Highly Efficient Parts of Speech Tagging in Low Resource Languages with Improved Hidden Markov Model and Deep Learning

Diganta Baishya, Rupam Baruah
Department of Computer Science and Engineering
Jorhat Engineering College
Jorhat, India

***Abstract*—Over the years, many different algorithms are proposed to improve the accuracy of the automatic parts of speech tagging. High accuracy of parts of speech tagging is very important for any NLP application. Powerful models like The Hidden Markov Model (HMM), used for this purpose require a huge amount of training data and are also less accurate to detect unknown (untrained) words. Most of the languages in this world lack enough resources in the computable form to be used during training such models. NLP applications for such languages also encounter many unknown words during execution. This results in a low accuracy rate. Improving accuracy for such low-resource languages is an open problem. In this paper, one stochastic method and a deep learning model are proposed to improve accuracy for such languages. The proposed language-independent methods improve unknown word accuracy and overall accuracy with a low amount of training data. At first, bigrams and trigrams of characters that are already part of training samples are used to calculate the maximum likelihood for tagging unknown words using the Viterbi algorithm and HMM. With training datasets below the size of 10K, an improvement of 12% to 14% accuracy has been achieved. Next, a deep neural network model is also proposed to work with a very low amount of training data. It is based on word level, character level, character bigram level, and character trigram level representations to perform parts of speech tagging with less amount of available training data. The model improves the overall accuracy of the tagger along with improving accuracy for unknown words. Results for "English" and a low resource Indian Language "Assamese" are discussed in detail. Performance is better than many state-of-the-art techniques for low resource language. The method is generic and can be used with any language with very less amount of training data.**

Keywords*—Hidden markov models; viterbi algorithm; machine learning; deep learning; text processing; low resource language; unknown words; parts of speech tagging

I. Introduction

Parts of speech tagging can be viewed as the problem of word classification. Each such class contains words having some common properties regarding their usage in the sentences. For example, the English language contains the following parts of speech: 'noun', 'pronoun', 'verb', 'adjective', 'preposition', 'conjunction', and 'interjection'. The meaning of a sentence and its grammatical correctness depends on the kind of parts of speech being used in the sentence. The accurate parts of speech tagging can only determine the correct interpretation of the text. Any Language Processing task, therefore, depends heavily on the accuracy of tagging. Information in natural languages like parts of speech can be helpful in many language-related tasks. But, most of the developments of natural language processing are observed for a few dominant languages spoken widely in the world. This is because of a lack of extensive research and the non-availability of computable resources for other languages. It is therefore very important to identify the key factors that affect the accuracy and to make proper use of them so that such languages can also benefit from the advancements of natural language processing techniques. The concern is to find the innovative idea to improve accuracy for languages with fewer amounts of data available for use. With the availability of methods to work well with low training, we can develop NLP applications for languages that are poor in terms of computable resources. Also, it is important to design systems that can be used across any language so that the benefit can be transferred to any language.

The Hidden Markov model is one of the most popular stochastic models used for natural language processing. The Viterbi algorithm uses Hidden Markov Model to find the most likely sequence of hidden states. It is used to derive an observed sequence. Thus, the algorithm can be used to predict parts of the speech tags of a sequence of words [1]. The model is trained with a large amount of already tagged training data. Such a model does not work properly when training data is less in volume. The model fails to learn the behaviour due to less training and also due to unknown words that were not used during training. The accuracy is further low in the cases of languages for which adequate training data are not available for training the model. A lot of research has been carried out to overcome this by modifying the hidden Markov model for unknown words but there is a lot of scope for improvement. Most of the research has been carried out to improve the Hidden Markov Model and Viterbi Algorithm using a large training dataset and with the imposition of some rules. Usage of huge training datasets limits the scope of the methods only to those languages that are very rich in computable resources. Also, rule-based methods limit the scope only to the specific language in context because rules are language-dependent. Hence, a language that is not studied in many details cannot get the benefit of rules-based NLP methods. This paper describes

two specific works to improve performance automatic parts of speech tagging for languages with low quantity of resources in computable form. At first, we introduce a new method to calculate the probability for unknown words. We use bigram and trigram of characters for this purpose and we have also made modifications to the Viterbi algorithm accordingly. The bigram and trigram combinations of characters are used as resources during training for the Hidden Markov model. Bigram and trigram of characters help to model unknown words and also to improve recognition for untrained words. Improving accuracy for unknown words improves the performance for languages with fewer amounts of computable resources available for training. Experiments have been conducted in English languages. We have also tested the same with a small set of Assamese, a low resource language spoken widely in North East India. We have reported results that show considerable improvement. The concept is generic and can be used with any language. This is helpful especially for languages with very less amount of computable resources. Such languages cannot be trained with a huge amount of vocabulary due to lack of data and it results in many unknown words being encountered while testing the system.

Next, a deep learning method is proposed to improve recognition for words using bigram and trigram of characters. Machine Learning and deep learning models are very much popular these days. This kind of model learns the behaviour of the system after being trained with enough labeled datasets. The model learns the association of training datasets and applies it to the real data with good accuracy. Such a model needs to be trained with a huge amount of training data to learn the behaviour well. Challenge is to make the system learn from the least amount of available data so that it works for low-resource languages. We have therefore discussed the proposed deep learning architecture that is capable of classifying the words into defined parts of speech with considerable accuracy. It is based on word level, character level, character bigrams, and character trigram level representation.

The rest of the paper is organized as follows: Section II next describes some of the earlier works and popular methods used for automatic parts of speech tagging, the Hidden Markov Model, parts of speech tagging related to low resource languages, and machine learning methods. Section III describes the research objective in brief. Section IV describes the proposed methods and the data sets used for the purpose. Section V outlines the experiments and results. Section VI discusses the outcomes and Section VII concludes with the future scope of improvements. The reference section outlines the references used in the paper.

II. Earlier Work

A. Works Related to Parts of Speech Tagging

The automatic parts of speech tagging is one of the key areas of research since people started working on processing natural languages. Rule-based methods, stochastic methods, and transformation-based learning approaches are the most widely used supervised techniques for parts of speech tagging. The stochastic or probabilistic methods use a training set and calculate the probability of a word belonging to all possible tags. Based on this calculation, it then assigns the tag with the highest value of probability. As outlined by Martinez (2011), one of the popular methods involved in POS tagging is the Rule-Based Method which is based on a set of rules set by humans [2]. However, it requires too much manual intervention and also requires in-depth knowledge of grammatical rules that varies from language to language. Transformation Based Learning is also used recently to automatically tag POS where the rules are learned from an initially annotated corpus. This method requires a huge amount of training to make the system learn and provide accurate results. The most popular method for POS tagging is Markov Model Taggers that are based Hidden Markov Model. It works on statistical methods to find the best possible sequence of tags out of the possible tag sequences. It consists of three components: outputs, transitions, and states where states represent the tags in case of POS tagging. Maximum Entropy Methods, Support Vector Machines, Neural networks, Decision trees, etc. are some of the other methods used for this purpose. Accuracy can be obtained above 95%, but the model needs to be trained with a huge training dataset [2]. Among the early works, Janas [3] in 1977 proposed a two-step method based on knowledge of linguistic regularities for English texts. He used a large corpus to get 84% of words tagged correctly. Research into parts of speech tagging for languages other than English has also progressed a lot recently. Fernando et. al (2016) recently presented a Support Vector Machine-based Part-Of-Speech tagger for the Sinhala language [4]. Application of Hidden Markov Models based tagger for the language is far behind as stated by the authors. They reported an overall accuracy of 84.68%, and unknown word accuracy of 59.86%. Better use of techniques like the hidden Markov model will be helpful to improve accuracy for the Sinhala language. But unavailability of a huge corpus like that of English is a concern to do so. Hyun-Je Song and Seong-Bae Parkhave (2020) have recently addressed two of tagging for Korean Language problems using a two-step mechanism [5]. Udomcharoenchaikit, Boonkwan, and Vateekul (2020) have introduced an evaluation scheme of Sequential Tagging Methods based on an example-based system using known spelling errors for the Thai language [6].

B. Related Works using Hidden Markov Model

The Hidden Markov Model is the most popular stochastic method for automatic parts of speech tagging. Cutting et al. described [1] some initial works on automatic parts of speech tagging based on the Hidden Markov Model (HMM). They reported an accuracy of 96% using Brown corpus [7], developed by Francis et al. (1979) for English. Cing et. al. (2019) presented a comparison of using HMM only, and HMM with morphological analysis for parts of speech tagging of Myanmar Language. Morphological rules are used to improve performance for unknown words. They have stated in conclusion that using only HMM for a small dataset has no scope at all. A large training dataset or rule-based method in combination with HMM is the only solution [8]. Jurgen et al. (2011) used infinite HMM for parts of speech tagging in an unsupervised manner [9]. Myint et al. [10] proposed to use lexical information with HMM for the Myanmar language as HMM is only capable of using in contexts, not lexical information. They have therefore proposed Lexicalized Hidden Markov Models (L-HMMs) for improving recognition.

Thorsten Brants [11] reported that a Markov model-based tagger performs at least as well as other approaches, including the Maximum Entropy framework. He has used some rules on top of HMM for unknown words and found an accuracy of up to 81%. Ferran PLA and Antonio Molina [12] applied Lexicalized Hidden Markov Models and reported improvement of accuracy for part-of-speech tagging. They reported 6% improvement for unknown words. Recently Tham et. al. (2020) have reported the usage of hybrid POS tagger for the Khasi language. A tagger is developed using the Hidden Markov Model (HMM). It is then integrated with conditional random fields (CRF) rules. The errors obtained from the first tagger are used to improve accuracy [13]. Rule-based features are very much dependent on language. Jassim et. al(2021) have used an N iterative HMM model for parts of speech tagging in Iraqi National Song. The iterative approach has improved accuracy as claimed by the authors [14]. Since HMM uses a huge amount of training data, it can very well map the context of the words and provide high accuracy as seen in the works conducted by the researcher discussed above. However, the model suffers in the case of words that are not trained during training. Among the early works, Ratnaparkhi (1996) proposed a maximum entropy model [15] to successfully tag unseen words with accuracy up to 96%. It uses some specialized features to take decisions and uses approximately 900 thousand of words from Wall Street Journal corpus as a training data set taken from the Treebank project (Marcus et al., 1994) [16]. Robert M. Losee [17] used tagging for improving decision-making with the help of linguistic information. Toutanova et al. [18] proposed a part-of-speech tagging using lexical features, preceding and following text context with fine-grained modeling for features of unknown words. Martin Haulrich reported [19] the implementation of a part-of-speech tagger based on the first-order Hidden Markov Model and compared different strategies to improve the result for unknown words. Other rule-based techniques take decisions based on the presence or absence of a number, upper-case letter, etc as proposed by researchers [15]. Rules may be added to improve accuracy for the unknown word. But these rules are language-dependent. Scott M.Thede [20] proposed a few statistical methods for predicting unknown words. The methods are applicable to any language. The author used Brown corpus [7] in this case too. Mikheev (1996) used the beginning and end of a word to predict the parts of speech for unknown words [21]. They used certain morphological rules: Prefix rules, suffix rules, and ending-guessing rules. Anastasyev et. al [22] described rules for detecting unknown word tagging for rich languages. Use of context, word endings are used as clues for detecting parts of speech. The following sections detail the basics principle of the Viterbi Algorithm and Hidden Markov Model.

C. Basic Principle of Hidden Markov Model

The Hidden Markov Model (HMM) basically comprises of two kinds of events: observed events and hidden events. For the part of speech tagging problem, observed events are the words that appear in the input text and hidden events are the parts of speech that are to be predicted. Components of hidden Markov model are: a matrix of state transition probability, sequence of observation, sequence of emission probabilities and initial probability distribution [23] [24].

1) Transition probability: It is the first Markov assumption which implies that the current state depends only on the previous state. We will call this as transition probability. A transition probability matrix is to be calculated based on training data for all pairs of tags (states). Mathematically it is represented by:

$$P(q_i|q_1...q_{i-1}) = P(q_i|q_{i-1}) \quad (1)$$

where, qi is the state at i^{th} instance.

2) Emission probability

$$P(o_i|q_1 ...q_i,...,q_T , o_1,...,o_i,...,o_T) = P(o_i|q_i) \quad (2)$$

Here, oi is the observation at ith instance.

This second Markov assumption implies that the observation at any instant depends only on the present state. We will term it as emission probability. An emission probability matrix is to be calculated based on training data for all pairs of tags (states) and words (observations).

3) Viterbi algorithm: It is used to predict the part of speech of a word based on maximum likelihood calculated as:

$$V(t(j)) = \max : V(t-1) * a(i,j) * b_j(O_t), \quad (3)$$

,V(t(j)) is the Viterbi path probability of the model being at state j at any instance, after observing the first t number of observations. Here a is transition probability matrix and b is emission probability matrix [25] [26].

The problem with this basic algorithm is the fact that the emission probability P(oi|qi) is zero for an unknown word. It is because.

$$P(o_i|q_i) = N(o_i|q_i) / Nq_i = 0 \quad (4)$$

where, N ($o_i|q_i$) is the number of time observation oi appears in training set as state q_j and Nq_i is the number of time state qi appears in training set. Hence, over the years scientists have come up with different methods to improve accuracy for such words. One simple method is Laplace smoothing, where the following modification is made:

$$P(o_i|q_i) = (1+N\ o_i|q_i) / (Nq_i + V)$$

where V is the length of vocabulary trained.

Considering only the transition probability for calculation in case of unknown words is another option to overcome it by replacing the value zero by one .A brief discussion on earlier works in this area is discussed above in this section.

D. Related Work for Low Resource Languages

The resources available for the available languages in the world are extraordinarily unbalanced [27]. There are many organizations that are working dedicatedly for technology development on low resource language. Under the LORELEI (Low Resource Languages for Emergent Incidents) Program of Linguistic Data Consortium for DARPA, researchers are working on developing NLP technologies for natural disaster management for almost three dozen low resource languages [28]. NLP research teams are working seriously on approximately 20 of the almost 7000 languages of the world

leaving a majority of the population not reachable to advanced NLP applications [29]. Low resource languages are those language that are less computerized, less privileged resource scared languages. These are languages where statistical methods cannot be applied due to the availability of fewer data [29], [30], [31], [32]. As Simpson et. al. [33] reported that a low resource language: "All meet the basic criteria of being significant in terms of the number of native speakers but poorly represented in terms of available language resources." Christopher et. al. states that defining a language as "low resource language" depends on the Demography, Linguistics, and Resource availability of the language and the speakers [31]. Some amount of work has been done for low resource languages, but the researchers are not yet able to develop NLP applications for majority of the languages. This is because a strong language-independent method is highly essential to work with languages with fewer amounts of training data to develop NLP applications. With the low amount of training, testing always encounters more and more unknown data and it eventually makes the hidden mark model not much useful for such cases. Recently, some works [34] [35] for resource-poor language and rule-based methods have shown some improvement. But still, rule-based NLP applications are language-specific and the advantages are limited. Researchers have used unsupervised techniques for low resource languages. N-gram Models [36] can be also very useful for of processing many natural language processing tasks. Authors reported some considerable result taking help from another resourceful language as parent language and with standardized text for the two languages [37], [38]. But they also reported difficulty in choosing the parent language because typologically close languages do not always work best. Researchers have used modern days supervised techniques based on long short-term memory networks (LSTM) on multilingual embeddings to get good results. But that also requires quite a large training dataset [39], which is not available for most of the languages. Some Researchers have used bilingual lexicon available to some extent for few languages to investigate the possibility of designing language models with limited training data. The method uses the learning of cross-lingual word embeddings to train monolingual language models. The training shows improvements due to the pre-training process [40]. Some amount of work has also been done for Assamese using some stochastic methods [41], [42], [43] [44] [45]. Recently authors have discussed the key areas of NLP research in Assamese [46]. Researchers have also been working recently on parts of speech and other nlp issues on Arabic languages [47] [48].

E. Related Work using Machine Learning

Recently deep learning methods have gained high popularity. The same is being used for Indian and other low-resource languages. Sequential deep learning methods are very popular in this regard. Some of them are long short-term memory (LSTM), bidirectional LSTM, gated recurrent units (GRU), recurrent neural network (RNN), etc. Authors in [49], have applied deep learning for tagging the Chinese Buddhist language. Their learning model is based on RNN. The model as informed by the author is more effective than traditional methods. Bidirectional LSTM is another popular method in this regards authors in [50], experimented with BLSTM and auxiliary loss over a set of 22 languages. They used the auxiliary loss to improve the performance for rare words. Techniques of using character level along with word-level representation are used recently for POS tagging using deep neural networks. Authors in [51] used the method for English and Portuguese. A convolutional layer was used to prepare the data with character representation. Authors in [52] discussed in detail, how to represent character-level information from raw text. They successfully did it to predict the next character from a given sequence of characters. They used a simple recurrent network for this purpose.

Authors in [53], reported using the character level outputs of convolutional neural network (CNN) as inputs to an LSTM RNN model. The authors have stated that it highly improves the performance in the case of morphologically rich languages like Russian, Spanish, French, Czech, Arabic, German. Machine learning approaches have recently been used with many low-level languages. Authors in [54] described their architecture for Korean POS tagging. They addressed the issue of rare word detection by input-feeding and copying mechanism and got considerable results. Authors in [55] used machine learning models for POS tagging of Sanskrit language. They represented each word as a point and then used clustering with LSTM autoencoder to get the tagging. Authors in [56] used deep learning methods for the Nepali language. They used Long Short-Term Memory Networks (LSTM), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and bidirectional variants to successfully tag Nepali words with high accuracy.

III. Research Objectives

Parts of speech is the preliminary pre-processing step that requires to be executed for any NLP application. But the popular methods available for tagging require a huge amount of training data. They perform very poorly for languages for which a huge amount of training data is not available. The problem is to develop parts of speech tagging methods that are applicable to any language in the world with a very low amount of computable resources.

Most common methods like the Hidden Markov Model and Machine Learning are not well applicable to languages where large amounts of computable data resources are not available. Accuracy is also not very high for words that are not trained earlier because the models cannot read much information for such words. Some of the rule-based methods are mainly used to address these issues. But the scope of such a rule is limited and dependent on the language for which rules are set. The need is to develop systems that can improve accuracy, especially for unknown words, with low training and applicable to any language. This paper concentrates on language-independent approaches towards using the Hidden Markov Model and deep learning techniques with a very small amount of training data so that it can be used for any low resource language with considerable performance. Main focus is to devise language-independent methods to improve accuracy especially, the accuracy of unknown words which is the major concern for any supervised method trained with a low training dataset.

IV. Proposed Method

A. Using Modified Viterbi Algorithm

We have proposed modifications to Hidden Markov Model to improve accuracy for untrained words with a small set of training data. We have used Subset from Brown Corpus [7], which is mostly used by researchers in this area. This will give a better platform for comparing proposed modifications with exiting methods. Proposed method is based on probability of character bigrams and character trigrams. The emission probability of a character bigram bi given tag ti is calculated as follows:

$$P(b_i | t_i) = N(b_{i} | t_i)/ N (t_i) \quad (5)$$

where $N(b_i | t_i)$ is number of time bigram b_i appears in training set as tag t_i and $N(t_i)$ is Number of time tag t_i appears in training set. Similarly

$$P(tr_i | t_i) = N (tr_i | t_i) / N (t_i) \quad (6)$$

where $N(tr_i | t_i)$ is the number of time trigram tr_i appears in training set as tag t_j and $N(t_i)$ is Number of time tag t_i appears in training set.

The probability of unknown words is calculated based on the fact that bigrams (and trigrams) constituting the unknown word may have already appeared in some trained words, and thus this information may be used to predict the possible tag of the unknown word being tested. We assume that the probability of a word given a tag is proportional to the product of probabilities of sequences of character bigrams (also trigrams) of the word given the tag. Thus instead of considering the emission probability of unknown word as zero or one, we make the following changes:

$$P(o_i|q_i) \propto c * P(b_1|t_i) * P(b_2|t_i) *\ldots\ldots* P(b_n|t_i) \quad (7)$$

where c is a constant and b1, b2…bn are the bigrams of the characters constituting the word o_i.

Hence, equation (3) can be rewritten as

$$V(t(j))=max:V(t-1)*(P(b_1|t_i)*P(b_2|t_i)*\ldots*P(b_n|t_i))*b_j(O_t) \quad (8)$$

If any value of $P(b_k| t_i)$ turns out to be zero, it is considered to be a very small value to avoid zero product.

Similarly the same kind of modifications can be made for trigrams of characters:

$$V(t(j))=max:V(t-1)*(P(tr_{i1}|t_i)*P(tr_{i2}|t_i)*\ldots*P(tr_{in}|t_i))*b_j(O_t) \quad (9)$$

If any value of $P(tr_{ik}|t_i)$ turns out to be zero, it is considered to be a very small value to avoid zero product. This is an alternative to considering a zero or one value for the emission probability for the entire word as discussed in the previous section. This helps us to guess the probability of an unknown word, by using the probabilities of bigrams that may have occurred with other words that are already trained.

B. Using Deep Learning Architecture

The recent usage of neural network and machine learning methods has proved to be very much useful in modern technology. One of the most popular neural networks used for this purpose is a recurrent neural network (RNN). It feeds the output of one stage as input to the next. The states in RNN can store input of variable length of sequences. This particular property makes it very much useful for inputs with variable lengths like text sentences, speech processing, etc. LSTM is a kind of RNN that uses a special unit that can store memory for the long term. This kind of model can keep information retained for a long time because of its ability to select the kind of information to retain.

We have designed an architecture for the deep learning model and have successfully implemented it to work considerably well with less amount of training data. The work is inspired by [57], [58], and [59], where authors have used character-level representation along with word-level representation to train the model. We take a sequence of tagged words and feed the words, characters, bigrams and trigrams of characters of words into the first layer of the learning model. We have experimented with different parameters of the layers to get the best-suited architecture. The first layer of the learning network transforms words into feature vectors. It captures information about words' semantic and their morphological characteristics. Every word is converted into a vector of sub vectors of word-level embedding, character-level embedding, bigram character-level wording, and trigram character-level wording.

1) Word-level embeddings: Word-level embeddings are encoded in an embedding matrix by column vectors where each column represents the word-level embedding of the corresponding word in the vocabulary. Every word thus is converted into its word-level embedding. A word is first encoded as a one-hot column vector. It is then fed to the input layer. A word embedding matrix is used to multiply it to finally get the word embedding. A word vector at time instant t, WV_t is multiplied with embedding matrix WM_w to get the Word Embedding E_w as follows:

$$E_w = WV_t \text{ x } WM_w \quad (10)$$

2) Character-level embeddings: All characters in a word are represented by a character level embedding. Like, word-level embedding, Character-level embeddings are encoded in an embedding matrix by column vectors where each column represents the character-level embedding of the corresponding character in the character vocabulary. The word embedding is calculated by concatenating word and character embeddings.

$$E_w = (WM_w \text{ x } WV_t) € (CM_c^1 \text{ x } CV_t^1) € (CM_c^2 \text{ x } CV_t^2) \ldots € (CM_c^n \text{ x } CV_t^n) \quad (11)$$

where, € is concatenation symbol. A character vector at time instant t and position i, CV^i_t is multiplied with embedding matrix CM_c to get the Character Embedding.

3) Bigram-character-level embeddings: All bigram combinations of characters in a word are represented by a bigram-character level embedding. Like, word-level embedding, bigram-character level embeddings are encoded in an embedding matrix by column vectors where each column represents the bigram character-level embedding of the

corresponding bigram-character in the bigram-character vocabulary. The word embedding is then calculated by concatenating word, character embeddings, and bigram character embeddings.

$$E_w = (WM_w \times WV_t) € (CM_c^1 \times CV_t^1) €…€(CM_c^n \times CV_t^n),$$
$$€ (BM_c^1 \times BV_t^1) € … € (BM_c^m \times BV_t^m) \quad (12)$$

where € is concatenation symbol. A bigram character vector at time instant t and position i, BV_t^i is multiplied with embedding matrix BM_c to get Bigram Character Embedding.

4) Trigram-character-level embeddings: Similarly, trigram combination of characters in a word is represented by a trigram level embedding. It is encoded in an embedding matrix by column vectors where each column represents the trigram character-level embedding of the corresponding trigram-character in the trigram-character vocabulary.

$$E_w = (WM_w \times WV_t) € (CM_c^1 \times CV_t^1) €…€ (CM_c^n \times CV_t^n),$$
$$€(TM_c^1 \times TV_t^1)€ ……€(TM_c^m \times TV_t^m) \quad (13)$$

where € is concatenation symbol. A trigram character vector at time instant t and position i, TV_t^i is multiplied with embedding matrix TM_c to get Trigram Character Embedding.

The addition of bigram combinations and trigram combinations helps the model to learn the inflections and morphological patterns related to tagging very well. The results are discussed in the next section. The basic principle of concatenating the different kind of embedding is illustrated in Fig. 1.

The difference with the usual LSTM applied is in the fact that word embedding in proposed model is a concatenation of word embedding, character embedding, and bigram character embedding or trigram character embedding. The order of characters is the same as the order in which they appear in the word. Similarly, the order of bigram and trigrams of characters are also in the same order in which they appear in the word. The embedding dimensions are decided after repeated experiments to get the most suitable value.

C. Data Sets

The English dataset is based on already tagged Brown Corpus [7]. The corpus is based on the current American English language containing about a million words. It comprises elements of statistics, psychology, linguistics, and sociology. Kučera and Francis (1967) reported their initial work on basic statistics on the corpus which eventually turned into Brown Corpus [7] [60].

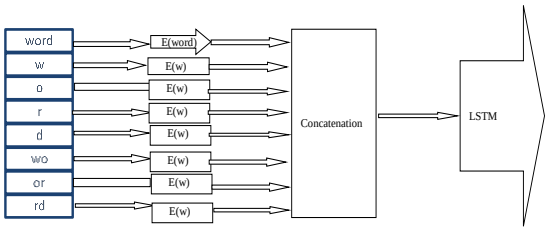

Fig. 1. Concatenating the Embeddings to Feed into LSTM.

The dataset considered for Assamese is developed in-house. Assamese is the language spoken over the state of Assam and entire North-East India. However, not much information is available in computable form. A publicly available set of tagged words is not available in Assamese. Hence it is prepared in-house. It comprises an annotated text of approximately ten thousand words. The dataset is prepared from a corpus collected from TDIL (Technology Development for Indian Languages). It contains articles of different categories like storybooks, scientific articles, health articles, drama, etc. The corpus was tagged into different kinds of parts of speech as per Assamese grammar.

V. Experiments and Results

We have prepared three small datasets of three different sizes of words taken from already tagged Brown Corpus. The original implementation of the Viterbi algorithm performs poor due to low training data. Due to low training, it cannot approximate the transition matrix accurately. Also, it encounters many unknown words during testing because of the small training size. In the original Viterbi algorithm, due to zero utterances of unknown words in training data, the emission probability evaluates to zero, thus resulting in zero value for the observation. This is already stated in equation (4)

$$P(o_i|q_i) = N(o_i| q_i) / Nq_i = 0$$

The transition probability is only considered by many researchers for the multiplication of transition probability and emission probability. It is equivalent to consider emission probability as one for unknown words. This is obviously a biased method, thus resulting in poor performances for both unknown and known words. Again, if zero value for emission probability is considered for unknown words, then the calculated value becomes zero for all unknown words, which is an obvious fault.

For small training data, this is a big challenge because, with a small training data, the transition history cannot be learnt properly and so it can never accurately measure the transition matrix. Therefore, the performance is very poor especially for unknown words. Proposed method replaces the transition probability with the probability of multiplication of individual emission probability of bigrams. The result shows improvement as it gives the word a tag based on the probability distribution of its constituent bigrams towards the tags as per corpus. The product of individual probabilities of bigrams is multiplied with transition probability, thus considering both emission and transition rather than considering none or only transition probability.

First, experiment was conducted for 5000 words as Training Set and 1250 words as Test Set for the very basic Viterbi Algorithm. The performance is poor because the system was trained with too low data. Similarly, the same was done with 10,000 and 20,000 words of training and test them with 2,500 and 5,000 words respectively for the basic Viterbi algorithm as a baseline for comparison.

The result of the implementation of the basic Viterbi algorithm on the three small sequences only tagged words (4:1 ratio for trained and tested words) used for training is tabulated

in Table I. The result shows improvement with more training data due to better guess of transition probability because of more training.

TABLE I. RESULT OF THE BASIC VITERBI ALGORITHM TRAINED WITH SMALL SET OF TAGGED WORDS

Training Size(words)	5000	10000	20000
Number of word tested	1250	2500	5000
unknown word tested	318	552	1131
Unknown Word accuracy	35.53%	39.86%	40.58%
Overall Word accuracy	79.68%	82.48%	81.06%

As discussed earlier, the evaluation of the correct tag(state) is based upon the equation:

$V(t(j)) = \max : V(t-1) * a(i,j) * bj(O_t)$

It can be simplified as finding maximum likelihood of state over all possibility based upon the equation.

P(State)=P(Tag/PrevTag)*P(Word/Tag) (14)

In simple term, it can be written as:

P(State) =P(emission) * P(transition)

Next, two modifications are made on the following basic formula:

P(State) =P(emission) * P(transition)

A. Modification Method1

It is based on emission probability of character bigrams and trigrams. The probability of a state is calculated based upon the following equation:

P(State) = Product of emission probabilities of bigrams (trigrams) (15)

The transition probability is discarded as work is concentrated on small set of training data.

B. Modification Method2

It is based on emission probability of character bigrams and trigrams and transition probability of tags. The probability of a state is calculated based upon the following equation:

P(State)=Product of emission probabilities of bigrams (trigrams) * P (transition) (16)

A part of the bigram probability matrix is also shown in Table II. It is calculated from first 5000 words of Brown corpus. The table shows probabilities of BIGR(Bigram) for the following parts of speech: NOU(Noun), VRB (Verb), ADJ(Adjective), DET (Determinant), ADP(Adposition), PRO (Pronoun), ADV(Adverb) and CON(Conjunction). Matrix for only five bigram combinations are shown.

The basic algorithm performs better with the larger size of the training data set. The accuracy value above 96% has been reported [1] using entire Brwon Corpus [7] of approximately 540 thousand words. However, unknown word accuracy is not good. Then the proposed modifications are applied upon the same set of data. The goal is to improve the results with a small set of training data. The trigram character probability has also been used instead of using bigram probability for the same sets of data. The results with the four datasets of different sizes are described in Table III, Table IV, Table V and Table VI, respectively.

A comparison of the results for the four different sizes of training data with the same 4:1 ratio of different testing and training dataset are shown in Fig. 2. Improvement is more visible with bigram characters than that of trigrams. For subsequent experiments, bigram characters are used.

Next, Experiments were also conducted for bigrams with increased rate of testing data (50% training and 50% testing) and the results are tabulated bellow in Table VII. In this case three different sizes of text were taken from Brown corpus and then the systems were trained using them. Equal volume of data was used to test each of the systems. The test dataset was selected from a different part of the corpus.

TABLE II. PART OF THE BIGRAM PROBABILITY MATRIX

BIGR	**NOU**	**VRB**	**ADJ**	**DET**	**ADP**	**PRO**	**ADV**	**CON**
Aa	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Ab	0.0012	0.0022	0.0098	0.0000	0.0000	0.0000	0.0130	0.0000
Ac	0.0045	0.0053	0.0033	0.0084	0.0015	0.0000	0.0047	0.0000
Ad	0.0034	0.0075	0.0025	0.0000	0.0000	0.0000	0.0083	0.0000
Ae	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

TABLE III. SUMMARY OF THE RESULTS FOR TRAIN SIZE OF 5000

	Original Method	**Modification Method1 (Equation 11)**		**Modification Method2 (Equation 12)**	
		Bigram	**Trigram**	**Bigram**	**Trigram**
Size of Test Set	1250	1250	1250	1250	1250
Unknow word	318	318	318	318	318
Unknown Word accuracy	35.5%	55.9%	54.1%	56.6%	55.6%
Overall Word accuracy	79.7%	85.0%	84.8%	85.3%	85.2%

TABLE IV. SUMMARY OF THE RESULTS FOR TRAIN SIZE OF 10000

	Original Method	Modification Method1 (Equation 11)		Modification Method2 (Equation 12)	
		Bigram	Trigram	Bigram	Trigram
Size of Test Set	2500	2500	2500	2500	2500
Unknow word	552	552	552	552	552
Unknown Word accuracy	39.9%	58.7%	57.6%	65.4%	59.0%
Overall Word accuracy	82.5%	86.8%	86.6%	88.4%	87.0%

TABLE V. THE COMPARATIVE SUMMARY OF THE RESULTS FOR TRAIN SIZE OF 20000

	Original Method	Modification Method1 (Equation 11)		Modification Method2 (Equation 12)	
		Bi gram	Tri gram	Bi gram	Tri gram
Size of Test Set	5000	5000	5000	5000	5000
Unknow word	1131	1131	1131	1131	1131
Unknown Word accuracy	40.6%	52.4%	51.9%	60.7%	55.6%
Overall Word accuracy	81.1%	83.7%	83.9%	85.7%	84.7%

TABLE VI. THE COMPARATIVE SUMMARY OF THE RESULTS FOR TRAIN SIZE OF 50000

	Original Method	Modification Method1 (Equation 11)		Modification Method2 (Equation 12)	
		Bigram	Trigram	Bigram	Trigram
Size of Test Set	12500	12500	12500	12500	12500
Unknow word	2076	2076	2076	2076	2076
Unknown Word accuracy	42.3%	46.4%	51.6%	60.4%	55.7%
Overall Word accuracy	85.8%	86.5%	87.5%	88.8%	88.2%

TABLE VII. RESULTS WITH 50:50 RATIO OF TRAINING AND TEST DATASET USING BIGRAM CHARACTERS

	Original Method		Modification Method1 (Equation 11)		Modification Method2 (Equation 12)	
Size of Train Set	10000	20000	10000	20000	10000	20000
Size of Test Set	10000	20000	10000	20000	10000	20000
Unknow word	2723	4328	2723	4328	2723	4328
Unknown accuracy	38.6%	43.2%	55.4%	49.9%	63.2%	59.9%
Overall accuracy	78.5%	82.3%	83.3%	83.8%	85.4%	86.0%

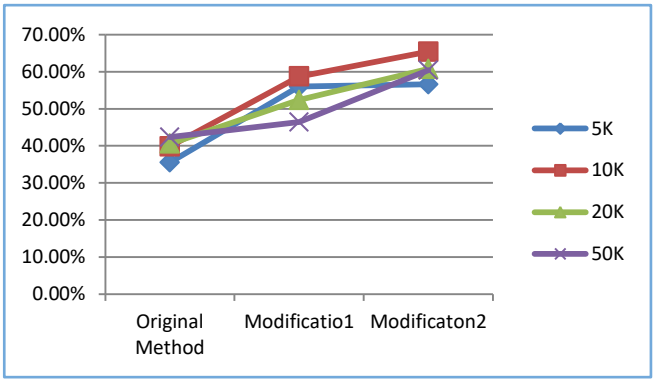

Fig. 2. Comparison of the Three Methods for unknown Words with 4:1 Ratio of different Testing and Training Dataset Bigram Characters.

With the size of test dataset being almost double than the previous experiments, the results are seen to be consistent for unknown words. A comparison is detailed in Fig. 3.

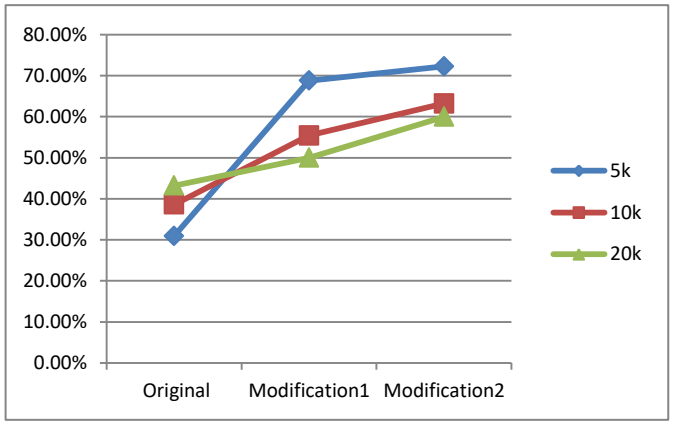

Fig. 3. Comparison of the Three Methods for unknown Words with 1:1 Ratio of different Testing and Training Dataset.

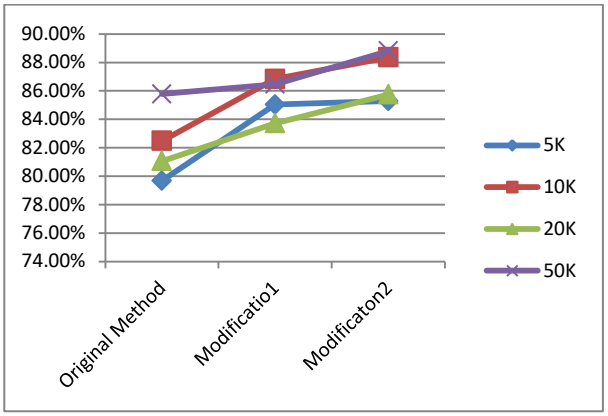

Fig. 4. Overall Accuracy for different Sizes of Train and Test Data (4:1) with Bigram of Characters.

The accuracy for unknown words is only detailed above. The overall accuracy is not any issue with Hidden Markov Model, which is high in this case too even with very low size of training data. The overall accuracy for the different sizes using bigram characters are depicted in Fig. 4.

The system is also tested with "Assamese" language with low training data. Assamese is a low resource language spoken in the state of Assam located in North East India. The findings are tabulated below in Table VIII and a brief comparison is shown in Fig. 5 and Fig. 6.

The system is also tested with "Assamese" language with 1:1 ratio of training and testing data. The findings are tabulated below in Table IX and Fig. 5.

The Assamese corpus was collected from TDIL (Technology Development for Indian Languages), which can be procured from their website after due permission. The corpus was tagged as per Assamese grammar rules to prepare train and test dataset. The result shown here is based on a train and test size of 10k words each that are non-overlapping. The results are satisfactory even with large proportion of unknown words.

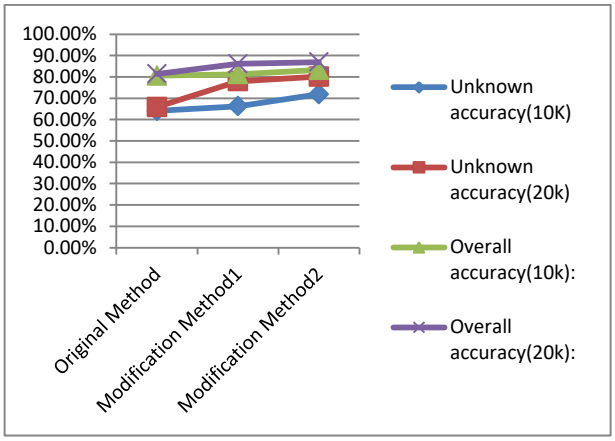

Fig. 5. Accuracy for 10K and 20 K of Training with 2.5 K and 5 K Testing Data respectively (4:1) using Bigram Characters for Assamese.

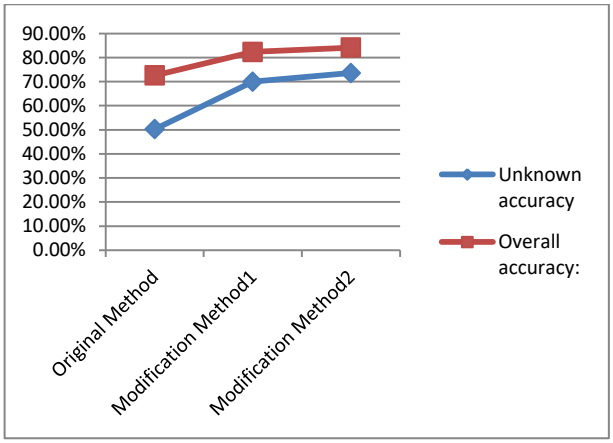

Fig. 6. Accuracy for 10K of Training and 10 K Testing data (1:1) using Bigram Characters for Assamese.

TABLE VIII. RESULTS WITH 4:1 RATIO OF TRAIN AND TEST SET USING BIGRAM CHARACTERS FOR ASSAMESE

	Original Method		**Modification Method1 (Equation 11)**		**Modification Method2 (Equation 12)**	
Size of Train Set	10000	20000	10000	20000	10000	20000
Size of Test Set	2500	5000	2500	5000	2500	5000
Unknow word	907	1960	907	1960	907	1960
Unknown accuracy	64.1 %	65.9%	66.3%	77.9%	71.9%	80.2%
Overall accuracy	80.6%	81.4%	81.2%	86.1%	83.3%	86.9%

TABLE IX. RESULTS OF 1:1 RATIO OF TRAIN-TEST SET USING BIGRAM CHARACTERS FOR ASSAMESE

	Original Method	**Modification Method1 (Equation 11)**	**Modification Method2 (Equation 12)**
Size of Train Set	10000	10000	10000
Size of Test Set	10000	10000	10000
Unknow word	5034	5034	5034
Unknown accuracy	50.26%	69.98%	73.60%
Overall accuracy	72.56%	82.34%	84.17%

C. Result with Deep Learning Model

The proposed deep learning model is then applied to automatically tag the same set of English words used above. The result obtained is satisfactory and described in Table X and Fig. 7. The training dataset used is much smaller to check its usefulness for low resource languages. Datasets of sizes 10k and 20 k are used to train the system and it is tested with the equal size of different data.

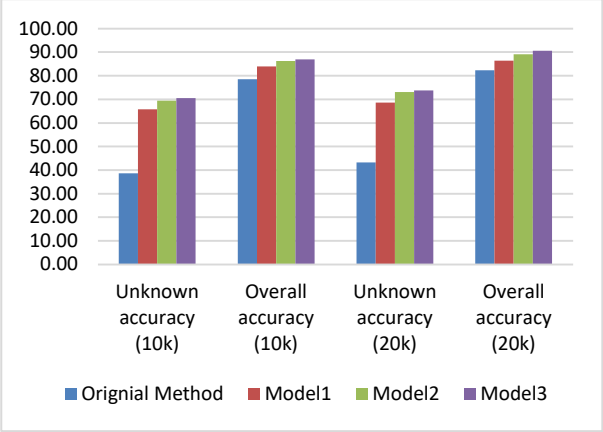

Fig. 7. Accuracy for 10K of Training and 10 K Testing Data (1:1) using Deep Learning for English.

Next, proposed deep learning model is used for the same purpose for the Assamese Language. Initially, only the traditional machine learning method of using word-level is applied embedding to get an accuracy of 72.51% which is comparable to the proposed traditional stochastic model. Next, the system is trained with word and character level embedding (Model1) and the accuracy jumps to 88.21%. Next, the model combining word and character sequences with tigrams (Model2) and trigrams (Model3) is implemented. As the models learn the morphological behaviours much better than before, the accuracy goes up to 93.52% for bigrams and 94.51% for trigrams. The accuracy of unknown words also raises due to better learning. Table XI and Fig. 8 compares the result of applying the proposed deep learning models for Assamese.

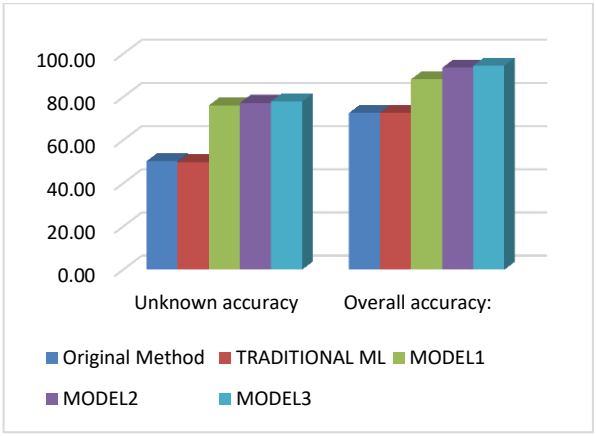

Fig. 8. Accuracy for 10K of Training and 10 K Testing Data (1:1) using Deep Learning for Assamese.

TABLE X. RESULT WITH 50:50 RATIO OF TRAIN AND TEST SET USING DEEP LEARNING FOR ENGLISH

	Original Method		**Model1**		**Model2**		**Model3**	
Train Set:	10000	20000	10000	20000	10000	20000	10000	20000
Test Set	10000	20000	10000	20000	10000	20000	10000	20000
Uknown word	2723	4328	2723	4328	2723	4328	2723	4328
Uknown accuracy	38.6%	43.2%	65.8%	68.6%	69.5%	73.1%	70.6%	73.8%
Overall accuracy	78.5%	82.3%	84.0%	86.4%	86.3%	89.1%	86.9%	90.6%

TABLE XI. RESULTS WITH EQUAL SIZE TRAIN AND TEST SET USING DEEP LEARNING FOR ASSAMESE

	Original Method	**Traditional ML Method**	**Model1**	**Model2**	**Model3**
Train Set Size	10000	10000	10000	10000	10000
Test Set Size	10000	10000	10000	10000	10000
Unknown word	5034	5034	5034	5034	5034
Unknown accuracy	50.26%	45.82%	76.08%	77.27%	77.89%
Overall accuracy	72.56%	72.41%	88.21%	93.52%	94.51%

VI. Discussion

The results of the experiments with modified HMM methods clearly state improvement of accuracy for unknown words. The training sets of 5k, 10k, 20k, and 50k are used on an experimental basis and the system can also be tested with a large dataset. However, as the goal is to improve accuracy with a small training set, so the limited sizes of data are considered for training. The methods do not use any specific rule, and hence can be implemented for any language. The initial experiments are based on a 4:1 ratio of training and testing dataset, which shows considerable improvement with Modifications even though the test size increased with an increase in training size to maintain the ratio. The results are also encouraging for the next set of experiments that are based on a 1:1 ratio of training and testing dataset. Even with equal sizes of testing and training datasets, the system performs considerably well specially for a second modification. In the experiments conducted, it is observed that the accuracy has improved with an increase in the training set from 5,000 to 50,000 particularly in the case of modification method2. This happens because, with an increase in the size of the training set, the transition probability starts contributing along with emission probability, thus increasing accuracy. The improvement observed is less in Method1 as it only considers emission probability. Table VII and Table IX clearly show that the accuracy does not degrade even after increasing test size and unknown words. The system maintains overall accuracy of above 85% in all cases, which is considered good with such a low amount of training. The overall accuracy is higher than 80% even when it is exposed to a test dataset of size equal to that of the training set.

The accuracy further improves with the proposed deep learning model. The words when trained with characters and bigram or trigrams of characters improve the accuracy further as shown in Table X and Table XI. The model learns better when bigram and trigrams of characters are fed into the input along with character sequences. The bigrams and trigrams of characters allow the model to learn better the inflections and hence accuracy improves. The improvement is more obvious for Assamese because of high inflections.

Table XII compares some of the works carried for Assamese in recent times.

TABLE XII. Comparing the Result with other Recent Works in Assamese

Reference Paper	**Accuracy**
[41]	89.21%
[42]	87.17%
[43]	87%
Work refrerred in this Paper	94.51%

VII. Conclusion

It is observed that the accuracy of transition probability increases with an increase in the size of the training dataset. The transition among the different parts of speech can easily be computed in the form of transition probability with the help of a very large training set. But, for a small set of training data, the transition probability is not predictable. Hence, for a small set of training data, emission probability plays the most important role to decide the total probability. Due to non-appearance in the training set, the emission probability for unknown words is zero and this is the root cause of the problem for detecting correct tags of unknown words. With the usage of bigram and trigram of characters in proposed modifications, unknown words may also have non-zero emission probability if such bigrams and trigrams have ever occurred during training other words. This increases the accuracy while classifying unknown words. The experiments conducted for English with low training data prove that the results are comparable with other methods used with large training data. The same technique of character sequences, bigrams sequences of characters, and trigrams sequences of characters applied to design a deep learning model also makes the system learn the behaviour so well that accuracy level increases up to a great extent. The system also performs well for low resource language like “Assamese” when used with a very small volume of training data. The methods are language independent and we hope that the methods will be useful for future implementation for any low resource language. More in-depth research on this will further improve accuracy for low resource language.

References

[1] Doug Cutting and Julian Kupiec and Jan Pedersen and Penelope Sibun, “A Practical Part-of-Speech Tagger”, https://www.aclweb.org/anthology/A92-1018.pdf.

[2] A. R. Martinez (2011), “Part-of-speech tagging. Wiley Interdisciplinary Reviews” Computational Statistics, 4(1), 107–113. doi:10.1002/wics.195.

[3] Jürgen M.Janas, “Automatic recognition of the part-of-speech for english texts”, Information Processing & Management, Volume 13, Issue 4, 1977, Pages 205-213.

[4] Fernando, Sandareka, Surangika Ranathunga, Sanath Jayasena, and Gihan Dias, "Comprehensive part-of-speech tag set and svm based pos tagger for sinhala." In Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP2016), pp. 173-182. 2016.

[5] Hyun-Je Song and Seong-Bae Park.,“Korean Part-of-speech Tagging Based on Morpheme Generation”, ACM Trans. Asian Low-Resour. Lang. Inf. Process. 19, 3, Article 41 (April 2020), 10 pages. DOI:https://doi.org/10.1145/3373608.

[6] Can Udomcharoenchaikit, Prachya Boonkwan, and Peerapon Vateekul., “Adversarial Evaluation of Robust Neural Sequential Tagging Methods for Thai Language”,ACM Trans. Asian Low-Resour. Lang. Inf. Process. 19, 4, Article 53 (July 2020), 25 pages. DOI:https://doi.org/10.1145/3383201.

[7] Francis, W. Nelson & Henry Kucera,” BROWN CORPUS MANUAL: Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English for Use with Digital Computers.”, 1979 http://icame.uib.no/brown/bcm.html.

[8] D.L Cing,. and K.M Soe, “Joint word segmentation and part-of-speech (POS) tagging for Myanmar language”, Seventeenth International Conference on Computer Applications (ICCA 2019).

[9] Jurgen Van Gael, Andreas Vlachos and Zoubin Ghahramani, “The infinite HMM for unsupervised PoS tagging.” ,EMNLP (2009).

[10] Phyu Hninn Myint, Tin Myat Htwe, and Ni Lar Thein, "Lexicalized HMM-based Part-of-Speech Tagger for Myanmar."

[11] Thorsten Brants, "TnT-a statistical part-of-speech tagger." arXiv preprint cs/0003055 (2000).

[12] Antonio Molina, & Ferran Pla, (2001), “Clause Detection using HMM” 10.3115/1117822.1455688.

[13] Tham, Medari Janai. "A Hybrid POS Tagger for Khasi, an Under Resourced Language." International Journal of Advanced Computer Science and Applications(IJACSA) 11, no. 10 (2020): 333-342.

[14] Jassim, Abbood Kirebut, and Boshra F. Zopon Al_Bayaty. "A Stochastic Approach to Identify POS in Iraqi National Song using N-Iterative HMM using Agile Approach.", IOP Conference Series: Materials Science and Engineering. Vol. 1094. No.1.IOP Publishing, 2021.

[15] Ratnaparkhi, Adwait ,"A Maximum Entropy Model for Part-Of-Speech Tagging", Conference on Empirical Methods in Natural Language Processing,1996, available "https://www.aclweb.org/anthology/W96-0213.

[16] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, Britta Schasberger "The Penn Treebank: annotating predicate argument structure." In HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994. 1994.

[17] Robert M.Losee,"Natural language processing in support of decision-making: phrases and part-of-speech tagging", Information Processing & Management Volume 37, Issue 6, November 2001, Pages 769-787.

[18] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003, "Feature-rich part-of-speech tagging with a cyclic dependency network", In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1 (NAACL '03). Association for Computational Linguistics, USA, 173–180. DOI:https://doi.org/10.3115/1073445.1073478".

[19] Martin Haulrich, "Different Approaches to Unknown Words in a Hidden Markov Model Part-of-Speech Tagger",2009. available at: https://cl.lingfil.uu.se/~nivre/statmet/haulrich.pdf.

[20] Scott M. Thede. "Predicting part-ofspeech information about unknown words using statistical methods.", Proceedings of COLING-ACL '98, pages 1505-1507.

[21] Andrei Mikheev. 1997, "Automatic rule induction for unknown-word guessing", Comput. Linguist. 23, 3 (September 1997), 405–423.

[22] Daniil Anastasyev, Andrew Andrianov, and Eugene Indenbom. "Part-of-speech tagging with rich language description." In Computational Linguistics and Intellectual Technologies. Proceedings of the International Conference "Dialogue, vol. 16, no. 23, pp. 2-13. 2017.

[23] L R Rabiner (1989) "A tutorial on hidden Markov models and selected applications in speech recognition." Proc IEEE 77(2):257–286.

[24] Daniel Jurafsky, James H Martin. 2000. "Speech and Language Processing. Pearson Education".

[25] A J Viterbi (April 1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". IEEE Transactions on Information Theory. 13 (2): 260–269. doi:10.1109/TIT.1967.1054010.

[26] G. David Forney Jr , "The Viterbi Algorithm: A Personal History",2005.

[27] Daniel Nettle," Explaining Global Patterns of Language Diversity", Journal of Anthropological Archaeology, Volume 17, Issue 4,1998,Pages 354-374, ISSN 0278-4165.

[28] S Strassel. and J Tracey "Lorelei language packs: Data, tools, and resources for technology development in low resource languages", In Proceedings of the Tenth International Conference on Language Resources and Evaluation (2016), pages 3273–3280.

[29] Magueresse, Alexandre, Vincent Carles and Evan Heetderks, "Low-resource Languages: A Review of Past Work and Future Challenges." ArXiv abs/2006.07264 (2020).

[30] Anil Kumar Singh. 2008, " Natural language processing for less privileged languages: Where do we come from? where are we going?" In Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages.

[31] Christopher Cieri, Mike Maxwell, Stephanie Strassel, and Jennifer Tracey. 2016. "Selection criteria for low resource language programs.", In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), pages 4543–4549.

[32] Yulia Tsvetkov. 2017, "Opportunities and challenges in working with low-resource languages". Carnegie Mellon University.

[33] Simpson, Heather, Christopher Cieri, Kazuaki Maeda, Kathryn Baker, Boyan Onyshkevych. 2008. Human Language Technology Resources for Less Commonly Taught Languages: Lessons Learned Toward Creation of Basic Language Resources, paper presented at the SALTMIL Workshop: Free/Open-Source Language Resources for the Machine Translation of LessResourced Languages satellite to the 7th International Conference on Language Resources and Evaluation, Marrakesh, May 28-30.

[34] Vamshi KG Reddy, Pratibha Rani, Vikram Pudi, and Dipti M. Sharma. "Decision tree ensemble for parts-of-speech tagging of resource-poor languages", In Proceedings of the 10th annual meeting of the Forum for Information Retrieval Evaluation, pp. 41-47. 2018.

[35] Elaheh Sadredini, , Deyuan Guo, Chunkun Bo, Reza Rahimi, Kevin Skadron, and Hongning Wang. "A scalable solution for rule-based part-of-speech tagging on novel hardware accelerators." ,In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 665-674. 2018.

[36] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language.Computational linguistics, 18(4):467–479.

[37] Jan Buys and Jan A Botha. 2016. "Cross-lingual morphological tagging for low-resource languages", arXiv preprint arXiv:1606.04279.

[38] Ronald Cardenas, Ying Lin, Heng Ji, and Jonathan May. 2019. A grounded unsupervised universal part-of-speech tagger for low-resource languages. arXiv preprint arXiv:1904.05426.

[39] Othman Zennaki, Nasredine Semmar, and Laurent Besacier. 2015. Utilisation des reseaux de neurones recurrents pour la projection interlingue d' ´ etiquettes morpho-syntaxiques a partir d'un corpus parall ` ele. , TALN 2015.

[40] Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. 2017, "Cross-lingual word embeddings for low-resource language modeling", In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 937–947.

[41] Surjya Kanta Daimary & Vishal Goyal & Madhumita Barbora & Umrinderpal Singh,"Development of Part of Speech Tagger for Assamese Using HMM," International Journal of Synthetic Emotions (IJSE), IGI Global, vol. 9(1), pages 23-32, January.(2018).

[42] A. K. Barman, J. Sarmah and S. K. Sarma, "POS Tagging of Assamese Language and Performance Analysis of CRF++ and fnTBL Approaches,"UKSim 15th International Conference on Computer Modelling and Simulation, 2013, pp. 476-479, doi: 10.1109/UKSim.2013.91.

[43] S Navanath Saharia, Dhrubajyoti Das, Utpal Sharma, and Jugal Kalita. 2009. Part of speech tagger for Assamese text. In Proceedings of the ACL-IJCNLP 2009 Conference Short Papers (ACLShort '09). Association for Computational Linguistics, USA, 33–36.

[44] D. Baishya and R. Baruah, "Improving Hidden Markov Model for very low resource languages: An analysis for Assamese parts of speech tagging," 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2021, pp. 142-146, doi: 10.1109/Confluence51648.2021.9377146.

[45] D. Baishya and P. K. Das, "Improving windows tasks recognizer for Assamese using bigram analysis," 2014 International Conference on Audio, Language and Image Processing, 2014, pp. 470-475, doi: 10.1109/ICALIP.2014.7009838.

[46] D. Baishya, R. Baruah and A. Neog, "Present state and future scope of Assamese text processing," 2021 International Conference on Computer Communication and Informatics (ICCCI), 2021, pp. 1-6, doi: 10.1109/ICCCI50826.2021.9402617.

[47] Imad Zeroual, Abdelhak Lakhouaja, Rachid Belahbib "Towards a standard Part of Speech tagset for the Arabic language", Journal of King Saud University-Computer and Information Sciences, vol. 29, pp 171-178,2017.

[48] V. Pirrelli and A. Zarghili, "Arabic Natural Language Processing: Models systems and applications", Journal of King Saud University-Computer and Information Sciences, vol. 29, pp. A1-A3, 2017.

[49] L. Qin, POS tagging of Chinese Buddhist texts using Recurrent Neural Networks, report, 2015.

[50] Plank, A. Søgaard and Y. Goldberg, Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss, arXiv:1604.05529, 2016.

[51] C. D. Santos and B. Zadrozny, Learning character-level representations for part-of-speech tagging, in: Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp. 1818–1826, 2014.

[52] G. Chrupala, Text segmentation with character-level text embeddings, in: Proceedings of the ICML Workshop on Deep Learning for Audio, Speech and Language Processing, 2013.

[53] Y. Kim, Y. Jernite, D. Sontag and A. M. Rush, ,"Character-Aware Neural Language Models",arXiv:1508.06615,2015.

[54] Sangkeun Jung, Changki Lee, and Hyunsun Hwang. 2018. End-to-End Korean Part-of-Speech Tagging Using Copying Mechanism. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 17, 3, Article 19 (May 2018), 8 pages. DOI:https://doi.org/10.1145/3178458.

[55] Prakhar Srivastava, Kushal Chauhan, Deepanshu Aggarwal, Anupam Shukla, Joydip Dhar, and Vrashabh Prasad Jain, "Deep Learning Based Unsupervised POS Tagging for Sanskrit" In Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence (ACAI 2018). Association for Computing Machinery, New York, NY, USA, Article 56, 1–6. DOI:https://doi.org/10.1145/3302425.3302487.

[56] G. Prabha, P. V. Jyothsna, K. K. Shahina, B. Premjith and K. P. Soman, "A Deep Learning Approach for Part-of-Speech Tagging in Nepali Language," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018, pp. 1132-1136, doi: 10.1109/ICACCI.2018.8554812.

[57] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12:2493–2537, 2011.

[58] Dos Santos, Cicero & Zadrozny, Bianca. (2014). Learning Character-level Representations for Part-of-Speech Tagging. 31st International Conference on Machine Learning, ICML 2014.

[59] Verwimp, Lyan & Pelemans, Joris & Van hamme, Hugo & Wambacq, Patrick. (2017). Character-Word LSTM Language Models. 10.18653/v1/E17-1040.

[60] Francis, W. Nelson & Henry Kucera. 1967. Computational Analysis of Present-Day American English. Providence, RI: Brown University Press.