# Chapter 3

# Baseline Experiments on Low Resource Languages for Sequence Labeling

## 3.1 Introduction

With the relatively sudden and so far enduring success of deep neural network based approaches, Artificial Intelligence (AI) technologies have come to play a vital role in the individual and collective lives of people across the world. Along with the new algorithms, the abundance of data for machine learning and the development of suitable computing resources, this almost radical change has affected AI problems. Natural Language Processing (NLP) has also benefited from the new AI approaches. The recent rapid development of methodologies and models have substantially affected different tasks, like word and sentence labeling, and have allowed new technologies to exceed the previous state-of-the-art results.

Part-of-speech (POS) tagging assigns a sequence of grammatical categories (POS tags) to the given word sequence (sentence), while Chunking links POS tagged words into groups

of words or 'chunks', which can be roughly defined as minimal phrases or minimal constituents. Named Entity Recognition (NER) is the process of identification of named entities (Person, Organization, Location etc) in natural language text. These tasks are often performed in the preliminary stages of any complex language processing task. Thus, for any new language, i.e., a language that does not yet have language tools developed for it, it is crucial to first build applications for performing these tasks for that language. These tasks are usually part of a Natural Language Processing (NLP) pipeline and play a significant role in various more complex tasks such as Question Answering, Information Extraction, Machine Translation, Search Engines, Automatic Indexing, Document Classification, Text Summarization and so forth. Even these tasks are essential tasks for computational purposes because it is not possible to build end-to-end Deep Learning systems for low resource languages due to the lack of data.

There are more than 7000 natural languages[1] that are still widely used in the world. It is important to remember that most of these languages are low resource languages or resource scarce languages [44]. This fact becomes even more important and relevant if we consider that many of these low resource languages are among the most spoken languages of the world. This is particularly true of the languages of South Asia. The majority of Indian (or South Asian) languages do not yet have as large or advanced language resources or language processing applications as we see for most European languages. Deep neural networks are a general paradigm to solve language processing problems, but their applicability to low resource languages is hampered by the fact that they need large quantities of appropriate data (usually with some kind of human annotation). Still, deep learning-based approaches have been used to develop language technology tools for low resource language [227, 209, 193]. This is made possible by various advances in such approaches that address the problem of resource sparsity.

I have evaluated Bhojpuri, Maithili and Magahi languages for POS tagging, Chunking and NER tools development. For which the baselines machine learning algorithms such as conditional random fields, maximum entropy Markov model, structured support vector machine and Tri-n-gram and deep learning technique, CNN-LSTM-CRF [140] have been used. All these learning algorithms have been evaluated on the POS and Chunk dataset

[1] `https://www.ethnologue.com/guides/how-many-languages`

and the best result provider algorithms in both machine and deep learning evaluated on the NER dataset since the annotated dataset of NER is comparatively less to POS and Chunk. Later on, a novel deep learning based architecture has been proposed that utilizes the concept of transfer learning.

As an example of advances in deep neural networks which address the sparsity issue, Transfer Learning techniques [259, 113] work well in a scenario where an ample amount of annotated data of source language is available, but not of the target language data, which is the low resource language. A large amount of annotated data of the source language which is resource rich from related tasks can help in enhancing the performance of deficient annotated data of the target language. The advantage of Transfer Learning is, thus, that it does not require additional resources for the target language to mitigate data sparsity as explored in the following section on related work. The deep learning model for the same task, say POS tagging, is trained on both the source and target language. After that, the model from the source language data is partially transferred to the target language by way of sharing the hidden representation (weights) between the two languages.

Here, Bhojpuri, Maithili and Magahi are the target languages, which are closely related to Hindi as the source language. Although Hindi is still not as resource rich as even some of the less spoken European languages, such as German, it is still relatively richer to the extent that we can get some useful results. Like most Indian languages, all these four languages have SOV word order [233], and they all use the Devanagari script for their writing systems. Linguistically, they belong to the same sub-family of Indo-Aryan (IA) languages. In fact, till recently, the three concerned languages were considered as dialects or variants of Hindi. This makes for an ideal situation for model transfer for POS tagging and Chunking. Later, we compare the results with deep learning based current baseline technique using monolingual embeddings. We found that Transfer Learning-based methods indeed outperform conventional machine learning for these languages on POS tagging and Chunking. These results show the potential of leveraging the Hindi model's parameters could help many other languages, which are also similar to Hindi. We further investigate if it is possible to get more improvement on the sequence labeling problem by fine-tuning the disambiguation layer.

Plank et al. used a multi-task Bi-LSTM model with auxiliary loss and they evaluated tokens and sub-levels for neural network-based POS tagging. The auxiliary loss can be used to improve the accuracy of rare words [189]. Mishra et al. used feature transfer from a rich-resource language to resource-poor languages, without any knowledge of the target language and human annotation [161].

The first reported attempt towards the development of deep learning-based Maithili POS tagger was trained on a continuous bag of word model (CBOW) word embedding trained with the help of available web resources and Wikipedia dump as corpus [192]. As such, no substantial state-of-the-art work on sequence labeling problems for Purvanchal languages had been attempted using deep learning techniques to the best of our knowledge.

## 3.2 Contributions of Chapter

Bhojpuri, Maithili and Magahi are Purvanchal languages that are often considered dialects of Hindi, even though they are widely spoken in parts of India. Bhojpuri is spoken even outside India. Due to their dialectal nature, they show more linguistic variations such as nominal case inflection and emphatic expressions. Like other computational resources, there is a lack of POS, Chunk and NER tools for these languages. Hence, state-of-the-art machine learning and deep learning approaches have been used and evaluated the obtained results. As a baseline, we have evaluated TnT, SSVM, MEMM and CRF as machine learning and LSTM-CNNs-CRF as a deep learning model. These systems are planned to be used in machine translation systems for Bhojpuri, Maithili and Magahi to Hindi. We have proposed two deep learning models: SAHBiLC (monolingual embedding) and Fine-SAHBiLC (character-level transfer learning) for POS tagging and Chunking. The chunk tool of Magahi is not in the scope of this thesis. SAHBiLC and Fine-SAHBiLC outperform Bhojpuri and Maithili, Magahi, respectively for both tasks. This indicates that fine-tuning is helpful in case of less training data and for complex morphology. However, we compare the results of CRF and LSTM-CNNs-CRF for the NER datasets of these languages. Here, the deep learning baseline provides a better result for Magahi only due to fewer intermediate entity tags. We observe that the obtained results are consistent with

the number of named entities in the datasets, rather than with the total size of the dataset in the number of tokens.

## 3.3 Methodology

For deep learning approaches, words are described as real-valued vectors in a low dimensional semantic space, usually a continuous vector space. The conventional way for such representations is to compute the term-document occurrence matrix on large corpora and then reduce the dimensionality of a matrix by singular value decomposition [30, 60, 241, 45]. Over recent years, that conventional two-phase approach has been replaced by a single supervised or unsupervised method, usually based on neural networks [130]. Numerous approaches [152, 155, 185, 106, 130, 129] have been proposed to learn word vectors based on co-occurrence of word in a corpus, generated word embeddings or distributional vector that points to similar meanings when it appears in the same context as measured by cosine distance, which means word vectors are capable of capturing the characteristics of surrounding words. The information about word morphology is not explicitly considered while building word representations because learning takes place only to capture syntactic and semantic information through corpus-based distribution. However, intra-word information can be immensely useful, especially when the language is morphologically rich and particularly for sequential processing such as part-of-speech tagging [211]. In low resource languages, the appearance of unknown words or out-of-vocabulary (OOV) words or tokens (such as dates and times) is an obvious problem that can be handled by character level embedding, where each word is represented as a sequence of characters.

The proposed model, Self Attention based Hierarchical Bi-LSTM CRF (SAHBiLC), which follows the hierarchical RNN-CRF as a base architecture [259]. The SAHBiLC has three components which capture the sub-word information, sensitive contextual information and label dependencies, as shown in Figure 3.1. The character level Bidirectional LSTM aids to elicit morpho-semantic information and affix information in a hidden representation without explicitly being fed to the network. The word level neural network, also implemented through bidirectional LSTM, can reveal sensitive syntactic information and extract

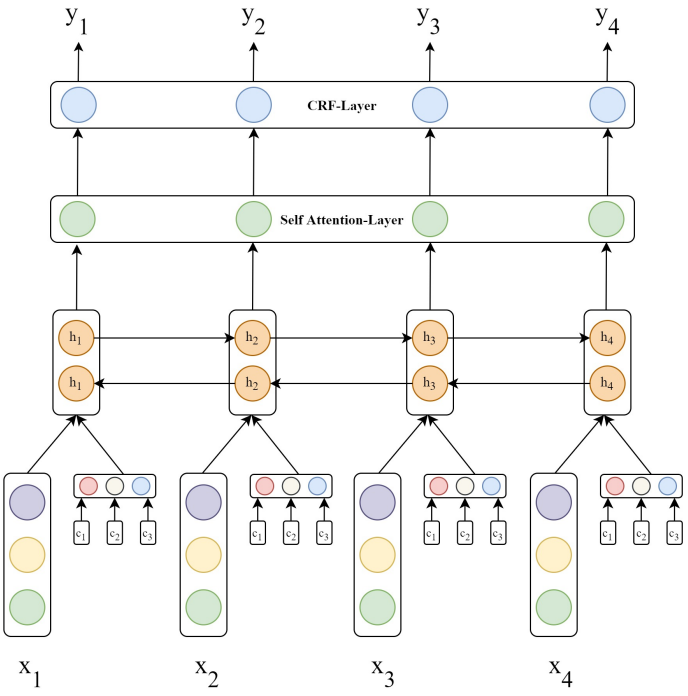


FIGURE 3.1: SAHBiLC model architecture for POS Tagging

dependency relations based on the self-attention layer from the input sentence, which extends the base architecture. A linear chain CRF helps to resolve the dependency among labels and produces inference. Given the input sequence of words $x_1 \dots x_i$, and character sequence corresponding to each word $c_1 \dots c_n$, the model first takes the word $x_i$ as input and extracts continuous representation through a distribution embedding. The input for character level $\overrightarrow{LSTM}$ is the character sequence corresponding to the word, each character encoded into continuous representation through a neural network embedding layer. The output from the end node of character level LSTM concatenates with word representation and admits it as the final word representation, which includes the sub-word information. The sequence of POS tags $t_1 \dots t_i$ corresponding to the sentence is concatenated with the word embedding as an additional input for Chunking.

$$w_i = x_i \oplus \overrightarrow{LSTM}(c_1 \dots c_n)_i \tag{3.1}$$

$$w_i = x_i \oplus \overrightarrow{LSTM}(c_1 \dots c_n)_i \oplus t_i \tag{3.2}$$

The word level Bidirectional LSTM layer works as a disambiguation layer [169] for sequence labeling tasks. The Final word representation is fed to the forward and backward LSTM layer and concatenates resultant representation for each word, holding the sentence-level syntactic information.

$$h_i = \overrightarrow{LSTM}(w_i, \overrightarrow{h_{i-1}}) \oplus \overleftarrow{LSTM}(w_i, \overleftarrow{h_{i-1}}) \tag{3.3}$$

Self-attention [242] produces a context vector for each word of the sentence which is independent of word positions. It provides flexibility concerning word order and helps to generalize better to hold contextual meaning. Each word's previous output (bidirectional LSTM) is multiplied by the Key, Query, and Value weight vectors. Xavier initializes these weight vectors. Attention scores have been generated after performing product operations between each Query and all Keys obtained, over which softmax is performed. The generated softmax attention score for each word is multiplied by its corresponding Value, and these weighted values are summed to obtain the context weight vector.

The output from self-attention, i.e., context weight vector merges with the disambiguation layer of the hidden representation. The CRF layer is employed after the self-attention layer and it is fed concatenated hidden representation:

$$H = [h_i \dots h_i] \tag{3.4}$$

$$h_i = Self\ Attention(H) \tag{3.5}$$

$$y' = argmax_y \left[ \prod_{i=1}^{T} exp \left( \sum_{k=1}^{K} \lambda_k f_k(y_{i-1}, y_i, h_i) \right) \right] \tag{3.6}$$

## 3.4 Experiment-I: POS Tagging and Chunking

### 3.4.1 Dataset Description

To conduct the tagging experiments with Purvanchal languages, we used them as low resource languages (target languages), while Khari Boli (Standard Hindi) was used as the high resource language (source language). Purvanchal languages' dataset from Mundotiya et al. [168] and Hindi dataset from Tandon et al. [236] are used here for the task of POS tagging and Chunking. The annotated data was tagged with the BIS tagset[2] for Indic languages. Chunk annotated dataset is not available for the Magahi, and Hindi is utilized to correlate the obtained results. The basic statistics of the corpus of each language are mentioned in Tables 3.1, 3.2. For example, the maximum sentence length of Bhojpuri, Maithili and Magahi is 118, 272, 109 words, respectively. The most common POS tags for the three languages were found to be NN (noun), VM (main verb), PSP (pronoun) and SYM (symbol, or roughly punctuation), and they cover 50% the annotated data, whereas CL (classifier), ECH (echo word), UNK (unknown word) and UT (quotative) are frequent, but their use is more subtle. From this dataset, we remove those sentences whose length is less than two as part of preprocessing for both the learning techniques.

TABLE 3.1: Annotation statistics of POS tagging datasets

| **Language** | **POS annotation** | | |
|---|---|---|---|
| | **# Sentence** | **# Token** | **# Types** |
| Bhojpuri | 16067 | 245482 | 26202 |
| Maithili | 12310 | 208640 | 21410 |
| Magahi | 14669 | 171509 | 14077 |
| Hindi | 20783 | 434856 | 22171 |

TABLE 3.2: Annotation statistics of Chunking datasets

| **Language** | **Chunk annotation** | | | |
|---|---|---|---|---|
| | **# Sentence** | **# Token** | **# Types** | **# Chunks** |
| Bhojpuri | 9695 | 60588 | 18090 | 40239 |
| Maithili | 1954 | 10476 | 5764 | 10436 |
| Hindi | 20783 | 434856 | 22171 | 233864 |

[2]http://tdil-dc.in/tdildcMain/articles/134692Draft%20POS%20Tag%-20standard.pdf

The proposed deep learning based model, SAHBiLC has been compared with traditional machine learning techniques which have become popular over time, such as Trigrams 'n' Tags (which is an extended variation of the HMM model, using interpolated smoothing), Maximum Entropy Markov Model, Conditional Random Fields and Structured Support Vector Machine (as discussed in section 2.2), which have been using for sequence tagging on Bhojpuri, Maithili and Magahi.

### 3.4.2 Machine Learning Strategy

The preprocessed annotated dataset is divided into a 1:3 ratio for training and validation set. The experiment was conducted using $k$-fold cross-validation, where 10 is the value for $k$, with a random sample generator through random seed to provide robustness and prevent overfitting. We conducted experiments with different training-validation ratios, as they require external features, and this helps us get more reliable predictions and use the learned model for evaluation. CRF, SVM and MaxEnt are feature-based techniques, the default feature set for these techniques is used, as shown in Table 3.3.

TABLE 3.3: Feature set used for POS Tagging and Chunking in machine learning techniques and contextual boundary value is upto 3, represented by $j$.

| **Category** | **Feature** | **Feature Values** |
|---|---|---|
| POS Tagging | Word Position | $i$ |
| | + Prefix's | $p_{i-1}, p_i, p_{i+1}$ |
| | + Suffix's | $s_{i-1}, s_i, s_{i+1}$ |
| | + Prefix's length | $p_{i-1_j}, p_{i_j}, P_{i+1_j}$ |
| | + Suffix's length | $s_{i-1_j}, s_{i_j}, s_{i+1_j}$ |
| | + Contextual Word | $w_{i-1}, w_{i+1}$ |
| | + Contextual Word length | $w_{i-1_j}, w_{i_j}, w_{i+1_j}$ |
| | + Digits | $d_{i-1}, d_i, d_{i+1}$ |
| | + Punctuation | .?!\| |
| Chunking | POS Tagging | |
| | + Contextual POS tag | $t_{i-1}, t_{i+1}$ |
| | + Contextual POS tag length | $t_{i-1_j}, t_{i_j}, t_{i+1_j}$ |
| | + Hyphenated | $-$ |
| | + Symbol | , - |

The CRF technique is trained on gradient-based L-BFGS algorithm with L1 and L2 regularisation till 100 iterations and we retain the remaining parameters as the default. Similarly, we have taken the same number of iterations with MaxEnt, and the threshold value for a rare word is 10. We use TnT and SVM with default settings which are available in NLTK and SVMTool, respectively.

### 3.4.3 Deep Learning Strategy

The SAHBiLC model takes word and character as input for POS tagging, while POS information is used as input only for Chunking. For the SAHBiLC model, the inputs are made in equal length and eliminate the OOV tokens at a word and character level by adding special tokens that are $\langle PAD \rangle$, $\langle UNK \rangle$. The similar length characters and words have been transformed into a 10-dimensional vector and 80 dimensions for character vector and word vector by distributional representation, respectively. For Chunking also, the POS labels have been represented into a 200-dimensional vector. A 10-dimensional character vector helps to generate word embedding after applying LSTM with 120 hidden units. The word vector and word embedding perform disambiguation with 200 hidden units of Bi-LSTM, and 0.2 dropouts to reduce the ambiguity. The contextual weightage is assigned by multiplicative self-attention, regularized by L1 bias and L2 kernel. The whole training time takes 32 samples in each step and iterates to 10 epochs. For the Fine-SAHBiLC model, the vocabularies of character and word from low and high resource language merge together to create a shared embedding space for transferring the learned parameters as base parameters, instead of random initialization. We employ Adam optimizer with a learning rate of 0.001 for performing the training strategies with the parameters and hyper-parameters for both models mentioned above. The parameters and hyper-parameters are summarized in Table 3.4.

#### 3.4.3.1 Feature transfer

Monolingual extended hierarchical RNN-CRF model labeling quality can be improved while using an ample amount of training data of another language which could be a high

TABLE 3.4: Parameters and Hyper-parameters employed during training the SAHBiLC model

| **Parameters and Hyper-parameters** | **Values** |
|---|---|
| Batch size | 32 |
| Epoch | 10 |
| Char embedding | 10 |
| Word embedding | 80 |
| POS embedding | 200 |
| Char LSTM unit | 120 |
| Word Bi-LSTM unit | 200 |
| Dropout | 0.2 |
| Attention | Multiplicative |
| Optimizer | Adam |
| Learning rate | 0.001 |

resource language. Here, we use Hindi as the high resource language for sequence labeling tasks for Bhojpuri, Maithili and Magahi. The orthographic systems of Hindi, Bhojpuri, Maithili and Magahi languages are very similar, and most of the characters are common, as all three languages use the popular Devanagari script.

I share the character at character level Bidirectional LSTM between Hindi to the corresponding language, and it should disambiguate the Hindi word features from corresponding language word features induced from this layer. The disambiguation layer fine-tuned for each low resource language from Hindi. This fine-tuned model is named as Fine-SAHBiLC model. After sharing the sub-word information, the learned weights of Hindi are used for initialization for low resource language. This helps to improve the performance of the sequence labeling task.

### 3.4.4 Results and Analysis

In this section, we have explained the experimental result at the token level obtained on the test set. The stated result of machine learning techniques depends on feature sets for POS tagging and Chunking, as mentioned in Table 3.3. The performance of each system is evaluated in terms of Precision, Recall, F-score and Accuracy as a weighted average.

**Machine learning** techniques applied to POS tagging and Chunking provide a satisfactory result for all three languages; CRF outperforms for all four languages. The comparative result of POS tagging and Chunking at different scales after applying diverse machine learning techniques mentioned in Table 3.5 and Table 3.6, respectively. For Chunking, CRF model performs the best for Bhojpuri, Maithili and Hindi with 0.95, 0.94 and 0.99, respectively.

TABLE 3.5: Results of traditional machine learning techniques (%) in terms of Accuracy, Precision, Recall and F-score for POS Tagging

| Language | Technique | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|---|
| **Bhojpuri** | **TnT** | 0.83 | 0.83 | 0.83 | 0.83 |
| | **CRF** | 0.86 | 0.86 | 0.86 | 0.86 |
| | **MaxEnt** | 0.83 | 0.83 | 0.83 | 0.83 |
| | **SVMTool** | 0.85 | 0.87 | 0.85 | 0.86 |
| **Maithili** | **TnT** | 0.80 | 0.85 | 0.81 | 0.83 |
| | **CRF** | 0.85 | 0.86 | 0.85 | 0.85 |
| | **MaxEnt** | 0.84 | 0.84 | 0.84 | 0.84 |
| | **SVMTool** | 0.84 | 0.86 | 0.84 | 0.85 |
| **Magahi** | **TnT** | 0.81 | 0.84 | 0.81 | 0.82 |
| | **CRF** | 0.83 | 0.83 | 0.83 | 0.83 |
| | **MaxEnt** | 0.81 | 0.82 | 0.82 | 0.82 |
| | **SVMTool** | 0.80 | 0.83 | 0.81 | 0.81 |
| **Hindi** | **TnT** | 0.93 | 0.95 | 0.93 | 0.94 |
| | **CRF** | 0.94 | 0.94 | 0.94 | 0.94 |
| | **MaxEnt** | 0.93 | 0.94 | 0.94 | 0.94 |
| | **SVMTool** | 0.92 | 0.93 | 0.92 | 0.92 |

A comparative performance was obtained from SVMTool and CRF for POS tagging on less frequent (INJ, CL, ECH, UNK, UT) and most frequent (NN, VM, PSP, SYM) tags of Bhojpuri, Maithili and Magahi. The F-score for each tag on Bhojpuri, Maithili and Magahi, each technique is shown in the Figure 3.2.

Only the TnT technique was able to predict the VGINF chunk tag in Bhojpuri, whereas the remaining less frequent (RBP) and most frequent (NP, VGF, CCP) chunk tags were more correctly predicted by CRF as compared to other techniques. Similarly, Maithili chunk tag (for less frequent and most frequent) was accurately predicted by CRF. The F-score for each chunk tag on Bhojpuri and Maithili, each technique is shown in the Figure 3.3.

TABLE 3.6: Results for traditional machine learning techniques in terms of Accuracy, Precision, Recall and F-score for Chunking

| Language | Technique | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|---|
| **Bhojpuri** | **TnT** | 0.67 | 0.76 | 0.67 | 0.71 |
| | **CRF** | 0.95 | 0.94 | 0.95 | 0.95 |
| | **MaxEnt** | 0.92 | 0.92 | 0.93 | 0.92 |
| | **SVMTool** | 0.94 | 0.93 | 0.93 | 0.94 |
| **Maithili** | **TnT** | 0.69 | 0.80 | 0.70 | 0.74 |
| | **CRF** | 0.94 | 0.94 | 0.95 | 0.94 |
| | **MaxEnt** | 0.91 | 0.91 | 0.92 | 0.91 |
| | **SVMTool** | 0.92 | 0.93 | 0.93 | 0.93 |
| **Hindi** | **TnT** | 0.95 | 0.94 | 0.95 | 0.94 |
| | **CRF** | 0.99 | 0.99 | 0.99 | 0.99 |
| | **MaxEnt** | 0.97 | 0.97 | 0.98 | 0.97 |
| | **SVMTool** | 0.98 | 0.98 | 0.99 | 0.98 |

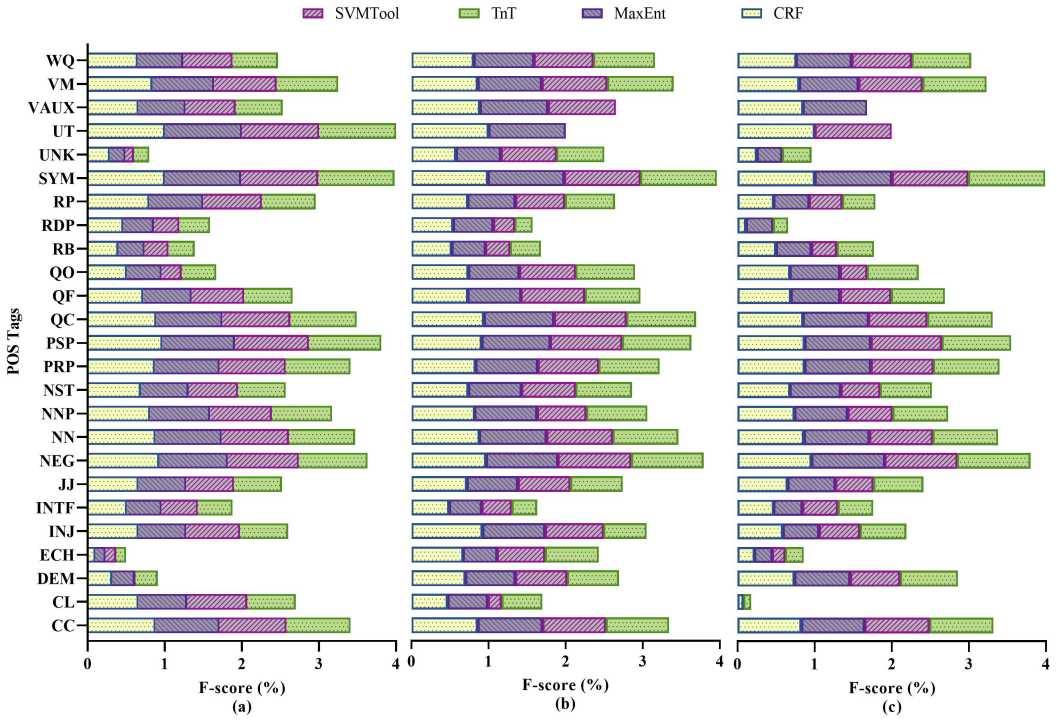


FIGURE 3.2: F-score result of POS tagging for (a) Bhojpuri, (b) Maithili and (c) Magahi

The proposed **Deep learning** based model for the POS tagging and Chunking has been compared with the state-of-the-art model, LSTM-CNN-CRF [140] and Hindi dataset. The results for deep learning based techniques for POS tagging are quite interesting because the

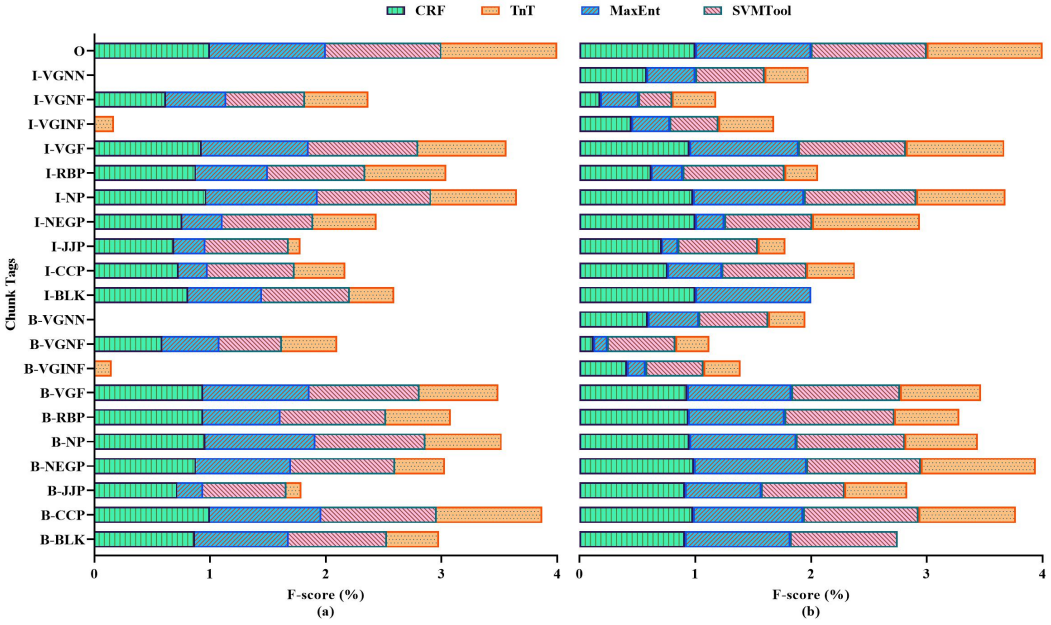


FIGURE 3.3: F-score results of Chunk tagging for (a) Bhojpuri and (b) Maithili

SAHBiLC model performs better for Bhojpuri, while Fine-SAHBiLC performs better for Maithili and Magahi, as shown in Table 3.7. After observing Table 3.8 for Chunking, SAHBiLC model performs better for Bhojpuri and Fine-SAHBiLC for Maithili. On comparing all the results in terms of F-score for POS tagging, we get Bhojpuri, Maithili, Magahi with 0.87 on the SAHBiLC model, 0.86 on Fine-SAHBiLC, 0.86 on Fine-SAHBiLC, respectively. Thus, our Fine-SAHBiLC model performs well for these languages where data size is low such as Magahi and Maithili, compared to Bhojpuri.

Deep learning-based techniques require a large amount of data to get effective features. In our case, not Hindi, but Bhojpuri, Maithili and Magahi are the low resource languages. According to the size of the annotated data, deep learning techniques extract effective features during training and provide more accurate results than traditional machine learning techniques. Maithili has minimally annotated data for POS tagging and Chunking compared to other languages for which the SAHBiLC model reported a lower result than CRF, but Fine-SAHBiLC improved SAHBiLC model performance further through passing its more effective features. The ratio of chunks (the number of tokens divides by the number of chunks) in Bhojpuri and Hindi is 1.50 and 1.85, respectively, indicating that

almost every word forms a Chunk. As a result, machine learning techniques provide better results compared to deep learning-based techniques. It might also perform better for morphologically complex languages, as fine-tuning might help in learning morphological idiosyncrasies.

TABLE 3.7: Results for Deep Learning techniques in terms of Accuracy, Precision, Recall and F-score for POS Tagging

| Language | Technique | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|---|
| **Bhojpuri** | **LSTM-CNN-CRF** | 0.84 | 0.84 | 0.84 | 0.84 |
| | **SAHBiLC** | 0.86 | 0.87 | 0.87 | 0.87 |
| | **Fine-SAHBiLC** | 0.85 | 0.86 | 0.85 | 0.85 |
| **Maithili** | **LSTM-CNN-CRF** | 0.83 | 0.82 | 0.82 | 0.82 |
| | **SAHBiLC** | 0.84 | 0.85 | 0.84 | 0.84 |
| | **Fine-SAHBiLC** | 0.86 | 0.86 | 0.86 | 0.86 |
| **Magahi** | **LSTM-CNN-CRF** | 0.83 | 0.83 | 0.84 | 0.83 |
| | **SAHBiLC** | 0.83 | 0.84 | 0.84 | 0.84 |
| | **Fine-SAHBiLC** | 0.86 | 0.87 | 0.87 | 0.86 |
| **Hindi** | **LSTM-CNN-CRF** | 0.92 | 0.94 | 0.91 | 0.92 |
| | **SAHBiLC** | 0.95 | 0.95 | 0.95 | 0.95 |

TABLE 3.8: Results for Deep Learning techniques in terms of Accuracy, Precision, Recall and F-score for Chunking

| Language | Technique | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|---|
| **Bhojpuri** | **LSTM-CNN-CRF** | 0.91 | 0.84 | 0.86 | 0.85 |
| | **SAHBiLC** | 0.94 | 0.94 | 0.94 | 0.94 |
| | **Fine-SAHBiLC** | 0.93 | 0.94 | 0.93 | 0.93 |
| **Maithili** | **LSTM-CNN-CRF** | 0.91 | 0.86 | 0.87 | 0.87 |
| | **SAHBiLC** | 0.93 | 0.93 | 0.91 | 0.91 |
| | **Fine-SAHBiLC** | 0.95 | 0.95 | 0.94 | 0.95 |
| **Hindi** | **LSTM-CNN-CRF** | 0.97 | 0.96 | 0.95 | 0.95 |
| | **SAHBiLC** | 0.98 | 0.98 | 0.98 | 0.98 |

The SAHBiLC model for Bhojpuri can accurately predict on the less frequent and most frequent tags, while the Fine-SAHBiLC model performs prediction on less frequent and most frequent tags of Maithili (except for UT, or utterance tag) and Magahi (except CL, UT, UNK) thoroughly, as shown in Figure 3.4. Apart from this, Figure 3.5 shows that the tags with below 50% accuracy in terms of F-score have degraded performance in Bhojpuri after successfully applying the Fine-SAHBiLC model. In contrast, Maithili and Magahi

have attained an improvement of 50% for INJ and 30% for INJ, INTF, respectively. This may be due to the fact that Bhojpuri is close to Hindi than the other two languages.

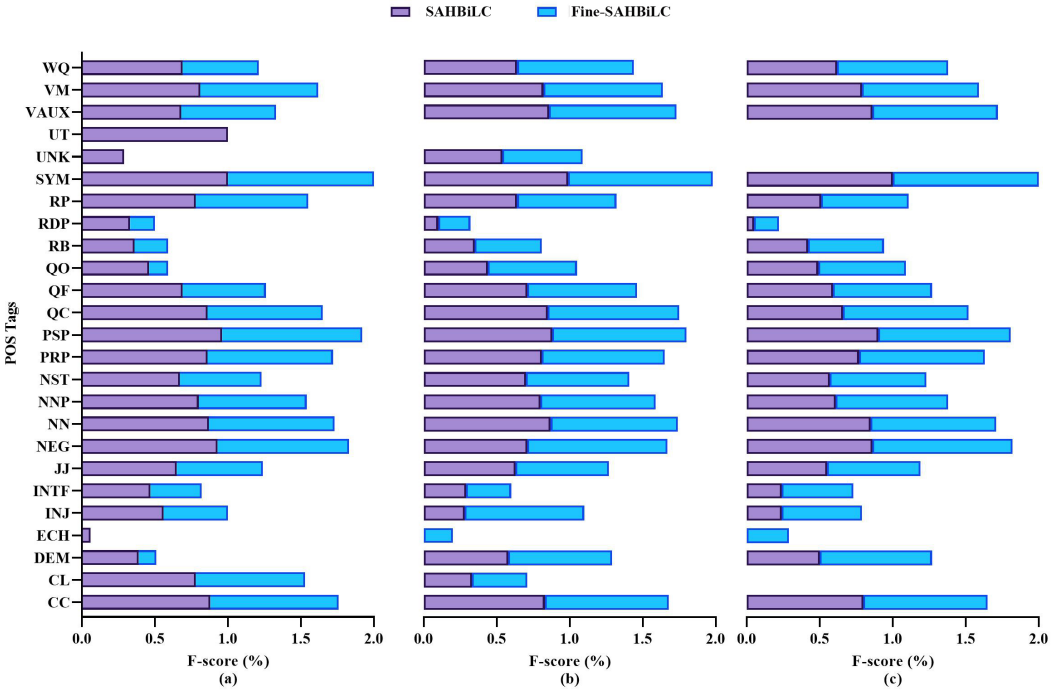


FIGURE 3.4: F-scores of POS tagging after applying SAHBiLC model and Fine-SAHBiLC model on (a) Bhojpuri and (b) Maithili

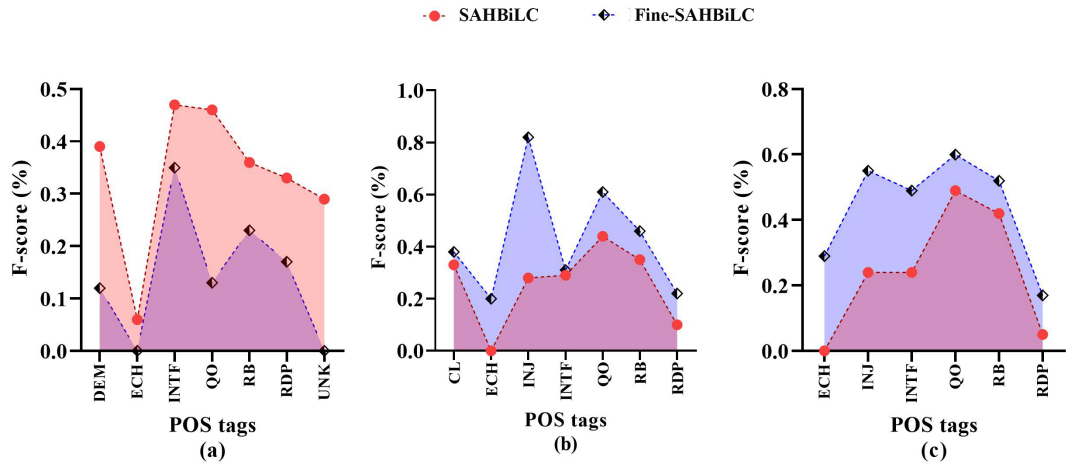


FIGURE 3.5: Most affected POS tags after applying SAHBiLC model and Fine-SAHBiLC model on (a) Bhojpuri, (b) Maithili and (c) Magahi

Furthermore, the less frequent (except VGNN) and the most frequent chunk tags in Bhojpuri attained improvement after applying the SAHBiLC model. Maithili attained the increment by 50% on less frequent and 5% on most frequent chunk tags after using Fine-SAHBiLC, as shown in Figure 3.6. The SAHBiLC model on Hindi improved the result compared to LSTM-CNN-CRF and machine learning techniques for POS tagging. Whereas the Chunking result is a bit low compared to CRF.

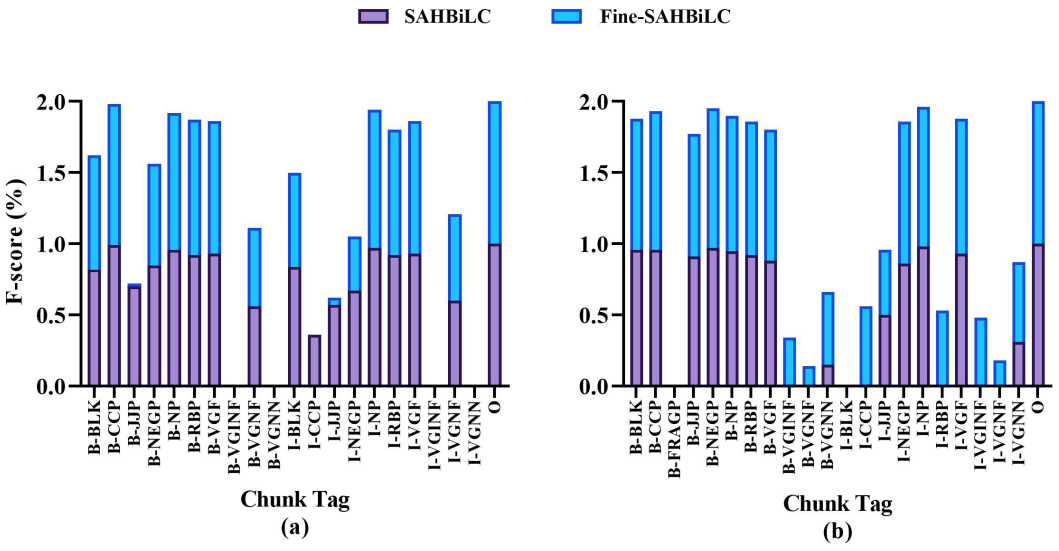


FIGURE 3.6: The F-scores of chunk tagging after applying SAHBiLC model and Fine-SAHBiLC model on (a) Bhojpuri and (b) Maithili

The performance for POS tagging has indeed turned out to be better in the case of deep learning techniques than traditional machine learning techniques for the rest of the languages. Bhojpuri has a higher amount of data as compared to both Maithili and Magahi. An effective solution is provided by employing data from a related task when a specific target task dataset is scarce. However, when shifting knowledge from less relevant data, if it reduces the performance of the target task, it is described as a negative transfer. SAHBiLC and CRF model turn out to be more accurate than the Fine-SAHBiLC due to negative transfer for all POS tags on Bhojpuri. On the other hand, Fine-SAHBiLC attains better performance in POS tagging for both Maithili and Magahi. In the case of Chunking, Fine-SAHBiLC degrades the overall accuracy for Bhojpuri. Therefore, CRF

is a better approach to Bhojpuri and Fine-SAHBiLC achieves better accuracy than other techniques for Maithili.

#### 3.4.4.1 Error analysis

There are many challenges while annotating the corpus of these low resourced languages, as we discussed in Mundotiya et al. [168]. In spite of being labeled as dialects or varieties of Hindi, morphological constructions of Bhojpuri, Maithili and Magahi considerably differ from Hindi. Bhojpuri, Maithili and Magahi are partially synthetic languages. Thus, the use of embedded case markers, emphatic markers, classifiers, determiners, etc. is frequent in these languages. These linguistic idiosyncrasies and lexical ambiguity create challenges for machine learning and are responsible for many problems in annotation as well as in prediction by the algorithms.

In the next section, we discuss some such cases. In the examples in the following section, the tags assigned by human annotator are written in ( ) brackets and tags assigned bythe machine are written in [ ] brackets. As some of the examples show, the machine sometimes gave the correct tag even when the human annotator had erred in manual annotation, as noticed in the test data. Still, there were fewer such cases than the ones where the machine made an error.

**Verb fused with negation markers-**

In general, negative markers occur separately in these languages but in some cases they get fused with verb. In the BIS tagset, we do not have a separate tag for this subcategory of 'Negative' markers. In any case, since they are functional words or morphemes used with the head word, the POS tag is given based on the head word. Sometimes this leads to faulty annotation. For example:

**Bhojpuri:** kavano $(PRP)[PRP]$ bAwa $(NN)[NN]$ naiKe $(NEG)[VM]$ . $(SYM)[SYM]$

**Hindi Translation:** koI bAwa nahIM hE .

**English Translation:** It doesn't matter.

**Magahi:** eka $(QC)[QC]$ azjurI $(NN)[NN]$ ParahI-PutahA $(NN)[NN]$ ke$(PSP)[PSP]$ neMvAnna $(VM)[VM]$ naz $(RP)[PSP]$ .$(SYM)[SYM]$

**Hindi Translation:** eka azjali (BI) ParuhI-Buje xAne navAnna nahIM hEM .

**English Translation:** There is not even a handful of roasted puffed-rice grains of the new harvest (to eat).

In the first example from Bhojpuri, the word *naIKe* is assigned the tag NEG (negation marker) by the annotator. However, in the sentence, it functions as a verb, and the machine has correctly tagged it as VM (main verb). In the second example from Magahi, the word *naz* similarly functions as a verb (VM), but the annotator has tagged it as a particle (RP), while the machine has tagged it as post-position (PSP). The forms of both words are like negation markers. In Bhojpuri, the machine correctly identifies them, even when the annotator is wrong, perhaps because there is significantly more training data for Bhojpuri than for Magahi. Magahi is also morphological more complex. Apart from that, there is no obvious verb morpheme in the second example, whereas it is in the first example.

**Embedded case markers-**

In these languages case markers are usually separate from the head word, but in many cases they get merged with nominals and pronominals. This is especially true of locative, instrumental and genitive case markers. This construction feature also creates challenges for machine learning. For example:

- **Nominals**
  - **Bhojpuri:** sAzJe $(NN)$*[RB]* suwa $(VM)$*[NN]* jAilAz $(VAUX)$*[VM]* . $(SYM)$ $[SYM]$
    **Hindi Translation:** SAma ko hI so jAwe hEM .
    **English Translation:** (He/She honorific, or they, usually)[3] Fall asleep in the evening.
  - **Maithili:** o $(PRP)[PRP]$ gAmakez $(NN)[NN]$ ekawAka $(JJ)$*[NN]* sUwrame $(NN)[NN]$ banhane $(VM)[VM]$ CaWi $(VAUX)[VAUX]$ . $(SYM)[SYM]$
    **Hindi Translation:** unhoMne gAzva ko ekawA ke sUwra meM bAzXA WA .
    **English Translation:** [He/She honorific, or they] tied the village with the thread of unity.

[3] *Note*: In English translations, the ellipsis is denoted with parenthesis, whereas alternative word translations are denoted by square brackets.

– **Magahi:** rAwe $(NST)$*[JJ]* iskUla $(NN)[NN]$ ke $(PSP)[PSP]$ sainaboda $(NN)$ $[NN]$ jagaha-jagaha $(NN)$*[NST]* se $(PSP)[PSP]$ lataka $(VM)[VM]$ gela $(VAUX)$ $[VAUX]$ hala $(VAUX)$ $[VAUX]$ . $(SYM)[SYM]$
**Hindi Translation:** rAwa meM skUla kA sAinaborda jagaha-jagaha se lataka gayA WA .
**English Translation:** The same night, the school signboard was hanging from place to place.

- **Pronominals**

– **Bhojpuri:** hamare $(PRP)$*[PSP]* rUpa $(NN)$ $[NN]$ U $(DEM)$*[PRP]* cAroM $(QO)$*[QC]* orI $(NST)$ $[NST]$ nihArelI $(VM)$*[NN]* .$(SYM)[SYM]$
**Hindi Translation:** mere rUpa ko vaha cAroM ora se nihArawI hE .
**English Translation:** She looks at me from all four sides.

In the case of nominals, there is inconsistency in the annotation of Bhojpuri and Magahi sentences. While the words *sAzJe* (in the evening) and *rAwe* (in the night) are tagged by annotators as NN (noun) and NST (relational noun), respectively. The machine tags the first as an adverb (RB) and second as an adjective (JJ), which are easy mistakes to make in the absence of enough data. The convention with the BIS tagset is to tag such words as NST.

**Diverse realizations of a single token-**

Analogous to other Indian languages, Bhojpuri, Maithili and Magahi languages also have several cases of diverse realization of a single token. These tokens generally have multiple functional and connotative meanings. In the annotated data of the Bhojpuri language the token *t* has the highest 9 tags for different realizations. Similarly, in Magahi, 9 tags are assigned to the token *khaali*, whereas in Maithili, the token *lel* has been assigned 10 tags for diverse realizations. The list of such tokens is long in all these three languages.

**Homophonous forms-**

Bhojpuri, Maithili and Magahi also have homophonous words in abundance, like many other Indian languages. These words look similar but their POS tags are varied which may cause confusion for the annotation task, especially for machine learning. For example:

**Bhojpuri:** Baila $(VM)[VM]$ biAha $(NN)$*[NST]* mora $(PRP)[PRP]$ karaba $(VM)[VM]$

kA $(WQ)$*[PSP]* ? $(SYM)[SYM]$

**Hindi Translation:** ho gayA vivAha aba karUz kyA ?

**English Translation:** I got married, now what?

sonala $(NNP)[NNP]$ sarakArI $(NN)$*[NNP]* gavAha $(NN)[NN]$ bana $(VM)[VM]$ gailI $(VM)$*[VAUX]* .$(SYM)[SYM]$

**Hindi Translation:** sonala sarakArI gavAha bana gaI .

**English Translation:** Sonal became a witness from government side.

AnhIM $(NN)[NN]$ me $(PSP)[PSP]$ Aam $(NN)$*[JJ]* niyara $(PRP)[PRP]$ Cappara $(NN)$ $[NN]$ cuawA $(VM)$ *[NN]* .$(SYM)[SYM]$

**Hindi Translation:** AzXI meM Ama ke samAna Cappara cU rahA hE .

**English Translation:** The shed is dripping like mangoes in the storm.

In these cases, PSP, VM and JJ are the most frequent tags respectively for *kA*, *bana* and *Ama*, which led to faulty annotation by the machine. These are some examples of confusion created by homophonous words for automatic annotation.

**Classifiers-**

Similar to Bengali and Oriya, the Bhojpuri, Magahi and Maithili languages also very often use classifiers with numerals, whereas Hindi and its other 'dialects' (or 'sub-languages') are classifier-less languages. Classifier markers in these languages are: *To*, *Te*, *go* and *Ke* etc. This feature often creates problems for machine learning. For example:

**Bhojpuri:** wIna $(QC)[QC]$ cAra $(QC)[QC]$ go $(CL)$*[RP]* Gara $(NN)[NN]$ A $(CC)[CC]$ xalAna $(NN)[NN]$ banAvala $(VM)[VM]$ rahe $(VM)$*[VAUX]* .$(SYM)$ $[SYM]$

**Hindi Translation:** wIna cAra Gara Ora xalAna banAe hue We .

**English Translation:** Three or four houses and verandahs were built.

**Maithili:** eka $(QC)$*[RP]* tA $(CL)$*[RP]* cunAva $(JJ)$*[NN]* karmacArIka $(NN)[NN]$ niXana $(NN)$*[VM]* seho $(RB)$ *[RP]* BaZ $(VM)[VM]$ gelanhi $(VAUX)[VAUX]$ .$(SYM)$ $[SYM]$

**Hindi Translation:** eka cunAva karmacArI kA niXana BI ho gayA .

**English Translation:** An election worker also died.

In these example sentences, the machine assigned an incorrect tag (RP) to the classifiers

of these languages, whereas the CL tag assigned by human annotators is correct. It is easy to mistake these classifiers for particles.

**Amalgamated emphatic expressions-**

Like other languages of the Magadhi group, Bhojpuri, Maithili and Magahi languages also have the feature of merged emphatic particles with nominals and pronominals in general. This kind of idiosyncrasy, combined with lack of data, makes annotation tasks difficult for machines as well. For example:

**Bhojpuri:** u $(PRP)[PRP]$ abbe $(NST)$*[RP]* Gare $(NN)$ $[NN]$ Aila $(VM)[VM]$ bA $(VAUX)[VAUX]$ . $(SYM)$ $[SYM]$

**Hindi Translation:** vo aBi hI Gara AyA hE .

**English Translation:** He came home just now.

**Maithili:** hamahuz $(PRP)$*[RP]* wapAkasaz $(RB)[RB]$ haz $(NN)$*[RP]* kahi $(VM)$ $[VM]$ xelahuz $(VAUX)[VAUX]$ .$(SYM)[SYM]$

**Hindi Translation:** mEMne BI wapAka se hAz kaha xiyA .

**English Translation:** I also said yes promptly.

**Magahi:** aisahIM $(DEM)$*[RB]* wo $(RP)$*[PSP]* bahanoI $(NN)[NN]$ rusala $(VM)$ $[VM]$ haWina $(VAUX)[VAUX]$ . $(SYM)[SYM]$

**Hindi Translation:** Ese hI wo bahanoI rUTe We .

**English Translation:** Brother-in-law was angry without any reason.

In Bhojpuri, the emphasized temporal adverb *abbe* is tagged by the annotator as a relational noun (NST), while the machine mistakes it for a particle (RP). This indicates there are still problems with the annotated data. In Maithili, the emphasized pronoun *hamahuz* is correctly tagged by the annotator as a pronoun (PRP), but the machine tags it as a particle (RP), which is perplexing. In Magahi, the demonstrative *aisahIM* is wrongly tagged by the machine as a particle again. It seems that due to the frequent use of particles in these languages, words are often wrongly identified as particles.

**Interrogative markers-**

The machine frequently assigned wrong tags to different interrogative markers of Bhojpuri, Maithili and Magahi. This may be because most of these markers are homophonous forms. For example:

**Bhojpuri:** I $(DEM)$*[PRP]* BojapurI $(NNP)$*[JJ]* pawrikA $(NN)[NN]$ ke $(PSP)[PSP]$

kaise $(WQ)$*[PRP]* paDZaba $(VM)$*[NN]* ? $(SYM)[SYM]$

**Hindi Translation:** isa BojapurI pawrikA ko kEse paDUz ?

**English Translation:** How do I read this Bhojpuri magazine?

**Magahi:** xeKahIM $(VM)[VM]$ wa $(CC)$*[RP]* ke $(WQ)$*[RP]* hai $(AUX)[AUX]$ xuariyA $(NN)[NN]$ para $(PSP)[PSP]$ . $(SYM)[SYM]$

**Hindi Translation:** xeKUz wo kOna hE xaravAje para .

**English Translation:** Let me see who is at the door.

**Maithili:** bahasa $(NN)[NN]$ para $(PSP)[PSP]$ kaya $(WQ)$*[PSP]* tA $(RP)[RP]$ kameMta $(NN)[NN]$ Ayala $(VM)[VM]$ aCi $(VAUX)[VAUX]$ ? $(SYM)[SYM]$

**Hindi Translation:** bahasa para kiwane kameMta Ae hEM ?

**English Translation:** How many comments have come on the debate ?

## 3.5 Experiment-II: NER

### 3.5.1 Dataset Description

For NER systems, we considered the BMM corpus [168] of the project on Bhojpuri, Maithili and Magahi to Hindi Machine Translation System under Project Varanasi. From these languages corpus, 16492, 9815 and 5320 sentences have been considered to create the NER annotated data. These sentences have 228373, 157468 and 56190 tokens and 32091, 23338 and 10175 types, for respective languages. After annotation, 12351 named entities in Bhojpuri, 19809 in Maithili and 7152 in Magahi were encountered, as mentioned in Table 3.9.

TABLE 3.9: Language-wise dataset statistics used for annotation of named entities

| **Language** | **#Sentences** | **#Tokens** | **#Types** | **#Entities** | **#Others** |
|---|---|---|---|---|---|
| Bhojpuri | 16492 | 228373 | 32091 | 12351 | 216022 |
| Maithili | 9815 | 157468 | 23338 | 19809 | 137659 |
| Magahi | 5320 | 56190 | 10175 | 7152 | 49038 |

The broad categories of NER tags and their statistics for each language are outlined in Table 3.10. And each category with its statistics of hierarchical entities is summarised in

Table 3.11.

TABLE 3.10: The statistics of annotated for the three broad categories for Bhojpuri, Maithili and Magahi

| **Language** | **ENAMEX** | **NUMEX** | **TIMEX** |
|---|---|---|---|
| Bhojpuri | 10504 | 1152 | 695 |
| Maithili | 15861 | 2214 | 1734 |
| Magahi | 5790 | 725 | 637 |

TABLE 3.11: The statistics of annotated hierarchical entities for **Bhoj**puri, **Mai**thili and **Mag**ahi. The ENAMEX, NUMEX and TIMEX categories contained 11, 4 and 7 hierarchical named entities. 'Other' denotes regular words or tokens which are not named entities.

| **Entities** | **Bhoj** | **Mai** | **Mag** | **Entities** | **Bhoj** | **Mai** | **Mag** |
|---|---|---|---|---|---|---|---|
| **ENAMEX** | | | | **NUMEX** | | | |
| Artifact | 635 | 752 | 638 | Count | 685 | 1797 | 558 |
| Disease | 34 | 9 | 18 | Distance | 14 | 24 | 2 |
| Entertainment | 347 | 532 | 31 | Money | 166 | 131 | 112 |
| Facility | 121 | 784 | 123 | Quantity | 287 | 262 | 53 |
| Location | 985 | 4330 | 763 | **TIMEX** | | | |
| Locomotive | 112 | 157 | 58 | Date | 69 | 48 | 5 |
| Material | 278 | 481 | 379 | Day | 36 | 99 | 169 |
| Organism | 481 | 222 | 566 | Month | 66 | 281 | 52 |
| Organization | 109 | 2081 | 20 | Period | 279 | 491 | 28 |
| Person | 7244 | 6462 | 3145 | Special_Day | 8 | 210 | 1 |
| Plant | 158 | 51 | 49 | Time | 175 | 413 | 337 |
| **OTHER** | | | | Year | 62 | 192 | 45 |
| Other | 216022 | 137659 | 49038 | | | | |

For performing the baseline experiments on the prepared NER dataset of Bhojpuri, Maithili and Magahi, we have used two standard techniques which are known to provide previous state-of-the-art results for NER for other languages along with the SAHBiLC model. These techniques are: Conditional Random Fields (CRF) [124], a machine learning algorithm and LSTM-CNNs-CRF [140], based on a deep learning algorithm. From our study on related work of NER and other sequence labeling tasks of the same languages (in section), a machine algorithm (CRF) yielded comparative results to the Deep Learning method, perhaps because of the lack of sufficient data.

### 3.5.2 Experimental Settings

The dataset was divided into training and testing splits with a ratio of 80-20. While splitting, it was ensured that the testing dataset included all the named entities with a frequency of at least one. The dataset statistics after splitting are shown in Table 3.12. The training strategy and the obtained results have been explained in the following sections.

TABLE 3.12: The dataset sizes for each language. The OOV percentage is calculated by token-type differences between test data and the training data.

| Language | Data-Mode | Sentences | Tokens | Types | OOV (%) |
|---|---|---|---|---|---|
| **Bhojpuri** | Train | 11544 | 160226 | 19642 | 25.50 |
| | Test | 4948 | 68147 | 12449 | |
| **Maithili** | Train | 7849 | 125442 | 15859 | 27.94 |
| | Test | 1966 | 32026 | 7479 | |
| **Magahi** | Train | 4256 | 44833 | 6868 | 22.34 |
| | Test | 1065 | 11357 | 3307 | |

#### 3.5.2.1 CRF model training

There are several implementations of CRF that are publicly available. We have used the CRFsuit [4] implementation with the training algorithm of L-BFGS, executed up to a maximum of 100 iterations. To avoid overfitting and underfitting issues of CRF, C1 and C2 regularization parameters with random search cross-validation have been used for training, where the value of cross-validation is 3, and the number iterations are 50. The current word, the neighboring words with the adjacency of 2, affixes of the current word with a window size of 3, whether the current word and the neighboring words are digits and whether the current word is first, or the last word of a sentence are considered as hand-crafted features for training the CRF model. The optimal values of C1 and C2 are 0.178 and 0.006 for Magahi, 0.440 and 0.018 for Maithili and 0.481 and 0.003 for Bhojpuri as obtained after training.

[4] https://sklearn-crfsuite.readthedocs.io/en/latest/

TABLE 3.13: The value of (hyper-)parameters used for training of the LSTM-CNNs-CRF model

| **(Hyper-)Parameter** | **Value** | **(Hyper-)Parameter** | **Value** |
|---|---|---|---|
| Word Embedding | [100, 200] | Word Hidden | 200 |
| Char. Embedding | [20, 30] | Char. Hidden | [25, 50] |
| Batch Size | 20 | Epochs | 20 |
| Convolution Layer | 4 | Optimizer | SGD |
| Dropout | 0.5 | L2 | 1e-8 |
| Learning Rate | 0.015 | Learning Decay | 0.05 |

#### 3.5.2.2 Deep learning model training

The LSTM-CNNs-CRF model takes input in the form of characters and words to generate word embedding to overcome the scarcity of annotated data. As Deep Learning models are very sensitive to the data size as well as the values of the parameters, it is not guaranteed that the same value of the parameter will provide optimal results for another language. The word embeddings, character embeddings and hidden representations play a vital role in obtaining the best possible results. For our experiments, the sizes of the word embedding size, the character embedding and the hidden representations 100, 20 and 25, respectively for Bhojpuri. Similarly, the values are 100, 30 and 50 for Magahi and 200, 20 and 50 for Maithili. The number of convolutional layers is 4. The model training is performed with the Stochastic Gradient Descent (SGD) optimizer, where the learning rate is 0.015, which decays over the epoch by 0.05 for constraint learning. During training, the L2 regularizer and dropout are also used with the values of 0.5 and 1e$-$8, respectively to prevent model overfitting. The summary of the (hyper-)parameters with their values is mentioned in Table 3.13.

### 3.5.3 Results and Analysis

Bhojpuri has a larger annotated dataset than the remaining two languages. For this language, the obtained results on the validation data for CRF and LSTM-CNNs-CRF are 70.56% and 61.41% as $F_1$-score, respectively. The entity-wise Precision, Recall and $F_1$-score given in Table 3.14 shows that LSTM-CNNs-CRF struggles to learn low-frequency entities such as Day, Disease, Distance, Organization, Special_Day and Year.

In the CRF training, the best transition is obtained from I-Money→ I-Money, B-Organization→I-Organization, B-Period→I-Period, B-Facility→I-Facility and B-Year→I-Year. Similarly the worst transition represents the wrong interpretation as B-Count→I-Person, B-Count→B-Count, I-Person→B-Artifact and B-Artifact→B-Artifact.

TABLE 3.14: NER tag-wise scores obtained by CRF and LSTM-CNNs-CRF for Bhojpuri. The metrics, which are **P**recision, **R**ecall and $\mathbf{F}_1$-score

| **Techniques** | **LSTM-CNNs-CRF** | | | **CRF** | | |
|---|---|---|---|---|---|---|
| **NER-Tag** | **P** | **R** | **F1** | **P** | **R** | **F1** |
| Artifact | 84.91 | 26.16 | 40.00 | 96.77 | 34.88 | 51.28 |
| Disease | 0.00 | 0.00 | 0.00 | 100.00 | 77.78 | 87.50 |
| Entertainment | 77.78 | 6.93 | 12.73 | 84.00 | 20.79 | 33.33 |
| Facility | 25.00 | 10.26 | 14.55 | 36.00 | 23.08 | 28.12 |
| Location | 97.20 | 39.10 | 55.76 | 95.36 | 54.14 | 69.06 |
| Locomotive | 65.00 | 32.50 | 43.33 | 73.91 | 42.50 | 53.97 |
| Material | 77.78 | 7.37 | 13.46 | 91.67 | 23.16 | 36.97 |
| Organism | 100.00 | 7.69 | 14.29 | 100.00 | 18.46 | 31.17 |
| Organization | 0.00 | 0.00 | 0.00 | 100.00 | 18.18 | 30.77 |
| Person | 98.88 | 72.45 | 83.62 | 98.97 | 79.01 | 87.87 |
| Plant | 90.91 | 25.64 | 40.00 | 84.62 | 56.41 | 67.69 |
| Count | 71.43 | 23.26 | 35.09 | 71.62 | 30.81 | 43.09 |
| Distance | 0.00 | 0.00 | 0.00 | 50.00 | 20.00 | 28.57 |
| Money | 100.00 | 6.67 | 12.50 | 88.46 | 38.33 | 53.49 |
| Quantity | 55.56 | 14.08 | 22.47 | 48.78 | 28.17 | 35.71 |
| Date | 100.00 | 5.88 | 11.11 | 100.00 | 23.53 | 38.10 |
| Day | 0.00 | 0.00 | 0.00 | 100.00 | 16.67 | 28.57 |
| Month | 0.00 | 0.00 | 0.00 | 100.00 | 27.78 | 43.48 |
| Period | 100.00 | 14.47 | 25.29 | 86.67 | 17.11 | 28.57 |
| Special_Day | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Time | 100.00 | 28.57 | 44.44 | 100.00 | 26.79 | 42.25 |
| Year | 0.00 | 0.00 | 0.00 | 57.14 | 40.00 | 47.06 |
| **Avg. score** | **91.14** | **50.51** | **61.41** | **93.64** | **59.90** | **70.56** |
| OTHER | 97.39 | 99.46 | 98.41 | 97.94 | 99.08 | 98.51 |
| Avg. score (with OTHER) | 96.15 | 96.84 | 96.25 | 96.59 | 96.99 | 96.73 |

For Maithili, we obtained 73.19% $F_1$-score for CRF and 71.38% for LSTM. The tag-wise scores are listed in Table 3.15.

Some of the remaining entities (Day, Date, Month, Year, Distance, Organism and Plant) have rare intermediates. The optimal transitions from this language's annotated dataset are B-Entertainment→I-Entertainment, B-Facility→I-Facility and B-Organism→I-Organism,

and worst transitions are B-Period→B-Count, B-Location→I-Person and B-Location→B-Period.

TABLE 3.15: NER tag-wise scores obtained for CRF and LSTM-CNNs-CRF for Maithili

| **Techniques** | **LSTM-CNNs-CRF** | | | **CRF** | | |
|---|---|---|---|---|---|---|
| **NER-Tag** | **P** | **R** | **F1** | **P** | **R** | **F1** |
| Artifact | 83.33 | 28.09 | 42.02 | 77.78 | 31.46 | 44.80 |
| Disease | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Entertainment | 70.18 | 68.97 | 69.57 | 87.80 | 62.07 | 72.73 |
| Facility | 81.25 | 43.82 | 56.93 | 80.77 | 47.19 | 59.57 |
| Location | 93.88 | 66.72 | 78.01 | 91.96 | 68.90 | 78.78 |
| Locomotive | 100.00 | 15.15 | 26.32 | 93.75 | 45.45 | 61.22 |
| Material | 88.57 | 39.74 | 54.87 | 85.71 | 38.46 | 53.10 |
| Organism | 100.00 | 5.41 | 10.26 | 66.67 | 5.41 | 10.00 |
| Organization | 92.31 | 57.93 | 71.19 | 93.64 | 55.86 | 69.98 |
| Person | 97.08 | 70.89 | 81.94 | 97.11 | 75.62 | 85.03 |
| Plant | 100.00 | 45.45 | 62.50 | 100.00 | 36.36 | 53.33 |
| Count | 83.57 | 72.65 | 77.73 | 87.69 | 69.80 | 77.73 |
| Distance | 0.00 | 0.00 | 0.00 | 50.00 | 33.33 | 40.00 |
| Money | 66.67 | 15.38 | 25.00 | 62.50 | 38.46 | 47.62 |
| Quantity | 77.78 | 11.11 | 19.44 | 83.33 | 7.94 | 14.49 |
| Date | 100.00 | 20.00 | 33.33 | 100.00 | 60.00 | 75.00 |
| Day | 69.23 | 45.00 | 54.55 | 85.71 | 30.00 | 44.44 |
| Month | 100.00 | 88.37 | 93.83 | 97.44 | 88.37 | 92.68 |
| Period | 92.16 | 63.51 | 75.20 | 90.32 | 75.68 | 82.35 |
| Special_Day | 100.00 | 62.07 | 76.60 | 95.00 | 65.52 | 77.55 |
| Time | 83.33 | 24.59 | 37.97 | 88.89 | 26.23 | 40.51 |
| Year | 100.00 | 66.67 | 80.00 | 100.00 | 59.26 | 74.42 |
| **Avg. score** | **91.60** | **60.65** | **71.38** | **91.53** | **62.86** | **73.19** |
| OTHER | 95.27 | 98.61 | 96.91 | 95.52 | 98.33 | 96.90 |
| Avg. score (with OTHER) | 93.34 | 93.85 | 93.33 | 93.32 | 93.87 | 93.33 |

For Magahi, we obtained $F_1$-scores of 84.18% and 86.39% for CRF and LSTM-CNNs-CRF, respectively. The tag-wise scores are listed in Table 3.16. As, Magahi has smaller annotated data as well as a high number of low-frequency entities such as Date, Distance and Special_Day [5]. Moreover, the LSTM-CNNs-CRF technique suffers from learning those entities.

[5]The Special_Day entity has one frequency; hence it has not appeared in the validation.

Although, few entities such as Disease, Entertainment, Organisation, Plant, Year and Distance have metric scores around 0 in the intermediate tag due to less frequency in the annotated dataset. Most of such entities are not longer than a single token; hence intermediates of these tags are rare. During an evaluation of the model, we found that the optimal transitions between B-Quatity→I-Quantity, B-Plant→I-Plant and I-Quantity→I-Material, where B-Person→B-Facility, B-Day→B-Person and OTHER→I-Quantity are worst transitions which indicate wrong annotations.

TABLE 3.16: NER tag-wise scores obtained for CRF and LSTM-CNNs-CRF for Magahi

| **Techniques** | **LSTM-CNNs-CRF** | | | **CRF** | | |
|---|---|---|---|---|---|---|
| **NER-Tag** | **P** | **R** | **F1** | **P** | **R** | **F1** |
| Artifact | 100.00 | 58.89 | 74.13 | 100.00 | 54.44 | 70.50 |
| Disease | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Entertainment | 100.00 | 40.00 | 57.14 | 100.00 | 40.00 | 57.14 |
| Facility | 100.00 | 72.22 | 83.87 | 100.00 | 72.22 | 83.87 |
| Location | 96.83 | 70.11 | 81.33 | 98.33 | 67.82 | 80.27 |
| Locomotive | 80.00 | 40.00 | 53.33 | 75.00 | 30.00 | 42.86 |
| Material | 94.29 | 91.67 | 92.96 | 96.77 | 83.33 | 89.55 |
| Organism | 97.96 | 77.42 | 86.49 | 100.00 | 75.81 | 86.24 |
| Organization | 100.00 | 25.00 | 40.00 | 100.00 | 50.00 | 66.67 |
| Person | 100.00 | 88.29 | 93.78 | 97.59 | 84.98 | 90.85 |
| Plant | 100.00 | 62.50 | 76.92 | 100.00 | 62.50 | 76.92 |
| Count | 95.65 | 85.71 | 90.41 | 96.67 | 75.32 | 84.67 |
| Distance | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Money | 100.00 | 95.00 | 97.44 | 100.00 | 95.00 | 97.44 |
| Quantity | 100.00 | 54.55 | 70.59 | 100.00 | 54.55 | 70.59 |
| Date | 0.00 | 0.00 | 0.00 | 100.00 | 66.67 | 80.00 |
| Day | 94.44 | 89.47 | 91.89 | 94.44 | 89.47 | 91.89 |
| Month | 100.00 | 72.73 | 84.21 | 100.00 | 72.73 | 84.21 |
| Period | 100.00 | 33.33 | 50.00 | 100.00 | 33.33 | 50.00 |
| Special_Day | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Time | 100.00 | 72.97 | 84.37 | 100.00 | 70.27 | 82.54 |
| Year | 100.00 | 75.00 | 85.71 | 100.00 | 62.50 | 76.92 |
| **Avg. score** | **97.82** | **78.42** | **86.39** | **97.67** | **75.00** | **84.18** |
| OTHER | 96.92 | 98.28 | 97.60 | 96.44 | 98.43 | 97.42 |
| Avg. score (with OTHER) | 95.44 | 95.62 | 95.44 | 95.11 | 95.29 | 95.04 |

#### 3.5.3.1 Effect of epoch in LSTM-CNNs-CRF model

Low resource languages have less annotated data; the deep learning model memorizes the data points over the high number of epochs during training, which leads to inconsistency in model performance. Hence, the deep learning model's accuracy evaluated over the epochs on the test dataset, is shown in Figure 3.7.

During the training of the model, accuracy for Bhojpuri on epoch is higher than the remaining two languages. However, there has been a slight improvement in accuracy with the number of epochs, mainly due to the 'OTHER' entity. The 'OTHER' entity covers 94.57% of the overall tokens of Bhojpuri, which increases the accuracy of the model at the beginning.

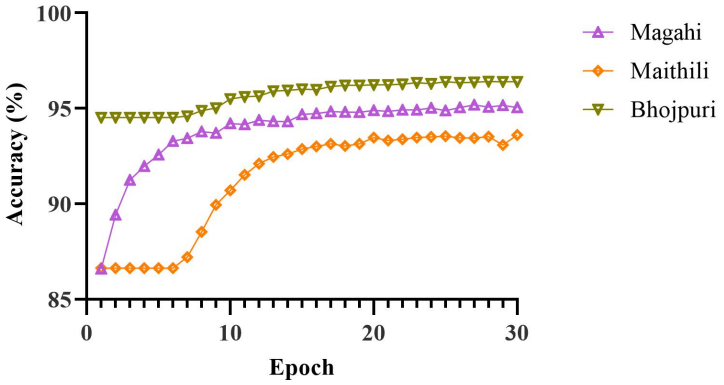


FIGURE 3.7: Effect of epochs on LSTM-CNNs-CRF model's accuracy on Bhojpuri, Maithili and Magahi.

#### 3.5.3.2 Error analysis

The following figures give the confusion matrices for the reported prediction experiments. While plotting a confusion matrix, we have ignored NEs that: (i) have both actual and predicted counts as zeros, (ii) or are all predicted correctly. For example, Year, Special_Day, Distance and Disease in Bhojpuri, and Distance and Special_Day in Magahi belong to the case (i), whereas Time and Period in Bhojpuri, and Month, Time, Year, Person, Facility, Organization and Entertainment in Magahi, and Month in Maithili are all predicted correctly by the LSTM-CNNs-CRF model.

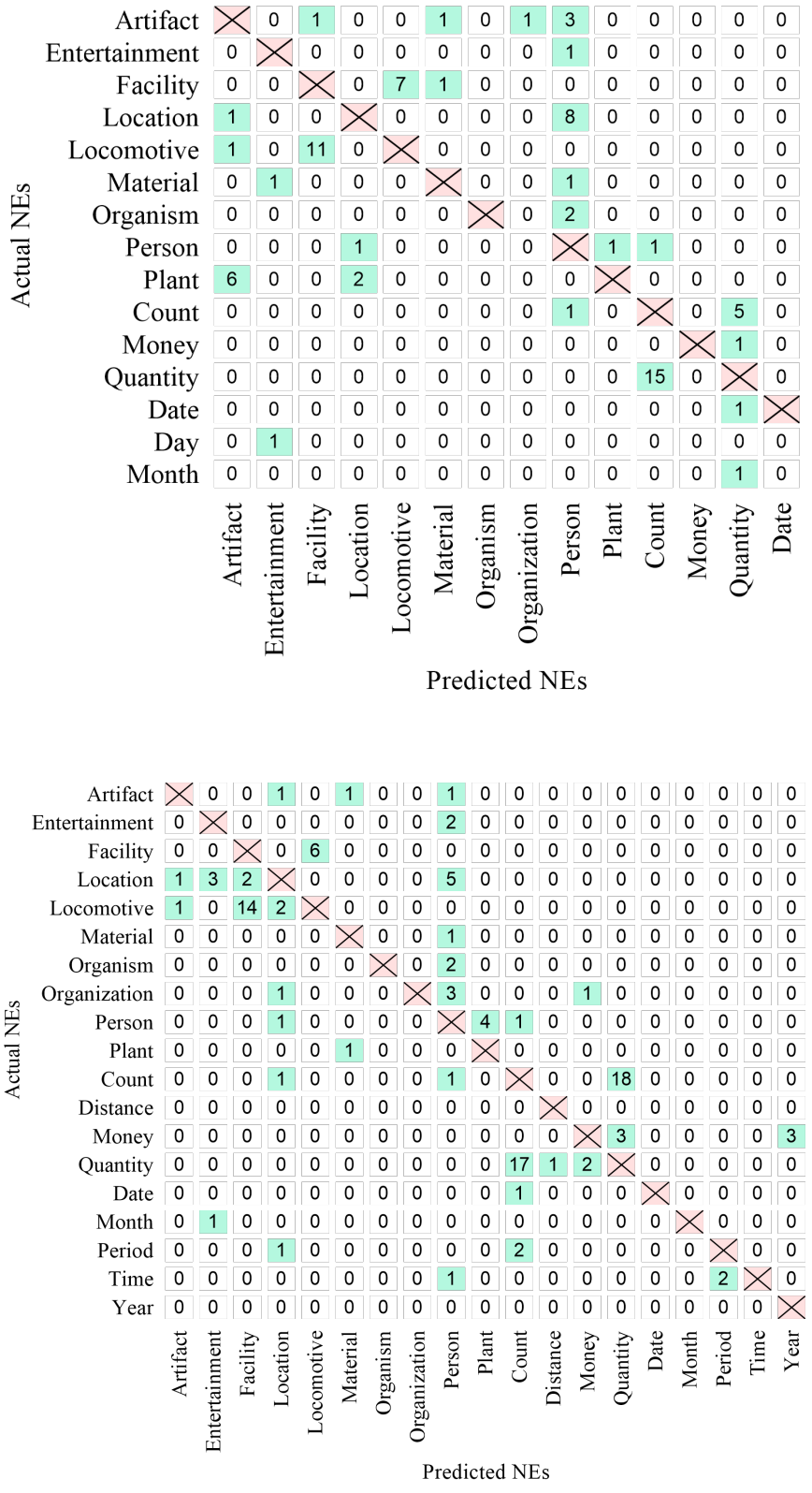


FIGURE 3.8: Confusion matrix for Bhojpuri for the LSTM-CNNs-CRF (above) and CRF (below) models; The × refers to correctly prediction

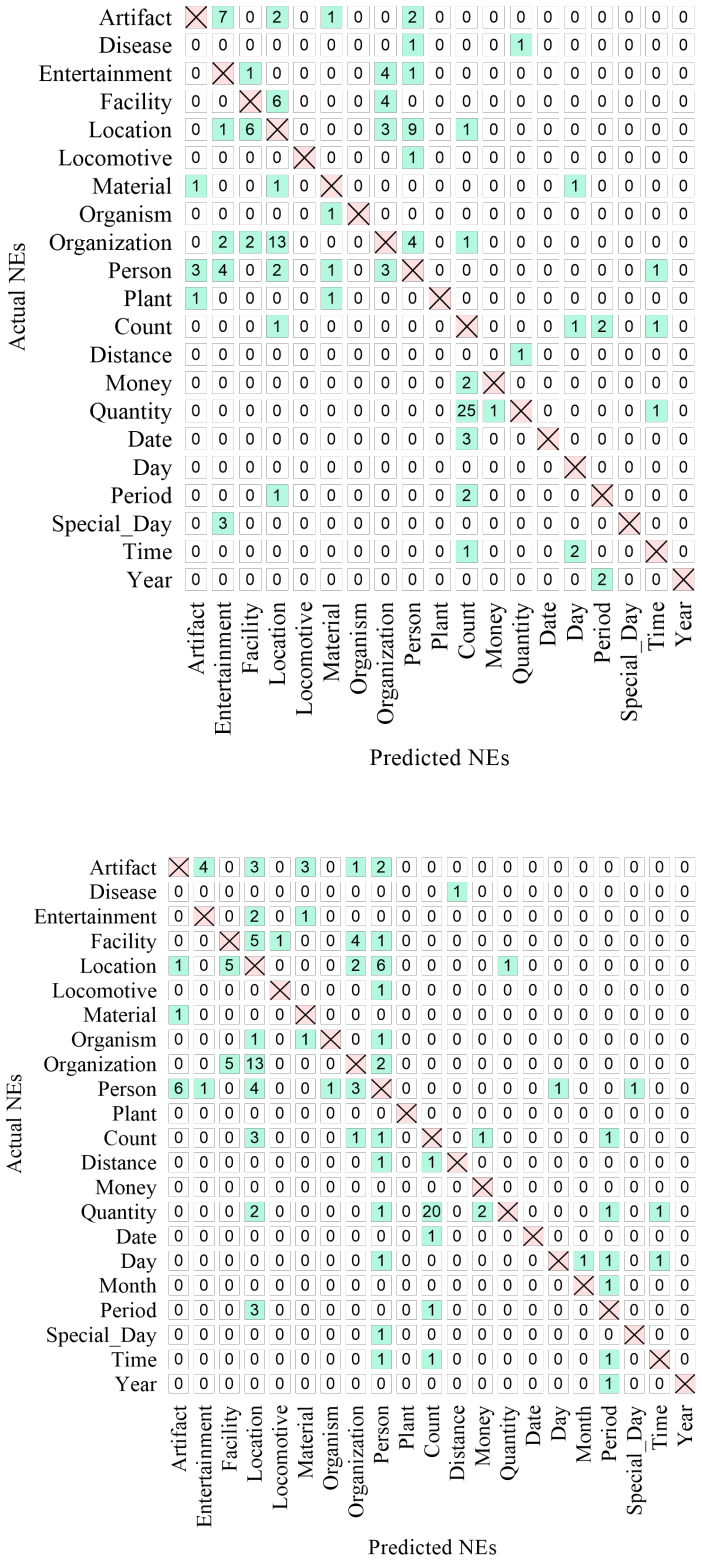


FIGURE 3.9: Confusion matrix for Maithili for the LSTM-CNNs-CRF (above) and CRF (below) models; The × refers to correctly prediction

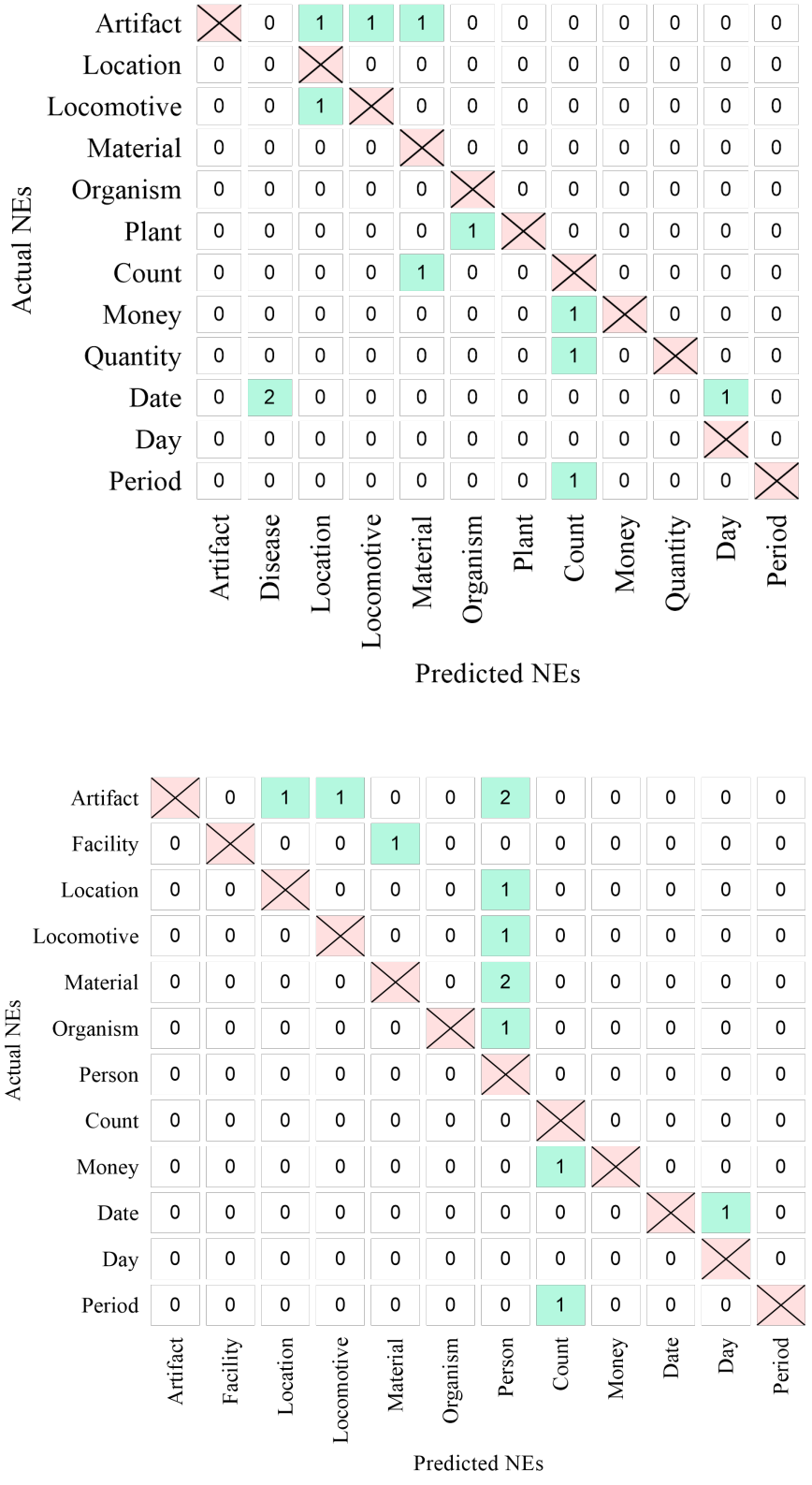


FIGURE 3.10: Confusion matrix for Magahi for the LSTM-CNNs-CRF (above) and CRF (below) models; The × refers to correctly prediction

## 3.6 Summary

The present chapter provides the baseline tools for POS tagging, Chunking and NER using seed annotated datasets of three low resource Indo-Aryan languages: Bhojpuri, Maithili and Magahi. There were no tools available for these languages. Hence, state-of-the-art machine learning and deep learning approaches have been used and evaluated the obtained results. Apart from that, a novel deep learning architecture based on self-attention has been proposed for POS tagging and Chunking, named Self Attention based Hierarchical Bi-LSTM CRF (SAHBiLC). The SAHBiLC provides competitive results over the baselines. Later on, found that the cross-lingual feature transfer helped to further improve the results for the SAHBiLC. Later, improve POS tagger and Chunker performance for these languages, using Hindi as a high-resource language through cross-lingual (learned) feature transfer and discuss the case of negative transfer. On these datasets for POS tagging and Chunking, CRF provided the best results as the same technique used for the NER and compared with the deep learning based baseline technique.