Parts of Speech Tagging for Kannada and Hindi Languages using ML and DL models

Advaith V
Dept. of Machine Learning
B.M.S. College of Engineering, affiliated to Visvesvaraya Technological University
Bengaluru, India
advaith.ai20@bmsce.ac.in

Anushka Shivkumar
Dept. of Machine Learning
B.M.S. College of Engineering, affiliated to Visvesvaraya Technological University
Bengaluru, India
anushka.ai20@bmsce.ac.in

Dr. Sowmya Lakshmi B S
Assistant Professor
Dept. of Machine Learning
B.M.S. College of Engineering affiliated to Visvesvaraya Technological University
Bengaluru, India
sowmyalakshmibs.mel@bmsce.ac.in

Abstract- Part-of-speech (POS) tagging is one of the vital Natural Language Processing (NLP) tasks that entails categorising words in a text (corpus) in accordance with a specific part of the speech, based on the word's context. POS tagging for Indian Languages is not widely explored. Kannada is extremely inflectional and contains one of the most complex and richest collections of linguistic traits. Hence, developing a POS tagger for a resource-poor language such as Kannada is difficult. The morphological complexity of Hindi becomes a challenge despite there having been numerous attempts of building a POS tagger for the language. The proposed work deals with the development of a POS tagger for both Kannada and Hindi by employing Machine Learning (ML) and Deep Learning (DL) algorithms. The results obtained are based on experiments conducted on a corpus consisting of around 3 lakh unique words for Kannada and Hindi combined. The 17 POS tags have been taken from the BIS tag set.

Keywords—Natural Language Processing, Machine Learning, Deep Learning, Part of Speech tagging.

I. INTRODUCTION

A decade ago, nobody would believe that computers could interact with humans as easily as it can today. However, with the rapid increase in complexity of technology, it has been possible for computers to not only communicate but also comprehend text as well as speech, the way humans do. It can automatically complete sentences, correct grammar, and even summarise paragraphs, all with the help of NLP.

An important technique of NLP is POS tagging, a preprocessing method used to categorise words of a sentence and place them under relevant tags depending on the context used. For instance, the same word may be a verb in one sentence, and an adjective in another. It is the task of a POS tagger to identify the part of speech which the words of the sentence belong to, whereupon sentiment analysis, question answering, Named Entity Recognition (NER), or other valid algorithms may be applied. These algorithms can further be used to build chatbots, and virtual voice assistants and can even be implemented for targeted marketing. The vast applications of POS tagging are due to the significant improvement of ML and DL algorithms.

ML and DL are two of the sub-groups of Artificial Intelligence (AI), which is based on the concept that the algorithm shall learn by itself and make appropriate predictions without any human intervention.

A popular example of ML algorithm used for classification is Support Vector Machines (SVM). SVM works on the basic principle that it creates a hyperplane in an N-dimensional space which is then used to tag sample data. The article uses the following ML algorithms: SVM to classify the words into their respective tags, the Decision Tree Algorithm to produce fast and accurate results. "Conditional random fields (CRFs) algorithm is widely known for its appreciating results on the text" [1]. Hence, the proposed work also uses CRF algorithm to classify words based on POS tags by considering the previous words. "SVM is one of the best classifiers, compared to that of Decision Tree and CRF" [2].

ML consists of a special class of algorithms called DL. DL algorithms are able to filter information, and progressively draw conclusions, similar to the working of the human brain. The DL algorithms used to obtain results for this article are Long Short-Term Memory (LSTM), Bi-directional Long Short-Term Memory (BiLSTM), and a combination of CRF with BiLSTM, abbreviated as BiLSTM - CRF.

Both the algorithms are implemented to divide the corpus into an appropriate percentage of a train set and a test set. Once the train set has been fed into the ML and DL algorithms, the program predicts the POS tags of the test set and cross-validates the results with the predefined POS tagged dataset.

The proposed approach uses two Indian languages, one being Hindi, an Aryan language, and another, Kannada, a southern Dravidian language.

Hindi is spoken as a first language in India, more than any other language spoken in the country. Due to India's large population, Hindi is one of the five most-spoken languages in the world. Kannada, on the other hand, is majorly spoken by the people of Karnataka and some of the neighbouring states. It is one of the first thirty languages spoken by most people of the world.

Both Hindi and Kannada are grammatically very rich. However, the morphological scarcity and lack of sufficient tagged datasets makes POS tagging for both these languages particularly difficult.

The article gives an overview of the integration of both DL and ML algorithms to get more accurate and precise results, despite having a limited data pool of the Hindi and Kannada datasets.

The remaining sections of the article are organised as follows: Section II offers a brief outline of previous work on the topic. Section III describes the system architecture in detail, as well as the methodology and tag set. The findings are described, and the article is concluded in section IV.

II. PREVIOUS WORK

Language is the most essential and age-old normal means of communication. One of the most basic text processing jobs in NLP is POS tagging. POS tagging on Indian languages,

2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) | 978-1-6654-9781-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/CONECCT55679.2022.9865745

especially on South Indian Languages, is quite a difficult task due to the lack of proper resources for these languages. A lot of research has been done using NLP with English in fields ranging from sentiment analysis, named entity recognition, and classification to word meaning disambiguation. Upon employing various ML and DL algorithms for POS tagging, accuracies greater than 95% have been achieved. Kannada has gotten far too little attention because of the shortage of high-quality data. The majority of contemporary Kannada POS tagging research have only used classic ML approaches such as SVM or CRF.

Research conducted by Shambhavi and Kumar [3] used the EMILE corpus which consisted of a mixture of various domains. The results showed that the CRF model which they had built obtained an accuracy of 84.58%. This was higher in comparison to the trigram HMM tagger which produced an accuracy of 79.9%.

Antony and Soman [4] used a linear SVM with a 'one vs rest' classification to obtain a state-of-the-art accuracy of 86%.

B. Jayashree et al [5] have proposed a work on extractive summarization techniques using POS tagging. They have used the Hidden Markov Model (HMM) model for tagging. The summarization method involves extracting the key phrase from each sentence and creating a summary.

As a part of NLPAI Machine Learning contest and SPSAL workshop at IJCAI-07 [6], many automated POS taggers for languages such as Bengali, Hindi, and Telugu were developed. Ravindran et al were the first to apply CRF to Hindi POS tagging, achieving a performance of 89.69 percent. Himanshu et al [7] also worked in the same field and obtained an accuracy of 90.89%

Suraksha, Reshma, Shiva Kumar [8] have trained their model with 3000 sentences using algorithms like SVM, CRF, and HMM. They proposed the parsing model for Kannada sentences using Natural Language Tool Kit (NLTK) and their model yielded an accuracy of 96.86% in tagging and chunking.

Ananth, S. S. Bhat and P. S. Venugopala [9] discussed a deep learning-based approach for POS tagging in their paper by utilizing hybrid models of BiLSTM and linear chain conditional random fields (CCRF). Their hybrid model was trained using a dataset of around 15000 manually tagged words which showed promising results of 81.02%.

In their paper [10], Sowmya et al presented a stochastic approach of POS tagging on Malayalam. Their study focused on a very effective method for improving word embeddings in resource-poor languages by leveraging bilingual word embeddings from a different repository.

Another unique approach for POS tagging was developed using Java programming and CRF++ (Yet Another CRF++ toolkit) on Linux platform by Pallavi and Anitha [11]. The results calculated using Perl scripting obtained an accuracy of 99.49%. Upon cross validation using 10 cross validation technique, the system obtained the average accuracy of 55%.

M. Rajani Shree & B. R. Shambhavi used two new approaches for tagging, in their paper [12]. The first one being a supervised machine learning technique CRF++0.50 and the second is a combination of word embedding and deep learning techniques. The accuracy obtained through CRF++0.50 tool was 76% and that with deep learning technique was 71%. The precision, recall and f-score of each tag using both the techniques are appreciable.

Another such paper [13] dealt with building a POS tagger mainly using two kinds of approaches i.e., ML and Neural Networks. The article examined a Kannada POS tagger that could be used on large, interpreted corpora with a high degree of accuracy. Their best-performing CRF model achieved an overall accuracy of 92.31%.

III. METHODOLOGY

A POS tagger's performance is influenced by several factors. Some of these factors include selecting a tagset, collecting the corpora, annotating the corpora, and the size and ambiguity of the corpora. This is further discussed in the section.

A. Corpus Used

The model was trained using a publicly available corpus. The Hindi dataset in the proposed work was sourced from the NLTK website [1]. The Kannada dataset was sourced from various genuine internet sources. Both of the corpora contained sentences from different domains. The corpus was split into a train and test corpus in an 80:20 ratio, with 80 percent of the words used for training and the remaining for testing. Because of its vastness and diversity, the corpora were extremely beneficial in the development of a generic POS Tagger. The Kannada dataset contains around 1 lakh words, and the Hindi dataset contains around 2 lakh words.

B. Preprocessing

The implementation of this model requires the data to be preprocessed. Preprocessing is the alteration of the given data to make it more suitable for the proposed work and to provide greater accuracy when it is given as an input to a ML or DL algorithm. General preprocessing techniques include removal of stop words, numbers, and special characters, all of which can be neglected while feeding it as an input to algorithms.

Preprocessing is required to reduce memory space, but more importantly, it is necessary to remove irrelevant entries from the tag set as these entries may confuse the machine, thereby producing erroneous results.

The Hindi dataset from the above cited website contained many Urdu words and numbers, which had to be removed to get more accurate results. This was done with the help of a code written in Python programming language, which read the file given as an input, searched for the lines containing Urdu letters, and deleted those lines.

Another issue while preprocessing the given dataset was ensuring that the tokens of Hindi as well as Kannada matched. In order to do so, the full forms for the tokens of both the languages were compared to produce a common tag set. Also, the tokens missing from either dataset were deleted using a python code similar to the one used to remove the Urdu words.

C. List of tagset

The tag collection was designed with the cooperation of communities of researchers who had previously worked on Indian languages. The proposed work's tag-list is a table with 18 rows of diverse parts of speech from both languages, along with examples. It's made to make use of ML while also making other NLP processing chores easier.

1. https://www.nltk.org/nltk_data/

TABLE I. LIST OF KANNADA AND HINDI TAGSET USED IN THE CORPUS

Sl. No.	Abbreviation	POS Tag	Example (Kannada)	Example (Hindi)
1.	JJ	Adjective	ಉತ್ತಮ	सम्माननीय
2.	RB	Adverb	ನಿಸ್ವಾರ್ಥವಾಗಿ	तुरंत
3.	QC	Cardinal	ಒಂದು	एक
4.	CC	Conjunction	ಮತ್ತು	और
5.	DEM	Demonstrative	ಈ	यह
6.	INTF	Intensifier	ಅತಿ	सबसे
7.	NEG	Negation	ಅಲ್ಲ	नहीं
8.	NN	Noun	ಅರಮನೆ	घर
9.	NST	Noun denoting spatial expressions	ಮೇಲ್ಗಡೆ	पहले
10.	QO	Ordinal number	ಮೊದಲ	दूसरी
11.	RP	Particles	ಸಹ	ही
12.	PSP	Post Position	ಯನ್ನು	द्वारा
13.	PRP	Pronoun	ಅವನು	इसका
14.	NPN	Proper Noun	ಭಾರತ	भारत
15.	QF	Quantifiers	ಹೆಚ್ಚು	अधिक
16.	WQ	Question words	ಹೇಗೆ	कैसे
17.	RDP	Re-Duplicative	ಪದೇ	अलग
18.	VM	Verb-finite	ನಡೆಸಿಕೊಡುವಾಗ	खेला

TABLE II. PERCENTAGE OF WORDS IN THE CORPUS FOR EACH TAGSET

POS Tag	Word Category Percentage	
	Kannada	Hindi
JJ	8.463	11.514
RB	4.036	0.35
QC	4.408	4.98
CC	5.707	2.992
DEM	2.447	1.151
INTF	0.746	0.3
NEG	0.083	0.173
NN	26.189	30.625
NST	2.721	1.059
QO	0.509	0.299
RP	0.288	1.28
PSP	0.315	21.529
PRP	13.633	4.085
NPN	9.884	8.56
QF	1.287	0.708
WQ	0.34	0.035
RDP	0.03	0.132
VM	18.913	10.228

D. Approaches

Two kinds of approaches have been used for training the POS tagger:

- ML Approaches
- DL Approaches

For the paper, three different ML techniques: SVM, CRF, and Decision trees have been used. SVMs are frequently used to solve a variety of pattern recognition issues and are well-known for their better generalization abilities.

Consider the following set of training data for a two-class problem: [(x_1, y_1),....,(x_N,y_N)], where x_i is the feature vector of the ith sample in the training data and $y_{i+1,-1}$ is the class to which x_i belongs. The goal is to find a decision function that predicts class y accurately from an input vector x. For an input vector, a non-linear SVM classifier produces the decision function f(x)=sign(g(x)), where:

$$g(x)=\sum_{i=1}^{m} w_i\, K(x, z_i) + b \qquad (1)$$

Here, f(x)=1 denotes that x is a member of a specific class, while f(x)=0 denotes that x is not. The representations of training examples are support vectors, or z_i, with m being the number of support vectors. As a result, g(x) has a computational complexity proportional to m.

The second technique employed is CRF, which is a sequence modelling algorithm for identifying entities and patterns in text. While learning a pattern, this model takes future data into account. It is thought to be the best method for entity recognition in terms of performance. The features employed are repeatedly modelled from the data to feed into the CRF. The CRF formula when y is the hidden state and x is the observable variable is given by:

$$p(x|y) = \frac{1}{Z(x)} \sum_{t=1}^{T} \exp\left\{\sum_{k=1}^{K} \theta_k\, f_k(y_t, y_{t-1}, x_t)\right\} \qquad (2)$$

Many ML projects have long employed decision trees because they have a clear structure that provides insight into the training data. They are also easy to conceptualise and apply. The third ML algorithm used, describes a tree-computation algorithm that has been optimised. In ML, decision trees are a well-known classification methodology in which the model is a tree with each node representing a decision and each leaf representing an output class. Although a node can have any number of children, most algorithms use binary trees and binary questions. On the test datasets, the proposed technique is significantly faster than a naive implementation and it produces accurate results.

In the second approach – Deep Learning, the utilized algorithms are LSTM, BiLSTM, and BiLSTM - CRF. LSTM features gating mechanisms to regulate the information that the network saves (and passes on to the next layer) or forgets. LSTM has a clear-cut memory unit that stores information relevant to the learning tasks. The structure of an LSTM cell is such that a back-propagating LSTM network can have a smooth and continuous flow of gradients. This network flow is also known as constant error carousel.

The BiLSTM model is made up of two LSTMs: one that takes input in one way and the other that takes it in the opposite direction. On the input sequence, two LSTMs are trained, one in normal reading order and the other in reverse reading order. In addition to word embeddings (WE), BiLSTM was the first to incorporate character embeddings for tagging. During training, this requires looking at not only the sequence of words in a sentence but also the sequences of characters within those words. The bidirectional LSTM significantly improved accuracy. This demonstrates the versatility of BiLSTM. The time required was, however, nearly double that of a standard LSTM network.

The BiLSTM - CRF is basically a recurrent neural network that combines an LSTM and a CRF. The LSTM processes each text token by token and later builds an intermediate representation, which is then given to the CRF. This further predicts the labels for all of the tokens. The LSTM, being a complex, nonlinear model, can successfully capture the sequential relationships between the input tokens. On the other hand, the CRF allows optimal and joint prediction of all the labels in the sentence, capturing the relationships at the label level. The name "bi" refers to the fact that the LSTM analyses each sentence in both left-to-right and right-to-left directions in order to integrate sequential dependencies in both ways.

IV. RESULTS

One of the performance metrics employed is accuracy, which is the ratio of correctly predicted observations to total observations. It is a useful tool when there are symmetric datasets but with almost identical false-positive and false-negative results. As a result, we've taken into account a variety of other factors in order to assess the proposed model's effectiveness.

Precision refers to the percentage of correctly predicted positive observations among all positive observations in the class, whereas Recall refers to the percentage of correctly predicted positive observations among all observations in the class. Precision is associated with a low percentage of false positives.

The F1 Score is calculated by taking the weighted average of Precision and Recall. As a result, this score takes both false positives and false negatives into account. Despite the fact that it is less intuitive, F1 is frequently more valuable than accuracy.

An example for how the model tags the POS for a Hindi sentence:

जल जीवन के लिए बहुत जरूरी है|

जल: NN	जीवन: NN	के: PSP	लिए: PSP
बहुत: INTF	जरूरी: JJ	है: VM	

TABLE III. RESULTS OBTAINED FOR ML ALGORITHMS FOR HINDI DATASET

Classifier	Accuracy	Precision	Recall	F-measure
SVM	0.782	0.796	0.752	0.773
Decision Trees	0.774	0.710	0.713	0.711
CRF	0.802	0.763	0.734	0.748

TABLE IV. RESULTS OBTAINED FOR ML ALGORITHMS FOR KANNADA DATASET

Classifier	Accuracy	Precision	Recall	F-measure
SVM	0.843	0.85.4	0.858	0.855
Decision Trees	0.801	0.81.3	0.799	0.805
CRF	0.844	0.84.5	0.818	0.831

For the ML algorithms used, the maximum accuracy was attained by the SVM model with an F-measure of 0.855 for Kannada and 0.773 for Hindi.

An example for how the model tags the POS for a Kannada sentence:

ಭಾರತ ಬಹಳ ಸುಂದರವಾದ ದೇಶ. ಇದು ಎರಡನೇ ಅತಿ ಹೆಚ್ಚು ಜನಸಂಖ್ಯೆ ಹೊಂದಿರುವ ದೇಶವಾಗಿದೆ.

ಭಾರತ : NPN	ಬಹಳ : INTF	ಸುಂದರವಾದ : JJ
ದೇಶ : NN	ಇದು : DEM	ಎರಡನೇ : QC
ಅತಿ : INTF	ಹೆಚ್ಚು : QF	ಜನಸಂಖ್ಯೆ : NN
ಹೊಂದಿರುವ : VM		ದೇಶವಾಗಿದೆ : VM

TABLE V. RESULT OBTAINED FOR DL ALGORITHMS FOR HINDI DATASET

Model	Features	Accuracy	Precision	Recall	F-measure
Simple LSTM	non-pre-trained word embeddings	0.71	0.73	0.74	0.734
Bi LSTM		0.70	0.71	0.72	0.714
BiLSTM - CRF		0.81	0.83	0.85	0.839
Simple LSTM	char + word embeddings	0.84	0.83	0.84	0.834
Bi LSTM		0.86	0.84	0.81	0.824
BiLSTM - CRF		0.88	0.85	0.89	0.869

TABLE VI. RESULT OBTAINED FOR DL ALGORITHMS FOR KANNADA DATASET

Model	Features	Accuracy	Precision	Recall	F-measure
Simple LSTM	non-pre-trained word embeddings	0.71	0.73	0.74	0.734
Bi LSTM		0.74	0.76	0.77	0.764
BiLSTM - CRF		0.81	0.83	0.85	0.839
Simple LSTM	char + word embeddings	0.84	0.83	0.84	0.834
Bi LSTM		0.84	0.84	0.81	0.824
BiLSTM - CRF		0.86	0.85	0.85	0.85

Furthermore, an inference that DL models produce better accuracy than ML models can be made since the mean accuracy of the ML algorithms was found to be 0.787, whereas the mean accuracy of the DL algorithms was 0.804. Since Hindi had a larger dataset in comparison to Kannada, the accuracy for Hindi in the BiLSTM - CRF model proved to achieve the maximum accuracy of 0.88.

In the BiLSTM - CRF model, character embedding has been employed along with pre-trained word embeddings, the best F1 score was 0.869 for Hindi and 0.85 for Kannada. The neural networks' accuracy can be enhanced by training them on a larger dataset. The models' discussed in the proposed work can also be used for improving the accuracy of NLP for other languages depending on the dataset and other available resources.

ACKNOWLEDGEMENT

We thank Visvesvaraya Technological University, B.M.S. College of Engineering and IEEE Bangalore section for providing us this opportunity.

REFERENCES

[1] Warjri, S., Pakray, P., Lyngdoh, S. A., & Maji, A. K. (2021). Part-of-speech (POS) tagging using conditional random field (CRF) model for Khasi corpora. International Journal of Speech Technology, 24(4), 853-864.

[2] Pranckevičius, T., & Marcinkevičius, V. (2017). Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. Baltic Journal of Modern Computing, 5(2), 221.

[3] BR, S., & Kumar, R. (2012). Kannada part-of-speech tagging with probabilistic classifiers. international journal of computer applications, 48(17), 26-30.

[4] Antony, P. J., & Soman, K. P. (2010, July). Kernel based part of speech tagger for kannada. In 2010 International Conference on Machine Learning and Cybernetics (Vol. 4, pp. 2139-2144). IEEE.

[5] Jayashree, R., Anami, B. S., & Poornima, B. K. (2021, January). Text Document Summarization Using POS tagging for Kannada Text Documents. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 423-426). IEEE.

[6] Awasthi, P., Rao, D., & Ravindran, B. (2006). Part of speech tagging and chunking with hmm and crf. Proceedings of NLP Association of India (NLPAI) Machine Learning Contest 2006.

[7] Agrawal, H., & Mani, A. (2006). Part Of Speech Tagging and Chunking Using Conditional Random Fields. In Proceedings of the NLPAI ML contest workshop, National Workshop on Artificial Intelligence.

[8] Suraksha, N. M., Reshma, K., & Kumar, K. S. (2017, June). Part-of-speech tagging and parsing of Kannada text using Conditional Random Fields (CRFs). In *2017 International Conference on Intelligent Computing and Control (I2C2)* (pp. 1-5). IEEE.

[9] Ananth, A., Bhat, S. S., & Venugopala, P. S. (2021, November). Grammatical Tagging for the Kannada Text Documents using Hybrid Bidirectional Long-Short Term Memory Model. In *2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)* (pp. 57-60). IEEE.

[10] Soumya, S., Idicula, S. M., & Manju, K. (2009, October). Development of a POS tagger for Malayalam-an experience. In 2009 International Conference on Advances in Recent Technologies in Communication and Computing (pp. 709-713). IEEE.

[11] Pallavi, A. S. P., & Pillai, A. (2014). Parts of speech (pos) tagger for kannada using conditional random fields (crfs). In *Proceedings of the National Conference on Indian Language Computing, NCILC*.

[12] Rajani Shree, M., & Shambhavi, B. R. (2020). POS tagger model for Kannada text with CRF++ and deep learning approaches. *Journal of Discrete Mathematical Sciences and Cryptography*, *23*(2), 485-493.

[13] Todi, K. K., Mishra, P., & Sharma, D. M. (2018). Building a kannada pos tagger using machine learning and neural network models. arXiv preprint arXiv:1808.03175.