VOL 8(1), 93-105
Copyright © Madda Walabu University
ISSN: 2523-1901 (Online)
URL: http://jessd.mwu.edu.et/
Email: JESSDeditorials@mwu.edu.et
Journal of Equity in Sciences & Sustainable Development

DOI: https: //doi.org/10.20372/mwu.jessd.2025.15671

Full Length Research Paper

Building a Named Entity Recognition model for Ethiopian Languages: a comparative analysis of composite feature embedding

Sintayehu Hirpassa (PhD)

Assistant Professor, Department of Computer Science, Adama Science and Technology University, Ethiopia.

Article Info

Article History
Received: 10 Dec 2024
Accepted: 27 Jan 2025

Keywords:
Named Entity Recognition, Amharic, conditional random field, Recurrent neural network, Long short term, Convolutional neural network

Abstract

In this study, we propose a deep learning NER model that effectively represents word tokens through a combinatorial feature embedding design. We conducted a comparative analysis with existing models for Ethiopian languages. The word vectors created for all tokens using an unsupervised learning algorithm are merged with a set of language-independent features specifically developed for this purpose. These combined features are then fed into a neural network model to predict word classes. Empirical results obtained from the Ethiopian language dataset demonstrate that incorporating character-level word embeddings along with other features in BiLSTM-CRF models yields state-of-the-art performance. In addition to showing the model's ability to generalize to different languages, we evaluated its performance and achieved remarkable accuracy rates: 92.88% and 82.35% on the AM_NER and Oro_NER datasets, respectively.

* *Corresponding* sintsha2002@gmail.com

1. Introduction

Named Entity Recognition (NER) is an essential task in natural language processing. It involves identifying names of persons, places, organizations, and other entities in a document (Li et al., 2022). NER is a critical step in machine translation, information extraction, and question-answering systems. In the past, hidden Markov models (HMM) or conditional random fields (CRF) models were commonly used for NER. However, with the advancements in deep learning, researchers have applied deep neural networks (DNNs) to achieve state-of-the-art results. Collobert and Weston (2008) were pioneers in using DNN models for NER. They designed an architecture based on convolutional neural networks (CNN) for token sequences. Subsequently, many scholars have made various improvements to this architecture. Huang et al. (2018) suggested replacing the CNN used in Collobert and Weston (2008) with a BiLSTM encoder. Chiu and Nichols (2016) and Lample et al. (2016) introduced a hierarchy in the architecture by replacing manually extracted character-level features with BiLSTM and CNN models, respectively.

Recent methods for NER incorporate various techniques, including (1) semantic information using word embeddings like Word2Vec and GloVe, (2) character-level features of named entities, (3) word order via BiLSTM, and (4) combining LSTM with a CRF at the output layer. In the embedding layer of the BiLSTM model, character-level embedding vectors are used alongside word-level embeddings to handle unseen words in the trained vocabulary set (Al-Rfou et al., 2013). The combination of word embedding and character-level word embedding has been explored by Ma and Hovy (2016) using CNNs, while Lample et al. (2016) employed LSTM networks. Additionally, Reimers and Gurevych (2019) analyzed and improved the performance of NER tools on general datasets, including CoNLL (2003), using character-level word embeddings. Furthermore, some studies have applied a BiLSTM-CRF model with LSTM-based character-level word embeddings to the NER task, achieving state-of-the-art performance that surpasses traditional feature-based models (Chiu & Nichols, 2016; Lample et al., 2016).

It is widely acknowledged that NER systems are language and domain-dependent. While state-of-the-art NER systems for English achieve near-human level performance, the error rates reported for Amharic and Afan Oromo are still relatively high. This discrepancy is due to the lack of resources, such as corpora, parsers, and lexicons, for developing NER systems in these languages. Amharic and Afan Oromo have a large number of speakers, yet very few NLP resources have been developed for them, and little effort has been made to provide useful higher-level computer-based tools, including NER systems, for speakers of these languages. Moreover, the morphological complexity of Ethiopian languages like Amharic and Afan Oromo further limits the availability of NLP resources and research in these languages.

In this paper, we propose and compare models that utilize different types of word embeddings and character-level encodings to generate empirical results for NER tasks in Ethiopian languages, particularly Amharic and Afan Oromo. The main contributions of our proposed model can be summarized as follows:

- We design a technically simple architecture that employs combinatorial feature embedding to accurately identify named entities.
  - We develop an effective word representation by leveraging character-level CNN and Bi-LSTM, which effectively capture comprehensive information about a word token.
  - Through comparison with state-of-the-art methods on newly created datasets, we demonstrate the empirical strength of our work.
  - We analyze the impact of each feature and the effectiveness of their combination.

The subsequent sections of this paper are organized as follows: Section 2 discusses related work, Section 3 introduces character-level embedding and methodology, Section 4 describes the proposed model, Section 5 presents experimental results using our system, and finally, Section 6 concludes the work and suggests future research directions.

2. Ethiopian Named Entity Recognition Models

The slow pace of development in language tools and resources, particularly Named Entity Recognition (NER) for Ethiopian languages, has hindered progress in overcoming language barriers and providing improved access to information and services. Only a limited number of works have been reported in NER for Amharic and Afan Oromo, the two major languages in Ethiopia. These works have been reviewed based on the models/algorithms used, the feature sets employed, the corpora utilized, and the experimental results achieved.

The first Amharic NER system, developed by Moges, employed a Conditional Random Field (CRF) framework. It investigated the impact of various language-independent features such as contextual words, prefixes, suffixes, beginning and end of a sentence, as well as language-dependent features including orthographic features

and Part-of-Speech (POS) tags. Besufekad implemented a similar approach with some different features, including context words, suffixes, prefixes, named entity tags, previous and next named entity tags, and POS tags. Mikiyas explored a hybrid approach combining rule-based and statistical methods, incorporating all the features used by Moges and Besufekad.

In a recent work by Gamback and Sikdar, a Deep Neural Network (DNN) approach using a Bidirectional Long Short-Term Memory (BiLSTM) model was applied for Amharic NER. They considered the semantic information of each token using word2vec embeddings and incorporated language-independent features, merging the word vectors. Kejela proposed and implemented the first NER system for Afan Oromo, combining rule-based and statistical approaches. A new feature introduced in this work was word shape, considering capitalization patterns specific to Afan Oromo named entities.

The performance of the reviewed works was evaluated using Precision, Recall, and F-measure. Moges achieved an F-measure of 74.61% using 10,340 manually annotated tokens from the ELRC corpus. The next NER system achieved a higher performance with an F1 score of 80.66% using 13,538 tokens from the ELRC corpus. It was concluded that the CRF model is a state-of-the-art machine learning model for NER. However, Besufekad argued that the POS tag feature does not significantly impact the performance of the Amharic NER system, although it is widely used by state-of-the-art NER systems for achieving excellent results. Mikiyas emphasized that relying solely on the features used by Besufekad is insufficient and proposed the inclusion of a nominal flag indicating noun words in the feature set, leading to an F-measure of 85.9%.

In another work by Gamback and Sikdar, the Amharic NER system achieved a 69.7% F1-score by class using the SAY corpus, which consisted of 4,237 sentences and 109,676 tokens. Kejela developed an NER system for Afan Oromo and achieved an average F1-measure of 76.60% using a newly prepared corpus containing 23,000 words.

In conclusion, all the researchers highlighted the limited availability of training data as a significant challenge, resulting in their systems not achieving the performance levels of state-of-the-art NER systems in English and other languages. The scarcity of resources and research in Ethiopian languages has impeded the progress of NER development, but efforts are being made to address these limitations and improve the performance of language tools and resources for Ethiopian languages.

3. Proposed model

In this section, we will provide a step-by-step description of how our model was constructed, including its subnetworks and the reasons for their utilization. Our paper introduces a novel system designed to effectively process the Ethiopian language. We have adopted the BiLSTM-CRF architecture, which is a network known for its ability to capture comprehensive contextual information in a bidirectional token sequence. To leverage the computational efficiency of CNN-based LSTM for character-level encoding, we have incorporated a CNN and proposed a lightweight architecture called CNN-BiLSTM-CRF for our specific problem.

In line with the findings of Luo et al. (2018), we have also considered additional features, such as syntactic information in the form of the Part-of-Speech (POS) tag associated with each token. Therefore, our model utilizes three representations: word-level, character-level, and POS-level LSTM. These representations are then combined to create a fused vector that captures rich semantic and grammatical aspects of the input sentence. Equation 1 demonstrates the process of combining all the embedding vectors into a single word vector, denoted as x_t. This x_t is the output of a fully connected network with a vector size of 200. Subsequently, the input word vector x_t is passed through the fully connected Bi-LSTM

network, which incorporates weight matrix W_i and bias vector, as indicated in Equation 2.

$$Xt = [Vc, V_C, V_P] \quad \text{Eq.1}$$

$$Wt = f(Wi * Xt + b) \quad \text{Eq.2}$$

Where,

Xt is the network output,

V_W is the word vector,

V_C is the character vector,

V_P is PoS vector

Wi weight matrix,

Wt total weight

After passing through the fully connected network, the resulting output serves as the final representation for each input token. In our architecture, we have replaced the Softmax layer with a CRF layer. This choice allows us to leverage the implicit constraints present in the sequence of tags and enhance the performance of our model. As Huang et al. (2018) discussed, the BiLSTM-CRF architecture is capable of capturing the dependencies between the labels in the output layer and the dependencies of the input features.

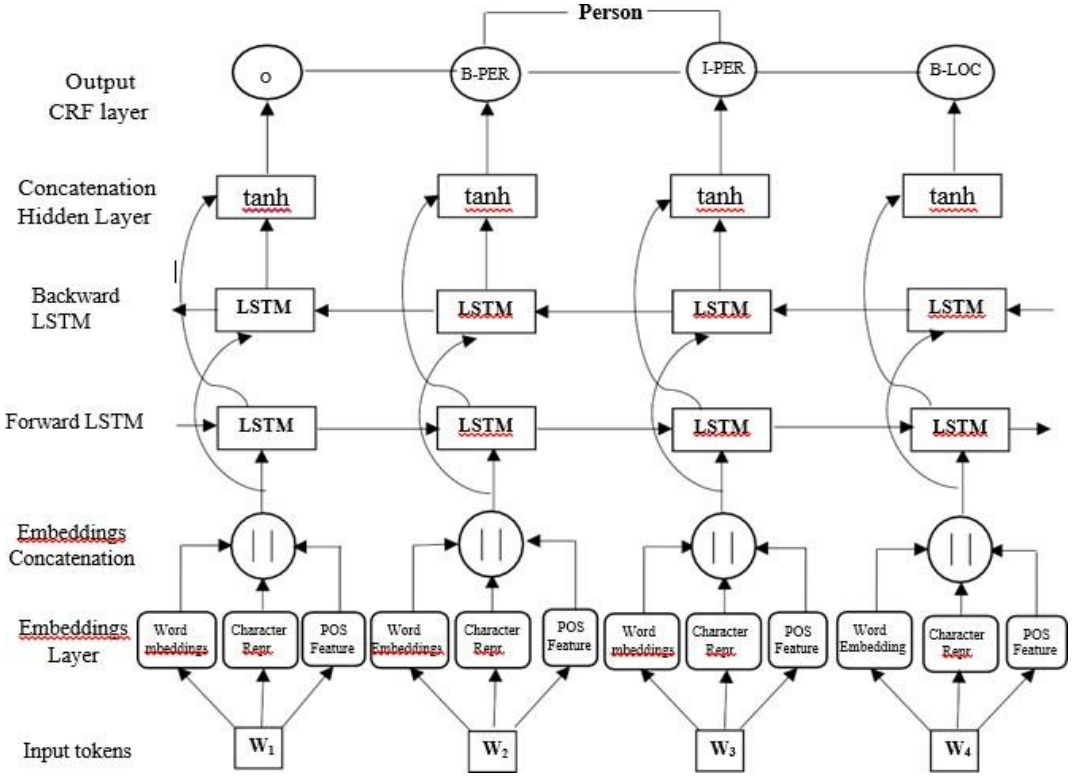

Figure 1: The conceptual architecture of the NER using BiLSTM-CRF network models with character-level word representation and additional features (Adapted from Reimersand Gurevych (2017).

The scores for each output label are calculated by the hidden layer. For instance, for the input "wi", the node outputs the following scores: 1.5 (B-PER), 0.9 (I-PER), 0.1 (O), 0.08 (B-ORG), and

0.05 (B-LOC), as shown in detail in Figure 2. Instead of independently predicting labels using Softmax, the CRF layer is utilized to determine the most accurate label path among all possible label paths. The predicted values from the hidden layer are then passed as input to the CRF layer. The CRF layer identifies the label with the highest prediction score in the label sequence as the best answer. To ensure the validity of the final predicted labels, the CRF layer may impose constraints, which are automatically learned during the training process based on annotated samples.

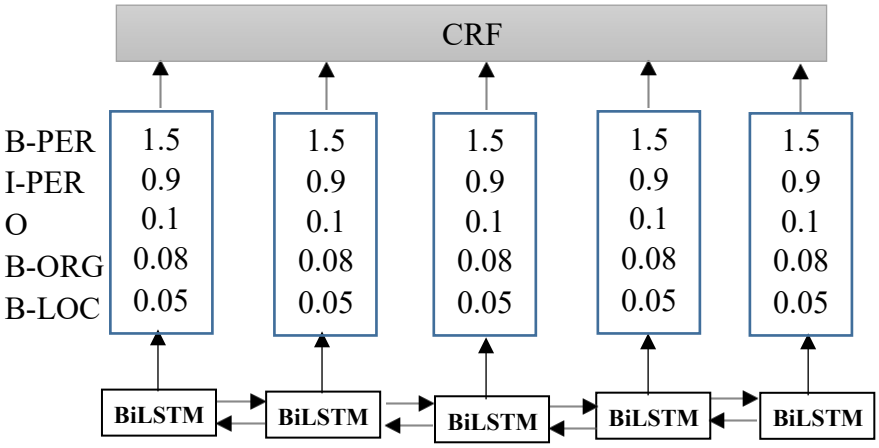

Figure 2: illustrates the integration of BiLSTM and CRF to enhance named entity recognition performance.

On the other hand, if we omit the use of the CRF layer, the hidden states would be independently fed into a Softmax classifier to predict the labels, and there would be little likelihood of imposing constraints. In this case, the label score is calculated for each word, and the label with the highest score among all labels is chosen as the output label. For instance, in the aforementioned scenario, for the word "w1," "B-PER" has the highest score, so it can be selected as the best predicted label. Similarly, "I-PER" can be chosen for "w1," "O" for "w2," "B-ORG" for "w3," and "O" for "w4." Although this approach might yield accurate labels for a sentence "x" in some cases, it is not always reliable. When strong dependencies exist among output labels, independent classification may fail as it does not consider the influence of previous labels on the decision-making process. Generally, for sequence labeling tasks, it is valuable to consider the relationships between labels in neighboring positions and jointly decode the optimal sequence of labels for the input sentence.

3.1 Character-Level Encoder

During both the training and testing stages, there may be entities whose words are not present in the word vocabulary due to limitations in constructing the dictionary. To address this issue, these words need to be substituted with a special word, such as "unknown." However, the prediction results for these words often tend to be poorer compared to others. To mitigate this problem, we employ character-level embeddings to represent input tokens, which extract morphological information from each word token in addition to word embeddings. To accomplish this, we utilize a CNN (Convolutional Neural Network) to capture character-level details, such as word prefixes and suffixes. Previous studies have shown that CNNs are effective in representing words based on their characters S. Gajendran et al. (2017) M. Gridach et al. (2017). As depicted in Figure 3, each character in the word token is mapped to a corresponding character vector.

Subsequently, filters of different sizes are applied to the embedding matrix to capture important features from adjacent inputs. We incorporate ReLU nonlinearities and dropout between the CNN layers. By employing this approach, the models generate a word vector that encompasses the most significant characteristics from each representation type.

To complete the representation of a word, we combine its character-level features with w_i^{emb}, a latent word embedding corresponding to that word:

$$w^{full} = (w^{char}; w_i^{emb}) \quad \text{Eq. 3}$$

In order to generalize for words unseen in the training data, is replaced with 50% probability while training, an approach that resembles the word-drop method as Lample et al. (2016) suggested.

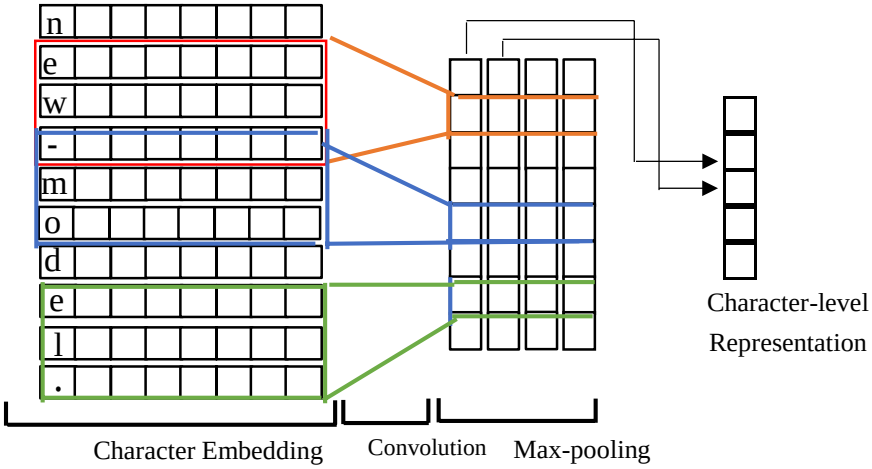

Figure 3: CNN-based character-level word representation

4. Experiments

At this point, we have focused on conducting a series of experiments to analyze the concert of the DNN for NER problem. Each word in a sentence provided as a sequence is denoted by joint vectors to preserve a feature at inter-word and intra-word level features. Also, due to the complex nature of the language, extra-linguistic feature, POS tag information of each word employed and explored its effect on NER system development. We implement our experiments in three scenarios: In the beginning, we execute the experiments on the BiLSTM and BiLSTM with CRF at the output layer instead of Softmax with word embedding only. Then, we run the same architecture by adding character embedding. At this time, we infix a CNN for character embedding. Following this, we introduce an additional feature, POS tag feature in addition to the preceding featureson BiLSTM-CNN-CRF. Meanwhile, the performance of BiLSTM with CRF on the output layer against Softmax has analyzed. Finally, we analyze the effect of training size on the proposed model.

4.1 The Dataset preparation

For NER experiment, we composed the Amharic and Afan Oromo corpus from texts, representing standard Amharic Ethiopian News Agency (ENA), and "Bariisaa and Kallacha Oromiyaa" respectively. Since NER is a domain-dependent problem, we tried to expand our corpus by taking News texts from two domains (Infrastructure and

Accident News) for Amharic. For Amharic, the corpus composed of Newswire articles with a total of 3461 and 1700 sentences, 61500 and 25300 running words, respectively; and for Afan Oromo, a total of 4014 articles were collected from a newsroom room that contains more than 23,692 words. We annotated each token of both corpora with five predefined categories (PER-TITLE, PER, LOC, ORG, and TIME) based on the IOB scheme manually. Furthermore, to increase the size of the Amharic dataset we employed the ready-made corpus, the SAY corpus, which contains 4,237 sentences and about 109,676 tokens), collected and annotated by R. Al-Rfou, (02013). In this corpus, 5,480of the tokens are named entity tokens.

Afan Oromo's corpus size is far less than the CoNLL 2003 shared task corpus, which contains more than 300k tokens, so we can't take this corpus as quite high for the intensive experiment, however, our Amharic corpus contained 200k tokens, so that, we can say this corpus is fair enough for Amharic NER system development. Because the training dataset is about 10 times larger than the previous experiments conducted on Amharic NER, hence we overcame the data limitations of those experiments. Statistics about the number of tokens distributed belonging to their appropriate classes are presented below.

Table 1: Number of instances of all NEs in both corpora

AM_NER		**Oro_NER corpus**	
Class Label	Number of Instances	Class Label	Number ofInstances
PER-TITLE	4391	TITLE_PER	912
PER	4464	PER	877
LOC	3841	LOC	589
ORG	3457	ORG	402
TIME	912	Time	92
All NEs	17065	All NEs	2872
O	179,411	O	20,820
Total	**196,476**	**Total**	**23,692**

4.2 Data Representation

Once the dataset is per process, the data representation is conducted to make the training data fit for the architecture. The first representation process was turning each sentence into an index of terms, and then each word would be represented by a unique index in the vocabulary constructed from the training set. Before heading into the actual model, one more thing has done is converting each label into one-hot encoding. Finally, we sat cross-validation with 80%, 10%, and 10% as training, development, and test data, respectively. For the sake of reproducibility, the random state is resolved.

4.3 Hyper-parameter setting

Optimal hyper-parameters can often make the difference between mediocre and state-of-the-art performance. We used the training set to learn the model parameters, the development set to select the optimal hyper-parameters, and the test set to report final results. For BiLSTM word- levelencoder, we use a two-layer model with 300 hidden neurons for both types of experiments (based on Chiu & Nichols (2010)). For CNN character-level encoder, we employed a single layer CNN with a kernel width of 3 and 50 filters (based on Chiu & Nichols (2010)). Dropout probabilities are all set as 0.5. We uniformly set the step size as 0.001 and the batch size as 64. By default, the additional feature,

POS tag of each token was incorporated through a 10-dimensional embedding. Early stopping was applied if there was no improvement after 10 epochs and the average training time for each epoch was also recorded. Other hyperparameters were also fixed as presented in *Table 2* during initialization.

Table 2: Hyper-parameter search space and values used in all experiments

	Hyper-parameter	**Value**
	Input dimension	n_words+1
Embedding Layer	Maximum input sequence length	Length of the longest sentence
	Word-level embeddings dimension	100
	Character embeddings dimension	25
BiLSTM Layer	Recurrent dropout	0.6
	Return sequences	True
	Dropout	0.5
	The number of recurrent units in each LSTM layer	100
	Vector size of character (charEmbedSize)	30
	BiLSTM layer	1
	LSTM size	25
The parameters during model fitting	Network cost optimizer:	Adams
	Epochs	20
	Batch size	64
	Hidden activation function	400 Tanh
	Learning rate	$1e^{-0}$, ---- $1e^{-5}$
CNN	charEmbedSize	30
	Window size	3
	# of filters	30

5. Experimental results and evaluation

To evaluate the enactment of our model as well as other alternatives, we run the experiments on two datasets: Am_NER and Oro_NER by using different features. The experimental results, presented in Table 3, pointed out that our model can obtain an acceptable performance on both datasets with only 80 samples for training and 20 samples for validation.

The gap in NER performance among the BiLSTM and BiLSTM-CRF with only word embedding in all experiments is quite large, Softmax gives the lowest performance for all settings indicates, that the CRF classifier is handy while integrating with neural network architectures for NER, this is an existing fact in different studies too I. El Bazi and N. Laachfoubi (2019) S. Gajendran et al. (2020). As Table 3 presents our empirical results, all models with the CRF layer outperformed the baseline modelin all named entities. On the Am_NER dataset, the training speed of CNN-BiLSTM is a comparable learning time with CNN-BiLSTM-CRF. However, on the Oro_NER dataset whichhas five named entity types, the learning time of CNN-LSTM is twice faster as CNN-BiLSTM-CRF, as the time complexity of computing the partition function for CRF is quadratic to the number of Named entities. Each model based on BiLSTM and BiLSTM-CNN achieved 80.07%and 83% accuracy, respectively.

5.1 Impact of character embedding

We argue that the morphological nature of both languages can be the leading cause for unableto achieve the maximum performance of BiLSTM using only word embedding. Nonetheless, itis easy to hypothesize that "such a matter can be managed by counting character-level learning S. Gajendran (2020), M. Gridach (2017). Notably, the model accomplishes a major improvement in both datasets while character-level features are added to the network. The experimental result revealed that the model with character-level embedding achieved 92.88% accuracy and 89.8% accuracy, outperforming BiLSTM-CRF by approximately 3% and 1.62% in absolute terms on Am_NER and Oro_NER datasets respectively, indicating that character-level embedding is useful for handling unknown words.However, the improvement in Oro_NER is mainly observed on ORG, PER, and LOC entities,implying that word embedding alone is, in some classes, sufficient, and adding character embedding may degrade or retain so rather than improve.

Table 3: Impact of character-level embedding

Model	Feature	Am_NER dataset		Oro_NER dataset	
		Accuracy (%)		Accuracy (%)	
		Validation	Test	Validation	Test
BiLSTM	Word embedding	91.38	85.24	83.05	77.7
BiLSTM-CRF	Word embedding	92.29	88.05	85.47	78.58
BiLSTM-CRF	Word and POS tags embedding,	96.13	89.87	87.32	76.14
BiLSTM-CNN	Character, word embedding	94.21	87.6	89.2	79.25
BiLSTM-CNN-CRF	Character, word embedding	99.02	91.24	89.8	80.2
BiLSTM- CNN-CRF	Character, word, and POS tags embedding,	99.89	92.88	90.12	82.35

5.2 Effects of POS tag embedding

Moreover, we examine the cases where our proposed model gives a good performance. Since we are typically interested in proper nouns, we hypothesized that "providing a syntactic class of each word could be very effective in improving the overall performance of named entity identification". Table 4 shows that when the POS tag feature is combined with the word and character embedding, has a positive influence on the model's performance. A moderateimprovement in the Am_NER dataset has been obtained, which is 0.83 % over the previous highest accuracy, but in the Oro-NER dataset, nothing change is found in the accuracy of the model over the previous highest accuracy. This demonstrates that combining the POS tag information with character embedding into the embedding vector seems to be advantageous because they have a vital role in capturing different characteristics of the terms.

Table 4: Impact of POS tag feature

Model	Feature	Accuracy
BiLSTM-CRF	Word embedding	89.87
BiLSTM-CRF	Character + word embedding	91.24

BiLSTM-CNN-CRF	Character + word + POS tag embedding	92.88

5.3 Effects of training data size

In addition to analyzing the feature set, we also measured how the training data size might affect the learning capability of the proposed model. Different experiments were undertaken using both word representation and features with distinctive sizes of the training data. An interesting fact that we observed in our experiments is that, although all mechanisms have a positive influence overall, the performance enhancements were generally continued as the training data size is increasing. Initially, the experimentation was conducted with only 40% ofthe total training data. Afterwards, a 20% additional training data sample is added, and the sameexperiment is repeated. When the number of samples is increased, our model nearly yields near-best performance as revealed in Figure 4.

This demonstrated that the performance gap between the models becomes greater in favor of increased training data size. Thus, we can claim that to get a powerful NER system using LSTMarchitecture, the large size of training data is a top priority. During the training process, overfitting was a concern due to the small size of the Oro-NER datasets.

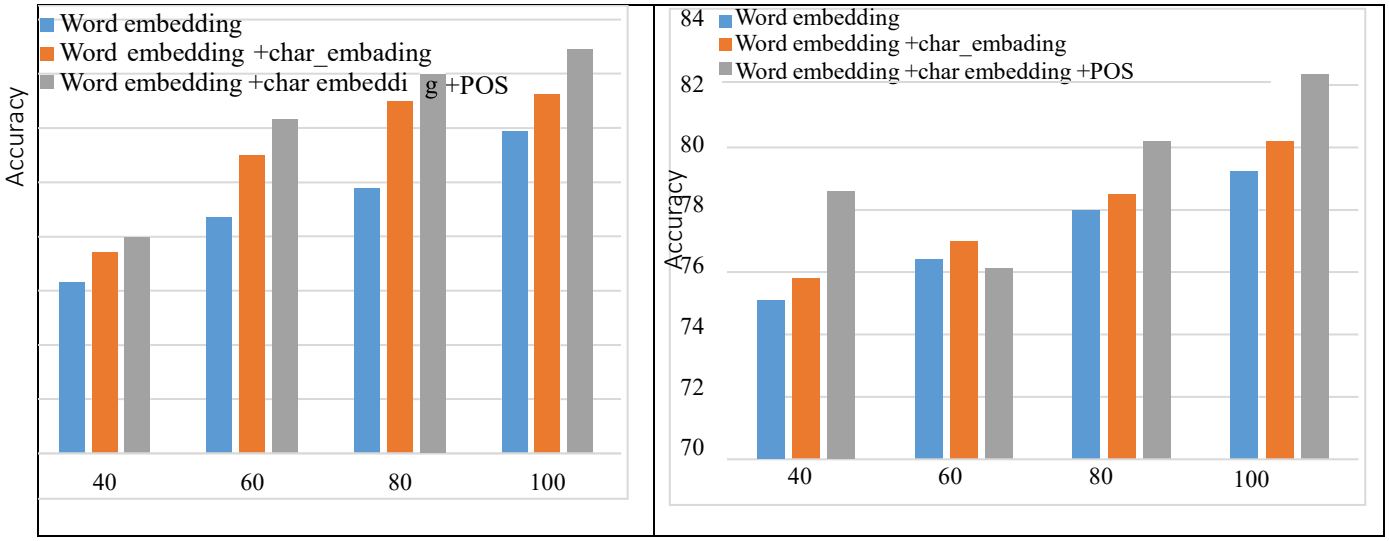

Figure 4: Illustrate the effect of different features on different training data sizes

Although we endeavored to crack this problem by applying a variety of regularization techniques, it is still challenging to resolve the issue of data shortage.

Furthermore, as Figure 5 portray, the validation accuracy of the BiLSTM-CRF based model was increased at each epoch. We ran the algorithm for 40 epochs, and we came to know that 20 epochs are optimal on the AM_NER dataset, as we see that the validated accuracy achievedin the 20th epoch is continued till to the end with only insignificant fluctuations.

Figure 5. Illustrates the effects of the number of epochs on the accuracy and loss Besides, the learning rate of the model at some intervals (1e^{-0}, 1e^{-1}, 1e ^{2}, 1e^{-3}, 1e^{-4}, 1e^{-5}) hasbeen examined. We have observed from the experiment is the BiLSTM-CRF based model canno longer learn when the learning rate is less than 1e-4 but increases from this point onward significantly.

We should note that our proposed BiLSTM-CNN-CRF model's performance is not directly comparable to the models conducted thus far for Amharic and Afan Oromo; almost all works were conducted using the classical machine learning model, and only one work has been implemented using a neural network, BiLSTM, by T. Mikiyas (2017).

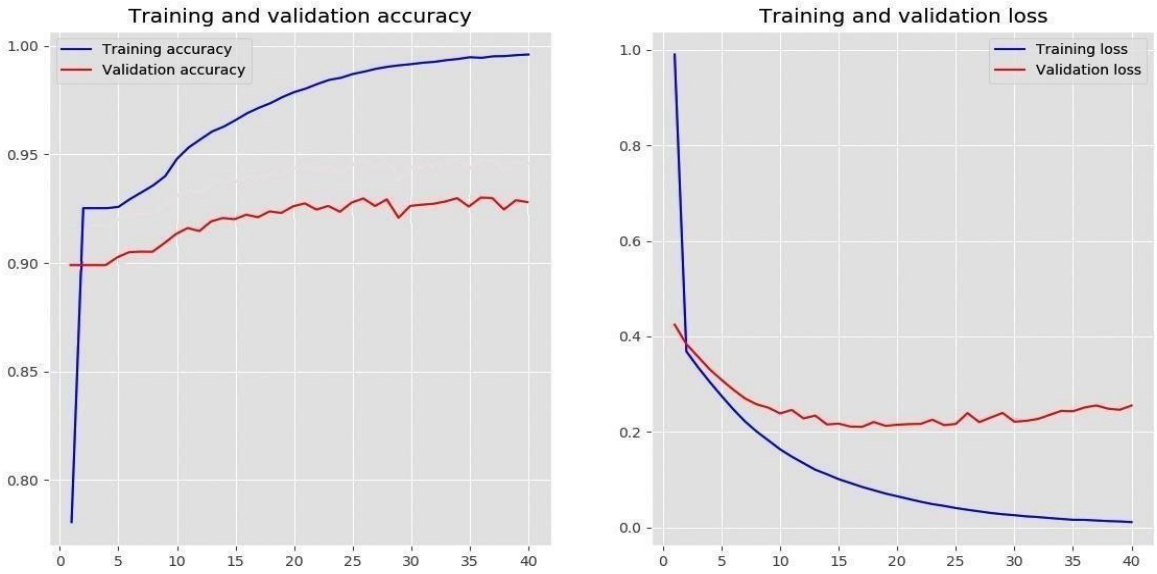

Figure 5: The effects of the number of epochs on the accu-racy and loss

As provided in Table 5, our proposedmodel outperformed all previously conducted studies in both languages. We can reason out why our study outdid these studies are: first of all, the corpus they used in their experiment wasan average of 11000 tokens, which is very small, but Gambäck and Sikdar (2017) used the medium-sized (SAY Amharic) corpus. Second, the algorithms, except Gambäck and Sikdar (2017) all are employed classical machine learning algorithms. Even though Gamback and Sikdar used BiLSTM for AmharicNER, they didn't analyse different modalities like character embedding, and hybridizing models, thus the performance of this work is not satisfactory as compared with a state-of-the-artresult. Hence, for both languages, compared to previous non-deep-learning methods, the BiLSTM-CNN-CRF models have a significant advantage in the accuracy of named entityidentification.

Table 5: Performance comparison for NER models in Ethiopia language

Paper	Corpus	Accuracy
Gambäck & Sikdar (2017)	SAY corpus	69.7%
Kejela (2017)	23,000 words	76.60%
Moges (2007)	10,340 tokens	74.61%
Besufkad (2013)	13,538 tokens	80.66%
Mikiyas (2017)	12,000 tokens	85.9%

Furthermore, we also analogized the performance of our model with the models developed for other languages using publicly available datasets.

Although there are numerous NER models developed using DNN, we believe that the most reliable NER models are those presented in competitions like MUC and CoNLL. Particularly, these are appeared in MUC6 (2019), MUC7 (2003)], and CoNLL (2003) , which marginally outperformed the accuracy of our proposedmodel as their accuracy is 96%, 94%, and 95.1%, respectively. We argued that there are two major factors that contribute unable to

achieve the best performance compared to these studies. First, the corpus's quality and quantity: the NER models that appeared in competitions such as MUC6 and CoNLL are trained on a super standard dataset. Also, the training data size they used was about 300K annotated tokens in different domains, but our model trained on very small as compared to these works, about 196K and 23.6K tokens for Amharic and Afan Oromo respectively. It is still instructive to compare our results with those morphological rich language NER systems like Arabic, though we used a different type and number of datasets. Generally, although our model achieved lowerperformance compared to above-mentioned previous works, it achieved the highest performance in morphologically rich Ethiopian languages (Amharic and Afan Oromo), outperforming the previously studied models.

6. Conclusions and Future Work

This paper has proposed a system for named entity identification and classification in an under-resourced Ethiopian language. Better performance was achieved after merging the different feature vectors with word embedding generated with word2vec and then feeding the results to the DNN for training and classification. Using character-level features significantly improves the tagging accuracy, especially in the case that the words that don't exist in a lexical dictionaryand the training set. In our model, we use a CNN network to capture character-level features due to its fast speed compared with the Bi-LSTM network.

In addition, the POS tags also are good features for the task of NER. In our experiments, we combined these features with word and character embedding. This aided to surge a little on ttheOro-NER dataset, but significant on the Am-NER dataset. In the decoding stage, using theCRF model is better than just applying the Softmax function due to the ability of the CRF model to make global decisions that depend on not only the representation vectors of input words but also the linear dependencies between tagging decisions.

During the training process, overfitting was a big concern due to the small size of the Oro- NERdatasets. While we tried to resolve this problem by using several regularization techniques, it is still challenging to resolve the issue of insufficient data. In future work, we plan toincorporate the transferring learning approach to improve performance.

Moreover, in the future, it would be reasonable to also develop some language-dependent features to improve the performance. Furthermore, it might be possible to utilize the word embedding generated for Amharic in the Polyglot or HaBiT projects (h.eu). A set of models can also be generated using several different classifiers and ensemble models with an evolutionary algorithm.

Conflict of Interes

We hereby declare that there are no conflicts of interest related to this research.

References

Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot: Distributed word representations for multilingual NLP. In Proceedings of the 17th Conference on Computational Natural Language Learning (pp. 586–594).

Besufkad, A. (2013). A named entity recognition system for Amharic (Master's thesis). Addis Ababa University.

Chiu, J. P. C., & Nichols, E. (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics, 4*, 357–370.

Chopra, D. (2012). Named entity recognition in Indian languages using gazetteer method and hidden Markov model: A hybrid approach. *International Journal of Computer Science & Engineering Technology, 3*(12), 12–20.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the Machine Learning* (pp. 261–269).

Dang, T. H., Le, H. Q., Nguyen, T. M., Vu, S. T.

(2018). Biomedical named entity recognition using BERT in the machine reading comprehension framework. *Bioinformatics, 34*(20), 3539–3546.

El Bazi, I., & Laachfoubi, N. (2019). Arabic named entity recognition using deep learning approach. *International Journal of Electrical and Computer Engineering, 9*(3).

Gajendran, S., Manjula, D., & Sugumaran, V. (2020). Character level and word level embedding with bidirectional LSTM – Dynamic recurrent neural network for biomedical named entity recognition. *Journal of Biomedical Informatics, 112*, 158–167.

Gambäck, B., & Sikdar, U. K. (2017). Named entity recognition for Amharic using deep learning. In *IST-Africa Week Conference* (pp. 1–8).

Gridach, F. M. (2016). Character-aware neural networks for Arabic named entity recognition for social media. In *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing* (pp. 23–32).

Gridach, M. (2017). Character-level neural network for biomedical named entity recognition. *Journal of Biomedical Informatics, 70*, 85–91.

Habibi, M., Weber, L., & Neves, M. (2017). Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics, 33*(14), 37–48.

Huang, Z., Xu, W., & Yu, K. (2018). Bidirectional LSTM-CRF models for sequence tagging. *arXiv*.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv*.

Legesse Kejela, M. (2017). Named entity recognition for Afan Oromo (Master's thesis). Addis Ababa University.

Li, J., Sun, A., Han, J., & Li, C. (2022). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering, 34*(1), 50–70.

Luo, L., Yang, Z., Pei, Y., Zhang, Y., Wang, L., Lin, H., & Wang, J. (2018). An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition. *Bioinformatics, 34*(8), 1381–1388.

Ma, X., & Hovy, E. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (pp. 120–131).

Mikiyas, T. (2017). Amharic named entity recognition using a hybrid approach (Master's thesis). Addis Ababa University.

Moges, A. (2007). Named entity recognition for Amharic language (Master's thesis). Addis Ababa University.

Reimers, N., & Gurevych, I. (2017). Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings on Empirical Methods in Natural Language Processing* (pp. 338–348).

Richa, S., Sudha, M., & Agarwal, B. (2018). Named entity recognition using neural language model and CRF for Hindi language. *Computer Speech & Language, 74*, 210–220.

Tjong, F. E., Sang, K., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003* (pp. 142–147).

Xu, C., Wang, F., Han, J., & Li, C. (2019). Exploiting multiple embeddings for Chinese named entity recognition. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (pp. 2269–2272).