLSTM + CW, BERT + BiLSTM + CRF and BERT + BiLSTM + CRF + CW (Protocol-I).**

BERT + BiLSTM + CRF + CW overall performed better than all of our previous model, especially in tags like B-PER, I-OBJ. Again in the ROC Curves of Figure 8, we can see that, tags like I-OBJ, B-PER experienced an increase in accuracy. Also the other tags were found to be closely accurate with the best model for each tag class. It also produced the best Macro F1 and MUC F1 measures among all of the experiments performed.

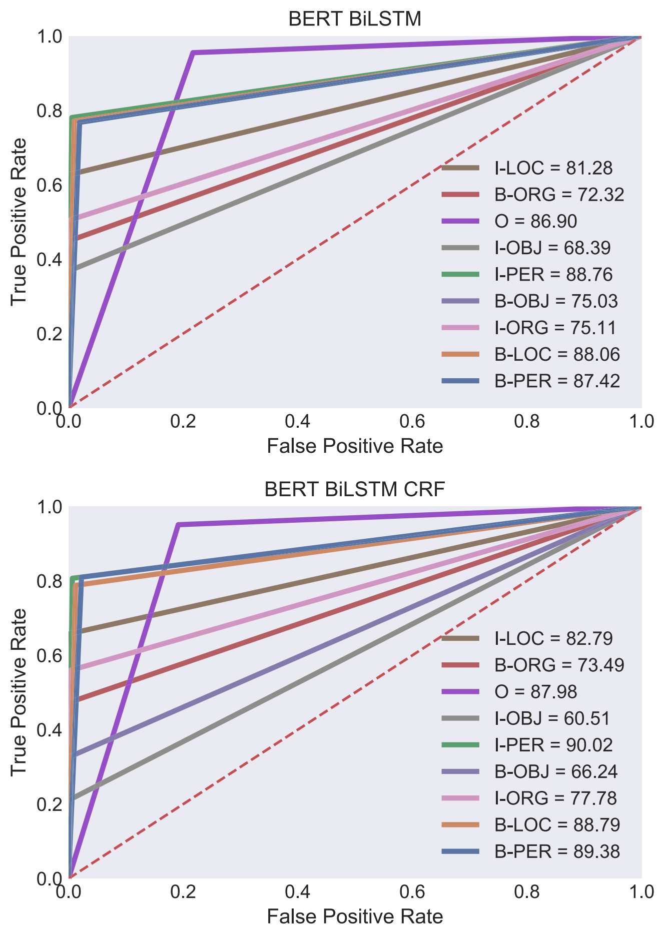
Table 9 also shows the result for BERT + BiLSTM + CRF + CW model trained using Protocol-II. Similar to the previous fine-tuning experiment, we took the best validation epoch weights of BERT + BiLSTM + CRF model. Then we fine-tuned the model along with Weighted Cross Entropy loss for 3 epochs. The model produced slightly lower results than BERT + BiLSTM + CRF + CW which was trained using Protocol-I, but still outperforming the previous models. The model scored 65.60% and 71.77% on Macro F1 and MUC F1 scores.

To understand whether it is categorical cross entropy or the weighted categorical cross entropy that is causing the positive change, we also experimented with adding Cross Entropy (Equation 18) to CRF loss in BERT + BiLSTM + CRF model without Class Weights. These are the BERT + BiLSTM + CRF + CE experiments. The model was trained both in end to end and fine-tuning manner using Protocol-I and Protocol-II. The end to end model scored 65.39% and 71.14% in Macro F1 and MUC F1 scores respectively. On the other hand, by fine-tuning the model scored 64.69% and 71.78% in Macro F1 and MUC F1 scores respectively. We conclude from these results that adding the Weighted Cross Entropy loss was responsible for the overall better Macro F1 score.

To further analyze the efficacy of our proposed cost-sensitive approach we also trained the BERT-based models on two sequence labeling datasets, namely the Microsoft IL-POST Data and MIT Movie Corpus. The training samples in both these data sets have class imbalance. The experiments show that introducing the class weighted cost-sensitive loss function, the overall Macro F1 score was improved. The detailed description and results can be found in Appendix.

D. SUMMARY

From all of our experiments, the best Macro F1 score was produced from BERT + BiLSTM + CRF + CW, trained with Protocol-I. Comparing Word2Vec and BERT models, we found BERT model produced better result which demonstrated the necessity of contextual embeddings for this task. We also compared with the addition of BiLSTM network to the BERT model and found improvement in the F1 score. We then experimented with Focal Loss and Class Weighted Cross Entropy to address the imbalance class issue of the dataset. The BERT + BiLSTM model trained with Class Weighted Cross Entropy yielded better results of the two. Following that, we added CRF layer to BERT + BiLSTM model in order to compare our BERT + BiLSTM + CW



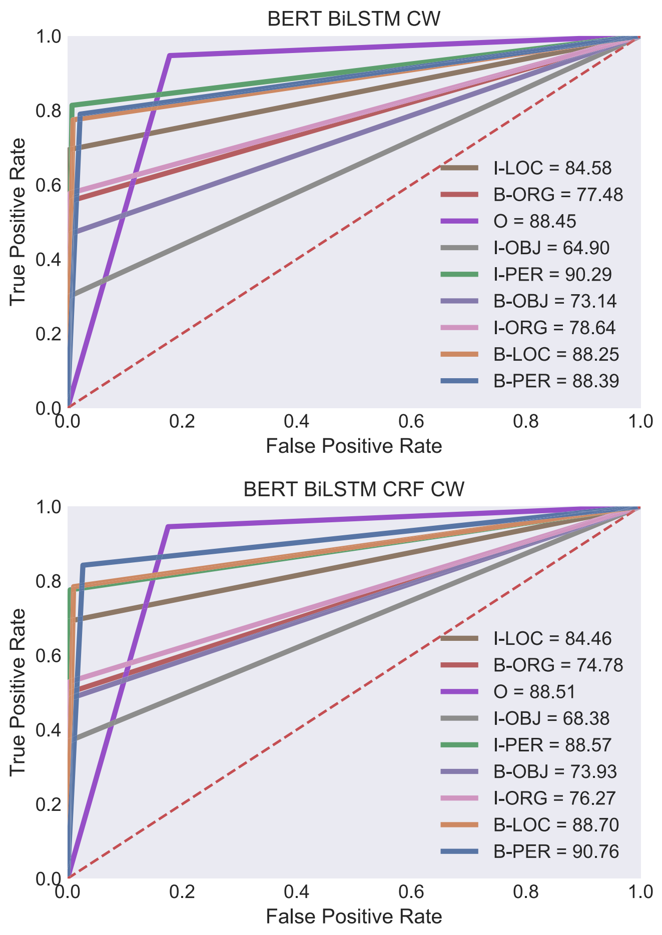


FIGURE 8. ROC curves of the best models in every stage of the experiments.

model. The addition of CRF layer resulted in decrease of Macro F1 score but showed promise in certain contexts. As such, the CRF layer was added to the BERT + BiLSTM + CW model. The model produced the best results when trained according to Protocol-I.

From the ROC curves of Figure 8, it can be seen that for some of the classes the addition of Class Weight to the baseline model (BERT + BiLSTM) improves the result. A significant improvement over almost all the class, specifically, in the subservient classes, can be seen due to the integration of Class Weighted Cross Entropy with BERT + BiLSTM + CRF. This indicated that the use of Cost Sensitive Learning ensures overall better learning for each individual class feature.

Again, introduction of CRF loss to BERT + BiLSTM model helped the model in gaining overall better result than BERT + BiLSTM alone. However, it was not able to produce overall better distribution than BERT + BiLSTM + CW, for some of the classes, such as - B-ORG, I-OBJ, I-PER, B-OBJ, it was not able to perform well. Hence, Class Weighted Cross Entropy was added with CRF loss in order to get the final model with the best Macro F1 and MUC F1 measure and overall well distribution of class features learnt.

VII. COMPARISON WITH OTHER MODELS

As discussed in Section II, there are a few previous studies which report results on Bangla NER. But we were not able to fairly compare the performance of our model with their approaches as their reported results are on different datasets, which we were not able to obtain. Moreover, we could not reproduce their architectures faithfully on our datasets because the architecture resources, gazetteers and word embeddings are not available openly.

However the one we can compare against is the work of Karim *et al.* [26] as we are reporting on the same dataset and have used the exact same train-validation-test split of data. Moreover, as far as we could ascertain, their proposed architecture DCN-BiLSTM is the most recent work in Bangla NER. Using the dataset they introduced, they obtained a best result of 63.37% MUC F1. They however do not report their Macro F1 results. Table 10 compares our best performing model with the ones reported in Karim *et al.* [26].

In terms of MUC F1, our proposed BERT + BiLSTM + CRF + CW model, which uses BERT embeddings processed through a BiLSTM layer followed by a CRF layer and which was optimized by a loss function consisting of a CRF loss

TABLE 10. Performance comparison with Karim *et al.* [26].

Architecture	MUC F1
CNN-BiLSTM (Karim *et al.*)	61.41
CNN-BiLSTM-CRF (Karim *et al.*)	61.07
DCN-BiLSTM (Karim *et al.*)	63.37
DCN-BiLSTM-CRF (Karim *et al.*)	63.28
BERT+BiLSTM+CRF+CW (Our Model)	**72.04**

added with our simple class weighted cross entropy loss, obtained an improvement of over 8% in terms of MUC F1. This is a significant improvement.

VIII. CONCLUSION

Bangla is a morphologically rich language with a large population base. But it has very few, if any, large scale annotated datasets for NLP research. Recent state-of-the-art architectures are seen to use pre-trained language models in order to gain embeddings which do not require large scale annotated dataset for training. These generic models have been found to be effective in training specific tasks like Named Entity Recognition. These architectures are also seen to particularly emphasize on Language Models that produce contextual embeddings, so that the model may understand the context of each word in a sentence, as explained in I. In this paper, we used BERT multi-lingual model which is trained on limited Bangla vocabulary of word pieces. We performed the NER task on the recently introduced dataset by Karim *et al.* [26]. Using the dataset, we experimented on 11 different model configurations, 10 of which were based on the BERT multi-lingual model. In our first set of experiments, although we found that BERT + BiLSTM model performed better than BERT model alone, the model suffered from the issue of data imbalance as shown in Figures 6 and 8.

In order to address the data imbalance issue, we used our proposed statistical approach of Cost Sensitive Learning and Focal Loss. The Focal Loss is a well established method of coping with class imbalanced data. However, for our imbalanced dataset, the proposed statistical approach yielded better results compared to Focal Loss. We further experimented with CRF layer in order to compare the BERT + BiLSTM + CW model with the present state-of-the-art architectures. Although introduction of CRF layer exacerbated the Macro F1 score in comparison to the BERT + BiLSTM + CW, closer inspection in Figures 7, 6 and 8 reveal that BERT + BiLSTM + CRF model was able to predict certain instances more accurately compared to the BERT + BiLSTM + CW model. Following this observation, we trained a model which was optimized jointly on the CRF loss and the class weighted cross entropy loss, which produced the best results out of all of our models.

For future works, we aim to take up the following tasks -

- Perform experimentation with language models other than BERT, e.g. ELMO, ALBERT etc., in order to compare the results across all those Language Models.
- Analyze our statistical approach with other approaches for defining class weights for Cost Sensitive Learning.
- Introduce cost sensitive learning to CRF layer, potentially using techniques similar to the one proposed by Lannoy *et al.* in [14]
- Explore different methodologies to address issues like Multiword Expressions, Proverbs and Phrases.

APPENDIX
ADDITIONAL RESULTS
A. DATASETS

Our experiments on different architectures show that, using cost-sensitive learning by the statistical approach described in Section V-H helps in addressing the imbalance of our dataset and increase the F1 Macro scores. Since the experiments carried out in this paper were on a single dataset, we further analyze the effectiveness of the statistical approach by experimenting on sequence labeling tasks from two additional datasets. The following sections describe the datasets and obtained results.

1) MICROSOFT IL-POST DATA

The Microsoft IL-POST Data[1] is designed for the task of Bangla Part-of-Speech-Tagging. It is based on the IL-POST framework which is a POS-tagset framework for Indian Languages. It covers the morph-syntactic details of Indian Languages which includes Bangla. The dataset contains manually annotated 7168 sentences and 31 labels.

Table 11 describes the dataset information and train-test split. 10% of the original dataset were taken as test set and 90% were taken as training set. Table 12 illustrates the imbalance of the dataset by showing the label distributions, which makes it eligible for our cost-sensitive experiments. Two majority and minority classes are also highlighted in the table.

TABLE 11. Microsoft IL-POST Data Details.

Dataset Type	POS Tagging
Number of Total Sentences	7168
Number of Total Labels	31
Number of Training Sentences	6451
Number of Test Sentences	717

2) MIT MOVIE CORPUS

The MIT Movie Corpus[2] is a Named Entity Recognition dataset based on the comments related to various movies. There are two types of data - *eng* and *trivia10k13* corpus. The *eng* corpus is based on simple queries whereas the *trivia10k13* corpus contains more complex sentences. The datasets are annotated in BIO format. For our experiments, we used *trivia10k13* corpus. There are 9769 sentences and 25 labels (12 unique labels) present in the corpus.

[1]https://catalog.ldc.upenn.edu/LDC2010T16

[2]https://groups.csail.mit.edu/sls/downloads/movie/

TABLE 12. Distribution of tags for Microsoft IL-POST Data.

Tag	Count	Tag	Count	Tag	Count
JJ	8332	**NC**	**29664**	**PU**	**13438**
CCD	2385	NP	6946	VM	11001
JQ	3935	PRL	341	CX	1660
DAB	1806	PPR	4604	CSB	1530
PP	2978	NV	1971	CCL	222
AMN	1488	RDS	1302	VAUX	1977
NST	1621	ALC	1466	PWH	345
RDF	1317	PRF	261	**PRC**	**15**
LC	498	DRL	300	**LV**	**72**
DWH	54	CIN	59	RDX	397
VA	238				

TABLE 13. MIT Movie Corpus Details.

Dataset Type	NER
Number of Total Sentences	9769
Number of Total Labels	25
Number of Training Sentences	7816
Number of Test Sentences	1953

TABLE 14. Distribution of tags for MIT Movie Corpus.

Tag	Count	Tag	Count	Tag	Count
B-Actor	5010	I-Actor	6121	**O**	**55895**
B-Plot	6468	**I-Plot**	**62107**	B-Opinion	810
I-Opinion	539	B-Award	309	I-Award	719
B-Year	2702	B-Genre	3384	B-Origin	3340
B-Director	1787	I-Director	1653	I-Genre	2283
I-Year	195	**B-Sound**	**50**	I-Sound	158
B-Relation	580	I-Relation	1206	B-Char	1025
I-Char	790	**B-Quote**	**126**	I-Quote	817
I-Origin	3340				

Table 13 describes the MIT Movie dataset information and train-test split. We used the official test dataset which is 20% of the original dataset. Table 14 illustrates the label distribution which indicates that this dataset is also imbalanced towards some labels. Two majority and minority classes are also highlighted in the table.

B. MODELS

We performed experiments on the BERT based models with the same hyperparameters and architectures which were originally carried out for the Bangla NER dataset. The only difference here is the number of labels. Table 15 provide a list of the architectures. Initially we fine tune BERT with BiLSTM without cost sensitive learning (Architecture 1). We then apply our cost-sensitive learning approach using Protocol-I and Protocol-II (Architecture 2,3). In case of experimenting with CRF based models, we add CRF with BERT and BiLSTM without cost-sensitive learning (Architecture 4). Following that, we experiment BERT-BiLSTM and CRF with cost-sensitive learning using Protocol-I and Protocol-II (Architecture 5,6). We also carry out the fine tuning with only Cross Entropy Loss (without cost-sensitive penalizing) to further analyze the effect of our proposed approach (Architecture 7,8). Details of the models are described in Section V.

TABLE 15. Results obtained from the different BERT-Based model demonstrating the effect of the class weighted cross entropy loss.

	Architecture	Microsoft IL-POST Result		MIT Movie Result	
		Micro F1	Macro F1	Micro F1	Macro F1
1	BERT + BiLSTM	86.67	78.00	87.82	73.29
2	BERT + BiLSTM + CW (Protocol-I)	86.21	78.54	87.93	**73.87**
3	BERT + BiLSTM + CW (Protocol-II)	86.97	**78.95**	87.91	73.81
4	BERT + BiLSTM + CRF	86.83	78.32	87.71	72.41
5	BERT + BiLSTM + CRF + CE (Protocol-I)	86.66	78.45	87.69	72.18
6	BERT + BiLSTM + CRF + C (Protocol-II)	86.45	78.55	87.66	71.51
7	BERT + BiLSTM + CRF + CW(Protocol-I)	86.74	**79.84**	87.54	**74.50**
8	BERT + BiLSTM + CRF + CW (Protocol-II)	86.76	79.03	88.00	73.18

C. RESULTS

Table 15 provides the list of obtained results on the different BERT based models on the two datasets. The results demonstrate that for the incorporation of the class weighted cost sensitive learning, using either Protocol I or Protocol II, helps to address the class imbalance problem. This is demonstrated across the two datasets used in this appendix and mirrors the findings on the Bangla NER dataset shown in Table 7. The introduction of cost-sensitive learning increased the overall F1 Macro scores significantly for both non CRF and CRF based architectures. Even though we proposed the cost-sensitive learning approach for the NER task, it performs similarly well for the POS Tagging and NER tasks with more labels. Therefore, our statistical approach for cost-sensitive learning is effective in addressing class imbalance issues in different data sets and thus verifying of its efficacy.

ACKNOWLEDGMENT

(Imranul Ashrafi and Muntasir Mohammad contributed equally to this work.)

REFERENCES

[1] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf, "FLAIR: An easy-to-use framework for state-of-the-art NLP," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics (Demonstrations)*, 2019, pp. 54–59.

[2] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 1638–1649.

[3] A. Baevski, S. Edunov, Y. Liu, L. Zettlemoyer, and M. Auli, "Cloze-driven pretraining of self-attention networks," 2019, *arXiv:1903.07785*. [Online]. Available: http://arxiv.org/abs/1903.07785

[4] S. Banerjee, S. K. Naskar, and S. Bandyopadhyay, "Bengali named entity recognition using margin infused relaxed algorithm," in *Proc. Int. Conf. Text, Speech, Dialogue*. Cham, Switzerland: Springer, 2014, pp. 125–132.

[5] N. Banik and M. H. H. Rahman, "GRU based named entity recognition system for bangla online newspapers," in *Proc. Int. Conf. Innov. Eng. Technol. (ICIET)*, Dec. 2018, pp. 1–6.

[6] I. Baris, L. Schmelzeisen, and S. Staab, "CLEARumor at SemEval-2019 task 7: ConvoLving ELMo against rumors," 2019, *arXiv:1904.03084*. [Online]. Available: http://arxiv.org/abs/1904.03084

[7] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," 2019, *arXiv:1903.10676*. [Online]. Available: http://arxiv.org/abs/1903.10676

[8] L. Ceriani and P. Verme, "The origins of the Gini index: Extracts from Variabilità e Mutabilità (1912) by Corrado Gini," *J. Econ. Inequality*, vol. 10, no. 3, pp. 421–443, 2012.

[9] B. B. Chaudhuri and S. Bhattacharya, "An experiment on automatic detection of named entities in Bangla," in *Proc. Workshop Named Entity Recognit. South South East Asian Lang. (IJCNLP)*, 2008, pp. 75–82.

[10] H. L. Chieu and H. T. Ng, "Named entity recognition: A maximum entropy approach using global information," in *Proc. 19th Int. Conf. Comput. Linguistics*, vol. 1, 2002, pp. 1–7.

[11] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 357–370, Dec. 2016.

[12] S. A. Chowdhury, F. Alam, and N. Khan, "Towards Bangla named entity recognition," in *Proc. 21st Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2018, pp. 1–7.

[13] K. Crammer and Y. Singer, "Ultraconservative online algorithms for multiclass problems," in *Proc. Int. Conf. Comput. Learn. Theory*. Berlin, Germany: Springer, 2001, pp. 99–115.

[14] G. de Lannoy, D. Francois, J. Delbeke, and M. Verleysen, "Weighted conditional random fields for supervised interpatient heartbeat classification," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 1, pp. 241–247, Jan. 2011.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2019, pp. 4171–4186.

[16] A. Ekbal and S. Bandyopadhyay, "Bengali named entity recognition using classifier combination," in *Proc. 7th Int. Conf. Adv. Pattern Recognit.*, Feb. 2009, pp. 259–262.

[17] A. Ekbal and S. Bandyopadhyay, "Named entity recognition in Bengali: A multi-engine approach," *Northern Eur. J. Lang. Technol.*, vol. 1, pp. 26–58, Feb. 2009.

[18] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang, "Named entity recognition through classifier combination," in *Proc. 7th Conf. Natural Lang. Learn. HLT-NAACL*, vol. 4, 2003, pp. 168–171.

[19] F. Godin, B. Vandersmissen, W. D. Neve, and R. V. D. Walle, "Multimedia Lab@ ACL WNUT NER shared task: Named entity recognition for Twitter microposts using distributed word representations," in *Proc. Workshop Noisy User-Generated Text*, 2015, pp. 146–153.

[20] M. F. A. Hady, A. Karali, E. Kamal, and R. Ibrahim, "Unsupervised active learning of CRF model for cross-lingual named entity recognition," in *Proc. IAPR Workshop Artif. Neural Netw. Pattern Recognit.* Cham, Switzerland: Springer, 2014, pp. 23–34.

[21] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*. [Online]. Available: http://arxiv.org/abs/1606.08415

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," 2018, *arXiv:1801.06146*. [Online]. Available: http://arxiv.org/abs/1801.06146

[24] N. Ibtehaz and A. Satter, "A partial string matching approach for named entity recognition in unstructured Bengali data," *Int. J. Mod. Edu. Comput. Sci.*, vol. 10, no. 1, pp. 36–45, Jan. 2018.

[25] C. Jia, X. Liang, and Y. Zhang, "Cross-domain NER using Cross-domain language modeling," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2464–2474.

[26] R. Karim, M. Islam, S. R. Simanto, S. A. Chowdhury, K. Roy, A. A. Neon, M. Hasan, A. Firoze, and R. M. Rahman, "A step towards information extraction: Named entity recognition in Bangla using deep learning," *J. Intell. Fuzzy Syst.*, to be published.

[27] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th Int. Conf. Mach. Learn. (ICML)*. San Francisco, CA, USA: Morgan Kaufmann, 2001, pp. 282–289.

[28] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 260–270.

[29] C. Lee, Y.-B. Kim, D. Lee, and H. Lim, "Character-level feature extraction with densely connected networks," 2018, *arXiv:1806.09089*. [Online]. Available: http://arxiv.org/abs/1806.09089

[30] F. Li, X. Zhang, X. Zhang, C. Du, Y. Xu, and Y.-C. Tian, "Cost-sensitive and hybrid-attribute measure multi-decision tree over imbalanced data sets," *Inf. Sci.*, vol. 422, pp. 242–256, Jan. 2018.

[31] X. Li, L. Bing, W. Zhang, and W. Lam, "Exploiting BERT for end-to-end aspect-based sentiment analysis," 2019, *arXiv:1910.00883*. [Online]. Available: http://arxiv.org/abs/1910.00883

[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[33] M. Liu, C. Xu, Y. Luo, C. Xu, Y. Wen, and D. Tao, "Cost-sensitive feature selection by optimizing F-Measures," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1323–1335, Mar. 2017.

[34] T. Liu, J.-G. Yao, and C.-Y. Lin, "Towards improving neural named entity recognition with gazetteers," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 5301–5307.

[35] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016, pp. 1064–1074.

[36] A. Mikheev, M. Moens, and C. Grover, "Named entity recognition without gazetteers," in *Proc. 9th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 1999, pp. 1–8.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[38] M. Oleynik, P. Nohama, P. Cancian, and S. Schulz, "Performance analysis of a POS tagger applied to discharge summaries in Portuguese," *Stud. Health Technol. Informat.*, vol. 160, no. 2, pp. 959–963, 2010.

[39] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. (2018). *Improving Language Understanding by Generative Pre-Training*. [Online]. Available: https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageUnderstand.paper.pdf

[40] S. Reshmi and K. Balakrishnan, "Enhancing Inquisitiveness of Chatbots Through NER Integration," in *Proc. Int. Conf. Data Sci. Eng. (ICDSE)*, Aug. 2018, pp. 1–5.

[41] C. N. dos Santos and V. Guimaraes, "Boosting named entity recognition with neural character embeddings," 2015, *arXiv:1505.05008*. [Online]. Available: http://arxiv.org/abs/1505.05008

[42] A. Senapati, A. Das, and U. Garain, "Named-entity recognition in Bengali," in *Proc. Post 4th 5th Workshops Forum Inf. Retr. Eval.*, 2013, p. 14.

[43] S. Sharma and R. Daniel, Jr., "BioFLAIR: Pretrained pooled contextualized embeddings for biomedical sequence labeling tasks," 2019, *arXiv:1908.05760*. [Online]. Available: http://arxiv.org/abs/1908.05760

[44] M. Skeppstedt, "Enhancing medical named entity recognition with features derived from unsupervised methods," in *Proc. Student Res. Workshop 14th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2014, pp. 21–30.

[45] F. Souza, R. Nogueira, and R. Lotufo, "Portuguese named entity recognition using BERT-CRF," 2019, *arXiv:1909.10649*. [Online]. Available: http://arxiv.org/abs/1909.10649

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[47] M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman, "A model-theoretic coreference scoring scheme," in *Proc. 6th Conf. Message Understand. (MUC)*, 1995, pp. 45–52.

[48] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "HuggingFace's transformers: State-of-the-art natural language processing," 2019, *arXiv:1910.03771*. [Online]. Available: http://arxiv.org/abs/1910.03771

[49] K. Xue, Y. Zhou, Z. Ma, T. Ruan, H. Zhang, and P. He, "Fine-tuning BERT for joint entity and relation extraction in chinese medical text," 2019, *arXiv:1908.07721*. [Online]. Available: http://arxiv.org/abs/1908.07721

[50] W. Yoon, C. H. So, J. Lee, and J. Kang, "CollaboNet: Collaboration of deep neural networks for biomedical named entity recognition," *BMC Bioinf.*, vol. 20, no. S10, p. 249, May 2019.

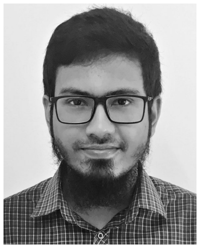
[51] A. Liu, Z. Huang, H. Lu, X. Wang, and C. Yuan, "BB-KBQA: BERT-based knowledge base question answering," in *Proc. Chin. Comput. Linguistics, 18th China Nat. Conf. (CCL)*, Kunming, China, Oct. 2019, p. 81.

[52] C. Zhang, K. C. Tan, H. Li, and G. S. Hong, "A cost-sensitive deep belief network for imbalanced classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 109–122, Jan. 2018.

[53] C. Zhang, K. C. Tan, and R. Ren, "Training cost-sensitive deep belief networks on imbalance data problems," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 4362–4367.

[54] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[55] G. Zhou and J. Su, "Named entity recognition using an HMM-based chunk tagger," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2002, pp. 473–480.

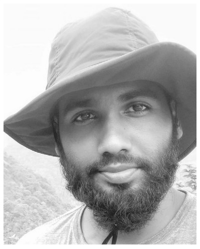


IMRANUL ASHRAFI is currently pursuing the B.Sc. degree in computer science and engineering (CSE) with the Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. His current research interests include computer vision applications, natural language processing, pruning methodologies, and digital security.

MUNTASIR MOHAMMAD was born in Bangladesh, in 1996. He received the B.Sc. degree in computer science and engineering from the Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. His current research interests include deep learning, machine learning, natural language processing, and computer vision.

ARANI SHAWKAT MAUREE is currently pursuing the B.Sc. degree in computer science and engineering (CSE) with the Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. Her current research interests include deep learning, natural language processing, machine learning, computer vision, and data analysis.

GALIB MD. AZRAF NIJHUM is currently pursuing the B.Sc. degree in computer science and engineering (CSE) with the Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. His current research interests include deep learning and natural language processing and analysis with satellite imagery.



REDWANUL KARIM was born in Bangladesh in 1995. He received the B.Sc. degree in computer science and engineering from the Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. His current research interests include deep learning, natural language processing, information extraction, and bioinformatics.

NABEEL MOHAMMED received the bachelor's degree in computer science from Monash University, Australia. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, North South University. He worked as a Software Developer for Editure Ltd., a Melbourne based Software Firm specializing in providing software solutions for K-12 schools. After three and half years in that role he moved back into academia to complete the Ph.D. degree at Monash University, Australia. His current research interests lie in the area of computer vision and natural language processing, particularly in areas pertaining to their application for the Bangla language.

SIFAT MOMEN received the M.Sc. (Hons.) degree in electronics and information technology from Sheffield Hallam University, U.K., and the Ph.D. degree from The University of Sheffield, U.K., in 2011. He joined the Department of Electrical and Computer Engineering, North South University (NSU), in September 2017, where he is currently an Associate Professor. Prior to this, he worked as an Assistant Professor for nearly six years with the Department of Computer Science and Engineering, University of Liberal Arts Bangladesh (ULAB). He also served as the Head of the Department, ULAB, for some time. His Ph.D. work was at the interface of biology and engineering and was strongly inspired by the behavior of social insects (e.g., ants and bees) that are well-known to exhibit self-organizing behavior. His current research interests include artificial intelligence, particularly that of natural language processing, machine learning, modeling and simulation of natural systems, swarm intelligence, swarm robotics and artificial life. He is an active researcher and regularly reviews numerous conference papers and journal articles.