JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Gurmukhi Punjabi Part of speech tagging using IndicBERT-BiLSTM architecture

Yogender Kumar[1], Dr Arun Khosla[2], Dr Geeta Sikka[3]
Student[1], Professor[2,3]
Centre for Artificial Intelligence[1]
Dr. B. R. Ambedkar National Institute of Technology, Jalandhar, India[1]

Abstract : Recent advancements in neural network-based language representation have facilitated the transfer of learned internal states from trained models to various downstream natural language processing tasks, including Part of Speech (POS) Tagging and question answering. It has demonstrated the notable improvements achieved by leveraging pre-trained language models, particularly when labeled data is limited. We collected our dataset from the publicly accessible Indian Languages Corpora Initiative (ILCI) phase-II project, which encompasses 28,733 sentences from six diverse text domains, namely science and technology, religion, health, entertainment, sports, and agriculture. The dataset has been meticulously annotated using the Bureau of Indian Standards (BIS) tagset, which includes a comprehensive set of 34 grammatical categories. It consists of 25,859 sentences for training, 1,437 sentences for validation, and 1,437 sentences for the test set. To best of our knowledge this is a first attempt to make a part of speech tagger using transformer architecture for low resource Punjabi language. With only 28733 sentences we have developed a system that achieved an F1 score of 84.46% on unseen data for the POS tagging task from six different domains.

Index Terms **- Part of speech tagging, Transformers, IndicBERT, BiLSTM, Natural language processing**

I. INTRODUCTION

Part-of-speech (POS) tagging is a crucial step in Natural Language Processing (NLP), involving the assignment of syntactic category labels to text tokens. While being essential for tasks like named entity recognition, speech recognition, sentiment analysis, question answering, word sense disambiguation, and chunking [29]. It enables accurate syntactic analysis, semantic understanding, and classification of words for diverse applications. However, challenges persist in developing robust POS taggers for different languages.. The complexity of POS tagging varies depending on the language's characteristics, particularly its morphology. Although languages like English and French have well-developed taggers with high accuracy rates (ranging from 95% to 98%), there is a need to explore recent advancements and address specific challenges in POS tagging. Accurate POS tagging plays a vital role in analyzing sentence structure, meaning, and syntax, encompassing key grammatical components such as nouns, pronouns, adjectives, verbs, and adverbs. This analysis serves as a foundation for developing robust language processing systems, enabling tasks such as information extraction, and machine translation. While extensive research has been conducted on POS tagging for English and several Asian languages, there exists a noticeable research gap in this domain for Indian languages, including Punjabi, necessitating focused investigation and development. However, challenges persist in developing robust POS taggers for different languages.

Punjabi is a language known for its complexity, primarily stemming from its rich system of morphological inflection and flexible word order. The intricate verb conjugation and noun declension patterns, combined with the ability to employ various syntactic constructions, make Punjabi a challenging language to analyze. In addition, POS tagging in Punjabi presents further complexities beyond the absence of capitalization and gender information. Factors such as the extensive use of compound words, verb serialization, and the incorporation of loanwords from other languages contribute to the intricate nature of POS tagging in Punjabi. To bridge this gap, our research centers on POS labeling for Punjabi. Our objective is to contribute to the development of Punjabi-specific language processing techniques that can effectively handle the intricate morphological inflection, flexible word order, and other linguistic complexities of Punjabi. This study investigates the effectiveness of different architectures for sequence tagging for Part of speech task. Specifically, we examine two architectures: the Linear Layer with Softmax and the Bidirectional long short-term memory (BiLSTM) [27] followed by a linear layer with Softmax function. Through our research, we aim to develop a robust part-of-speech labeling system tailored specifically for Punjabi, utilizing state-of-the-art NLP techniques.

II. LITERATURE SURVEY

In the context of Punjabi language, Mittal et al. [1] employed a rule-based approach for POS tagging. For Hindi language, early work began with the development of a partial POS tagger by Ray et al. [2]. Shrivastava et al. [3] further proposed harnessing specific morphological characteristics of Hindi for POS tagging, which was enhanced in [4] using detailed morphological analysis and lexicon lookup. The system achieved an accuracy of 93.45% with a tagset of 23 POS tags. The International Institute of Information Technology (IIIT), Hyderabad, initiated the NLPAI ML contest for Indian languages in 2006, where several teams explored various approaches for POS tagging in Hindi, Bengali, and Telugu. Sankaran Bhaskaran [5] attempted an HMM-based

statistical technique utilizing probability models of contextual features. Ravindran et. Al. [6] and Himanshu et. al. [7] applied CRFs to Hindi, achieving performances of 89.69% and 90.89%, respectively. In the SPSAL workshop during IJCAI-07, IIIT Hyderabad organized a competition on POS tagging and chunking for Hindi, Bengali, and Telugu. The average POS tagging accuracies for developed systems were 73.93%, 72.35%, and 71.83% for Hindi, Bengali, and Telugu, respectively. Aniket Dalal et al. [8] developed a Maximum Entropy Markov Model-based POS tagging system for Hindi, incorporating context-based features, dictionary features, word features, and corpus-based features. Ankur Parikh [9] explored the use of Neural Networks for POS tagging. NLPAI 2006 and SPSAL 2007 participants attempted POS tagging with Bengali in addition to Hindi and Telugu. The highest accuracies obtained were 84.34% and 77.61% for Bengali in the contests, respectively. [10] reported an HMM-based tagger, while [11] presented a Maximum Entropy based tagger. Additionally, [12] and [13] reported taggers based on CRF and SVM, respectively, showcasing their specific findings or performances. Manish Shrivastava & Pushpak Bhattacharyya [14] designed a simple HMM-based POS tagger for Hindi, utilizing the morphological richness of the language and achieving an accuracy of 93.12%.

Research on part-of-speech (POS) tagging techniques has evolved over time, with different approaches being explored for various languages. Initially, rule-based systems were used, despite their limitations in terms of human effort, time consumption, and limited learning potential. Recent research has showcased the effectiveness of deep neural-based POS taggers. For instance, [15] proposed a deep neural model that combined word and character-level representations, achieving high accuracy in English and Portuguese. W. Ling et al. [16] introduced a Bi-LSTM model that effectively captured morphological features, demonstrating strong performance in languages with rich morphology. [17] developed a multilingual POS tagging system using Bi-LSTM, leveraging word embedding and character embedding features, which achieved state-of-the-art results across 22 languages. In the specific context of Punjabi language processing, there have been notable developments. Using a rule-based part-of-speech tagging approach, a grammar checking system was created for Punjabi [18]. Kanwar et al. [19] developed an HMM-based POS tagger for Punjabi, achieving an accuracy of 87.6% on a 4-million-word corpus. The tagger incorporated a Bigram Hidden Markov Model and employed the Viterbi algorithm for implementation. Kaur [20] utilizing an n-gram stochastic method. Results indicate its superior performance compared to rule-based approaches, by creating an annotated corpus using existing rule based methods.

In recent years, the field of natural language processing (NLP) has witnessed a growing emphasis on large-scale pre-trained language models, driven by the emergence of influential models like embeddings from Language Models (ELMO) [21] and Bidirectional Encoder Representations from Transformers (BERT) [22]. These models have demonstrated significant advancements in various NLP tasks, including sentiment analysis, document classification, language translation, text summarization, and question-answering systems. Vaswani et al. [23] presents self-Attention, a mechanism inspired by the human brain's ability to selectively focus on key elements and allocate attention accordingly. It allows deep learning models to identify and highlight the most critical parts by calculating the probability distribution of attention, thereby optimizing traditional deep learning models.

III. RESEARCH METHODOLOGY

In our approach for Punjabi POS tagging, we have utilized a specific architecture illustrated in Figure 1. This architecture consists of two main blocks: Transformer, Bi-LSTM and classification layers.

3.1 Transformer

In recent years, Transformer-based architectures have demonstrated superior performance compared to recurrent neural networks (RNN) and their variants, such as LSTM and GRU, in various natural language processing (NLP) tasks. A Transformer can be regarded as an encoder-decoder-based neural network.

One of the main advantages of Transformer-based architectures is their exclusive reliance on the attention mechanism, eliminating the need for recurrent connections. This characteristic enables parallel processing of the input sequence, resulting in significantly faster training and inference times compared to recurrent models. The absence of recurrent connections also alleviates the issue of vanishing or exploding gradients often associated with recurrent architectures. By leveraging the attention mechanism and parallel processing capabilities, Transformer-based models have achieved remarkable success in a wide range of NLP tasks, including machine translation, text summarization, sentiment analysis, and question-answering systems. The ability of Transformers to capture global dependencies and generate high-quality contextual embeddings has propelled them to the forefront of NLP research. In the BERT model, the input sentences are tokenized using the BERT tokenizer, which splits the sentence into tokens and inserts special tokens like [CLS] and [SEP] in the appropriate positions. The purpose of these tokens is to provide contextual information for downstream tasks. While BERT incorporates tokenization as part of its preprocessing, it is possible to use an external tokenizer if desired. However, it is important to note that the BERT tokenizer will automatically insert the necessary special tokens, so there is no need to explicitly add them. For POS tagging tasks, each token in the input is represented by the corresponding BERT representation, and these representations are fed into fully-connected layers to output the part-of-speech tag for each token. When using a wordpiece tokenizer like the one used in BERT, it is important to establish a 'token mapping' strategy to associate the wordpieces with the appropriate labels. The original BERT paper suggests assigning the word label to the first subword and using a dummy label, denoted as 'X', for the remaining subwords. During the computation of the loss function, the 'X' labels on the sub-tokens are ignored.

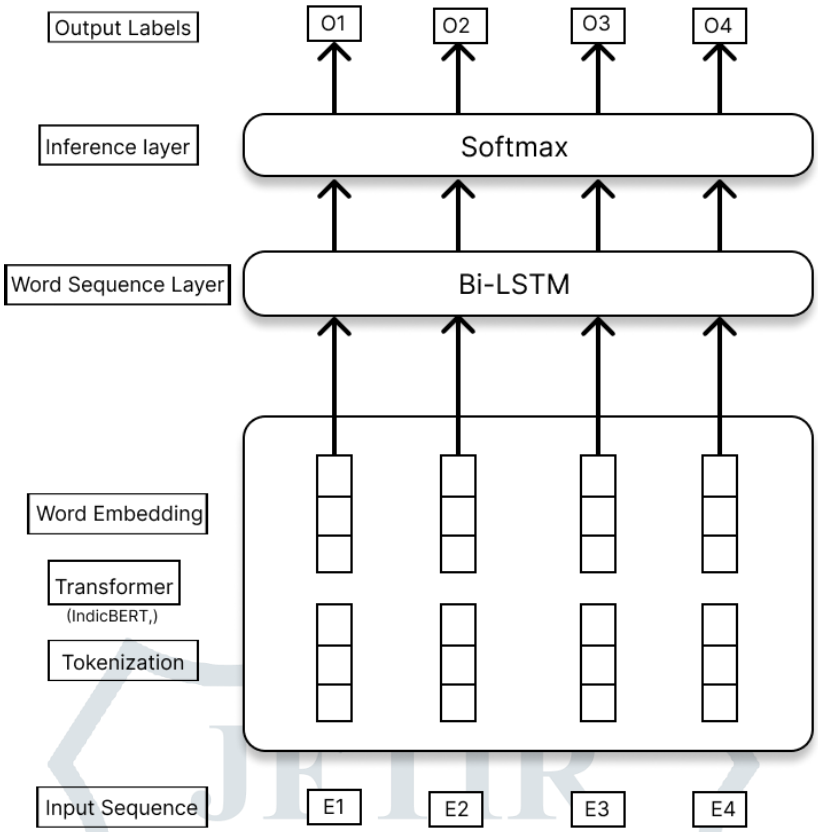

Figure 1 Proposed Architecture for Punjabi POS tagging

Alternatively, the label of the word can be assigned to the last wordpiece representation, or the word label can be propagated to all the subwords, and an average representation of the wordpieces can be computed. The first wordpiece representation is used in this contribution, and other mapping strategies will be studied in the future. These strategies address the challenge posed by wordpiece tokenization and ensure a correspondence between the input tokens and labels. By utilizing the BERT model and adopting appropriate token mapping strategies, accurate POS tagging can be achieved in various NLP tasks.

IndicBERT is a multilingual ALBERT model proposed by Kakwani et al. [24] has been extensively trained on extensive corpora, encompassing 12 significant Indian languages: Assamese, Bengali, English, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, Tamil, and Telugu. This comprehensive training approach enables IndicBERT to effectively process and comprehend diverse linguistic nuances present within these languages. By incorporating a broad range of linguistic variations, IndicBERT demonstrates remarkable proficiency in understanding and generating high-quality text across this multilingual spectrum.

In our research, we utilized IndicBERT, which was trained on a dataset known as IndicCorp. In Table 1 of our research paper, we have provided specific details regarding the pre-trained data used in the model.

Table 1 IndicCorp monolingual corpora statistics: number of sentences, number of tokens in millions (reference [24])

Language	Number of sentences	Number of tokens
Punjabi (pa)	29.2	773
Hindi (hi)	63.1	1860
Bengali (bn)	39.9	836
Odia (or)	6.94	107
Assamese (as)	1.39	32.6
Gujarati (gu)	41.1	719
Marathi (mr)	34.0	551
Kannada (kn)	53.3	713
Telugu (te)	47.9	674
Malayalam (ml)	50.2	721
Tamil (ta)	31.5	582
English (en)	54.3	1220
Total	452.8	8789

During the IndicBERT pre-training phase, a sentence piece tokenizer was trained using the methodology proposed by Kudo and Richardson [25] to effectively tokenize the sentences in each language. The tokenizer was used to tokenize sentences in each language present in the dataset. The resulting tokenized corpora were used to train a multilingual ALBERT [26] model, leveraging the standard masked language model (MLM) objective for effective language modeling. This training process was instrumental in the development of IndicBERT, a highly robust multilingual model with the capability to comprehend and generate text across multiple Indian languages.

3.2 BiLSTM and Linear Layer

The Bi-LSTM block is a recurrent neural network component that captures both the forward and backward contextual information of the input sequence. It is effective in modeling sequential dependencies and has been widely used in various NLP tasks, including POS tagging. The final context vector is obtained by combining the hidden vectors from both LSTM layers, resulting in:

In LSTM, neurons are calculated using Formulas (3)–(8).

$$f_t = \sigma\left(w_f \cdot h_{t-1} + u_f \cdot x_t + b_f\right) \quad (3)$$
$$i_t = \sigma\left(w_i \cdot h_{t-1} + u_i \cdot x_t + b_i\right) \quad (4)$$
$$\tilde{c}_t = tanh\left(w_c \cdot h_{t-1} + u_c \cdot x_t + b_c\right) \quad (5)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (6)$$
$$o_t = \sigma\left(w_o \cdot h_{t-1} + u_o \cdot x_t + b_o\right) \quad (7)$$
$$h_t = o_t \odot \tanh\left(c_t\right) \quad (8)$$

Where i_t, o_t, c_t, h_t and f_t denotes the input gate, output gate, current cell gate, hidden layer and forget gate, respectively. w_o,w_c,w_f and w_i each indicate the weight corresponding to the previous hidden layer h_{t-1}, u_o, u_c, u_f and u_i denotes the weights corresponding to the current input vector x_t; and b_i,b_c,b_o and b_f indicates the relevant bias vector.$\tilde{c}_t$ is the new candidate state vector. σ is the dot product operation, and $\odot$ is the sigmoid activation function.

The classification layers, which follow the Bi-LSTM, play a crucial role in mapping the learned features to the POS tag space. These layers employ techniques such as linear transformations and softmax activation to carry out the classification task. By combining the Bi-LSTM block and the Transformer-based architecture in our approach, we harness the strengths of both models and effectively enhance the performance of Punjabi POS tagging.

In this paper, we explore the fine-tuning of a pre-trained BERT model for POS tagging. Fine-tuning involves adapting the pre-trained model to the specific task at hand. There are different techniques for fine-tuning BERT models, including training the entire architecture, training some layers while freezing others, or freezing the entire architecture and adding untrained layers. In our approach, we adopt the latter technique by freezing all the layers of BERT and extending the architecture with a dense layer and a classification layer. The dense layer serves to capture additional patterns and representations specific to the POS tagging task, while the classification layer with a Softmax activation function outputs the tag sequence of the input sentence. By freezing the BERT layers, we leverage the knowledge captured by the pre-trained model and reduce the risk of overfitting. Only the weights of the added layers are updated during training, allowing them to learn task-specific features. The Softmax activation function in the classification layer enables the generation of probability distributions over the possible tags, facilitating the prediction of the most likely POS tags. Cross entropy loss is used to optimize them in this case.

IV. EXPERIMENTS

4.1 Dataset

The research utilized a publicly accessible corpus obtained through the Indian Languages Corpora Initiative (ILCI) phase-II project, which was initiated by the Ministry of Electronics and Information Technology (MeitY) of the Government of India. The corpus was accessed via the Technology Development for Indian Languages (TDIL) website. The TDIL program, initiated by the Ministry of Communication and Information Technology (MCIT) in 1991, aims to develop resources and tools for major Indian languages. The dataset used in this study was annotated using the Bureau of Indian Standards (BIS) tagset [28]. However, the corpus contained empty lines, untagged sentences, and redundant space characters, requiring preprocessing before training the model. The cleaning process involved removing unwanted words and converting the dataset into the CoNLL format. Following data cleaning and formatting, a well-formatted CoNLL dataset has been produced. The dataset included the word itself and its corresponding part-of-speech tag. Finally, the dataset was split into three sets: training, testing, and validation, to facilitate the training and evaluation of the POS tagging models. In the following tables, we show the statistics for the different datasets and tagsets.

Table 2 Dataset sample

Word	POS Tags
ਇਸ	DM_DMD
ਦੀ	PSP
ਪੈਦਾਵਾਰ	N_NN
ਸਟੋਰ	N_NN
ਕੀਤੀ	V_VM_VF
ਜਾ	V_VM_VNF
ਸਕਦੀ	V_VM_VF
ਹੈ	V_VAUX
।	RD_PUNC

Table 3 Sentence and Word Counts for Punjabi POS corpus

	Punjabi	
	Number	Number

	of sentences	of words
Train	25859	489100
Test	1437	26993
Validation	1437	26907
Total	28733	543000

Table 4 Distribution (Train, Validation, Test) of ILCI Punjabi POS corpus over BIS tagset

POS Tags	Punjabi		
	Train	Validation	Test
N_NN	139855	7685	7558
PSP	74645	4099	4110
RD_PUNC	44073	2425	2424
V_VM_VF	40018	2268	2251
N_NNP	33026	1833	1835
V_VAUX	21272	1139	1201
RB	18967	1086	1035
JJ	18396	999	988
V_VM	14019	770	801
QT_QTC	13304	735	768
CC_CCD	12505	654	712
PR_PRP	8869	463	490
V_VM_VNF	6185	335	338
RP_RPD	6116	315	351
DM_DMD	5515	322	333
CC_CCS	4130	237	206
PR_PRL	3552	197	202
V_VM_VINF	3286	185	178
RP_NEG	3231	199	192
QT_QTF	2929	152	144
DM_DMQ	2816	139	143
QT_QTO	2295	129	124
PR_PRF	2260	122	120
PR_PRQ	825	45	57
RP_CL	604	18	40
RD_RDF	384	25	28
N_NST	364	14	21
V_VM_VNG	194	16	21
RD_ECH	110	5	7
RP_INTF	103	2	4
PR_PRC	67	2	6
N_NNV	26	6	0
RD_UNK	6	0	0
CC_CCS_UT	2	0	0

Table 5 BIS tagset details of Punjabi language

S.No	Category	Type	Tag	Examples
1	Noun	Proper Noun	N_NNP	ਹਰਿਵੰਦਰ ਦੱਲੀ ਤਾਜਮਿਹਲ
		Common Noun	N_NN	ਘਰ ਿਕਤਾਬ ਕਹਾਣੀ ਸਡਕ
		Verbal Noun	N_VNN	
		Abstract Noun	N_ANN	
		Material Noun	N_MNN	
		Noun (Location)	N_NST	ਥੱਲੇ ਅੱਗੇ ਿਪੱਛੇ
		Noun (unclassified)	N_NN	ਘਰ ਿਕਤਾਬ ਕਹਾਣੀ ਸਡਕ
2	Pronoun	Personal	PR_PRP	ਮ ਤੂੰ ਉਹ
		Reflexive	PR_PRF	ਆਪਣਾ ਆਪ ਖੁਦ
		Reciprocal	PR_PRC	ਆਪਸ

		Relative	PR_PRL	ਜੋ, ਿਜਸ ਿਜਹਡਾ, ਜਦ
		Wh-words	PR_PRQ	ਕਣ ਕਦ ਿਕੱਥੇ
		Indefinite	PR_PRI	ਕੋਈ, ਿਕਸ
3	Demonstrative	Deictic	DM_DMD	ਇਹ ਉਹ
		Relative	DM_DMR	ਜੋ ਿਜਸ
		Wh-words	DM_DMQ	ਕਣ
		Indefinite	DM_DMI	ਕੋਈ ਿਕਸ
4	Verb	Auxiliary Verb	V_VAUX	ਹੈ ਸੀ ਸਿਕਆ ਹੋਇਆ
		Main Verb	V_VM	ਆਇਆ ਜਾ ਕਰਦਾ ਮਾਰਗਾ ਰਿਹੰਦਾ
		Non-finite	V__VM__VNF	ਜਦਆਂ ਆਦਆਂ ਕਰਿਦਆਂ ਖਾਕੇ ਜਾਕੇ
		Infinitive	V__VM__VINF	ਿਗਆਂ ਆਇਆਂ ਕਿਰਆਂ
		Gerund	V__VM__VNG	ਜਾਣ ਖਾਣ ਪੀਣ ਮਰਨ
		Transitive	V_VBT	
		In-transitive	V_VBI	
5	Adjective		JJ	ਸੋਹਣਾ ਚੰਗਾ ਮਾਡਾ ਕਾਾ
6	Adverb		RB	ਹੌਂਾ ਕਾਹਲੀ
7	Post Position		PSP	ਨ ਨੂੰ ਤ ਨਾਲ
8	Conjuction	Co-ordinator	CC_CCD	ਅਤੇ ਜ
		Subordinator	CC_CCS	ਿਕ ਜੋ ਤ
9	Particles	Particles (unclassified)	PR_RPD	ਵੀ ਤ ਹੀ
		Classifier	RP_CL	
		Interjection	RP_INJ	ਉਏ ਅਡਆ ਨੀ ਜਨਾਬ
		Negation	RP_NEG	ਨਹ ਨਾ ਿਬਨ ਵਗੈਰ
		Intensifier	RP_INTF	ਬਹੁਤ ਬਡਾ
10	Quantifiers	General	QT_QTF	ਥੋਡਾ ਬਹੁਤਾ ਕਾਫੀ ਕੁਝ
		Cardinals	QT_QTC	ਇੱਕ ਦੋ ਿਤੰਨ
		Ordinals	QT_QTO	ਪਿਹਲਾ ਦੂਜਾ
11	Residuals	Foreign word	RD_RDF	
		Symbol	RD_SYM	$, &, *, (,)
		Punctuation	RD_PUNC	., : ;
		Echowords	RD_ECH	(ਪਾਣੀ-) ਧਾਣੀ (ਚਾਹ-) ਚੂਹ
		Unknown	RD_UNK	

4.2 Evaluation matrix

To evaluate the precision of the model, the precision rate, recall rate, and F1 value are used. Using the following formula:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$
$$Precision = \frac{TP}{TP+FP}$$
$$Recall = \frac{TP}{TP+FN}$$

The POS tags that are correctly identified are denoted by *TP*; POS tags that are incorrectly identified by *FP*; tags that are not identified are denoted by *FN*; Precision is *P*; Recall is *R*. F1 score is the harmonic mean of precision and recall, and it is the balanced F1-score that is commonly used:

$$F1 = \frac{2*Precision*Recall}{Precision+Recall} = \frac{2*TP}{2*TP+FP+FN}$$

4.3 Experiment settings

All Experiments were conducted on Google Colab with a Tesla T4 GPU. The model was implemented using the PyTorch API, and the Adam optimizer was employed. The maximum input sequence length was set to 128, with a fixed LSTM dimension of 200. A batch size of 32 was used during training, with an initial learning rate of 6e-5 and weight decay of .001. Early stopping with a patience of 3 was applied to prevent overfitting. The classifier model consisted of a 1-layer BiLSTM with a hidden size of 256, followed by a Linear layer with a Softmax activation function. For the fine-tuning approach, the BiLSTM layer was jointly updated with a linear layer during training. A model without the BiLSTM layer was also evaluated, optimized by minimizing the cross entropy loss.

IV. RESULTS AND DISCUSSION

We present the performance results of different experiments using our annotated dataset in this section. To conduct the experiments, We use the same training, validation and testing dataset described in the table 3 in all the experiments. This helps us evaluate the performance of these word embeddings on the TDIL dataset. The micro average F1 score, accuracy, Precision and recall on testing dataset and micro F1 score of validation dataset of each tagging model is presented in table 5. By leveraging, IndicBERT word embedding with linear layer achieves the highest F1-score of 84.46% among all the models. The IndicBERT-BiLSTM-linear layer architecture shows comparatively lower tagging performance.

Table 6 shows the classification report of the both the model.

Model architecture	**Performance evaluation (Validation)**	**Performance evaluation (Test)**			
	micro F1 score	**Accuracy**	**F1 score**	**Precision**	**Recall**
IndicBERT- Linear layer	0.8481	0.8466	0.8446	0.8435	0.8466
IndicBERT-BiLSTM-linear layer	0.8466	0.8444	0.8424	0.8417	0.8444

Below are the classification report of both the architectures and its analysis.

	precision	recall	f1-score	support
N_NN	0.8491	0.8598	0.8544	7558
PSP	0.9508	0.9596	0.9552	4110
RD_PUNC	0.9868	0.9851	0.9860	2424
V_VM_VF	0.8192	0.8614	0.8398	2251
N_NNP	0.8194	0.8158	0.8176	1835
V_VAUX	0.9688	0.9575	0.9631	1201
RB	0.6738	0.6686	0.6712	1035
JJ	0.6086	0.5587	0.5826	988
QT_QTC	0.8862	0.9128	0.8993	768
V_VM	0.6333	0.5930	0.6125	801
CC_CCD	0.8482	0.8792	0.8634	712
PR_PRP	0.7970	0.7612	0.7787	490
DM_DMD	0.6925	0.8048	0.7444	333
RP_RPD	0.9249	0.8775	0.9006	351
V_VM_VNF	0.6667	0.5976	0.6303	338
PR_PRL	0.6901	0.7277	0.7084	202
CC_CCS	0.8247	0.7767	0.8000	206
RP_NEG	0.9572	0.9323	0.9446	192
V_VM_VINF	0.4877	0.4438	0.4647	178
DM_DMQ	0.5823	0.6434	0.6113	143
QT_QTF	0.5044	0.3958	0.4436	144
QT_QTO	0.7460	0.7581	0.7520	124
PR_PRF	0.9496	0.9417	0.9456	120
RP_INJ	0.8684	0.8839	0.8761	112
RD_SYM	0.7091	0.8041	0.7536	97
DM_DMR	0.6780	0.4167	0.5161	96
PR_PRQ	0.6721	0.7193	0.6949	57
RP_CL	0.9091	0.7500	0.8219	40
RD_RDF	0.8750	1.0000	0.9333	28
N_NST	0.3333	0.0952	0.1481	21
V_VM_VNG	0.0000	0.0000	0.0000	21
RD_ECH	0.0000	0.0000	0.0000	7
PR_PRC	0.0000	0.0000	0.0000	6
RP_INTF	0.0000	0.0000	0.0000	4
accuracy			0.8466	26993
macro avg	0.6739	0.6583	0.6622	26993
weighted avg	0.8435	0.8466	0.8446	26993

	precision	recall	f1-score	support
N_NN	0.8513	0.8574	0.8543	7558
PSP	0.9481	0.9603	0.9542	4110
RD_PUNC	0.9839	0.9827	0.9833	2424
V_VM_VF	0.8133	0.8552	0.8337	2251
N_NNP	0.8104	0.8104	0.8104	1835
V_VAUX	0.9729	0.9575	0.9652	1201
RB	0.6615	0.6589	0.6602	1035
JJ	0.6022	0.5547	0.5774	988
QT_QTC	0.8931	0.9245	0.9085	768
V_VM	0.6508	0.5793	0.6129	801
CC_CCD	0.8447	0.8708	0.8575	712
PR_PRP	0.7967	0.7837	0.7901	490
DM_DMD	0.6911	0.7928	0.7385	333
RP_RPD	0.9394	0.8832	0.9104	351
V_VM_VNF	0.6614	0.6243	0.6423	338
PR_PRL	0.6384	0.7079	0.6714	202
CC_CCS	0.8163	0.7767	0.7960	206
RP_NEG	0.9323	0.9323	0.9323	192
V_VM_VINF	0.4540	0.4157	0.4340	178
DM_DMQ	0.5515	0.6364	0.5909	143
QT_QTF	0.5354	0.4722	0.5018	144
QT_QTO	0.7287	0.7581	0.7431	124
PR_PRF	0.9500	0.9500	0.9500	120
RP_INJ	0.8448	0.8750	0.8596	112
RD_SYM	0.6607	0.7629	0.7081	97
DM_DMR	0.7174	0.3438	0.4648	96
PR_PRQ	0.7368	0.7368	0.7368	57
RP_CL	0.8378	0.7750	0.8052	40
RD_RDF	0.8710	0.9643	0.9153	28
N_NST	0.5000	0.0476	0.0870	21
V_VM_VNG	0.0000	0.0000	0.0000	21
RD_ECH	0.0000	0.0000	0.0000	7
PR_PRC	0.0000	0.0000	0.0000	6
RP_INTF	0.0000	0.0000	0.0000	4
accuracy			0.8444	26993
macro avg	0.6734	0.6544	0.6557	26993
weighted avg	0.8417	0.8444	0.8424	26993

Figure 2 (a) Classification report of using IndicBERT-Softmax (b) Classification report of using IndicBERT--BiLSTM-Softmax

Based on the comparison of F1-scores between IndicBERT-BiLSTM-Softmax and IndicBERT-Softmax, we can analyze the performance differences for each tag category:

1. Similar Performance:

Tags such as N_NN, PSP, RD_PUNC, V_VM_VF, N_NNP, V_VAUX, V_VM, CC_CCS, RP_NEG, PR_PRF, RP_INJ, RD_RDF, V_VM_VNG, RD_ECH, PR_PRC, and RP_INTF show similar F1-scores for both models, with a difference of less than or equal to 0.01. This indicates that both models perform similarly in predicting these tags.

2. IndicBERT-Softmax Outperforms IndicBERT-BILSTM-Softmax:

Tags such as RD_SYM, V_VM_VINF, DM_DMR and N_NST show higher F1-scores for IndicBERT-Softmax compared to IndicBERT-BILSTM-Softmax. The difference in F1-scores ranges from 0.03 to 0.06, suggesting that IndicBERT-Softmax has better performance in predicting these tags.

3. IndicBERT-BILSTM-Softmax Outperforms IndicBERT-Softmax:

Tags such as QT_QTF, PR_PRQ and PR_PRP show higher F1-scores for IndicBERT-BILSTM-Softmax compared to IndicBERT-Softmax. The difference in F1-scores ranges from 0.02 to 0.06, indicating that IndicBERT-BILSTM-Softmax performs better in predicting these tags.

Overall, both models have similar performance for the majority of tags, but there are specific tags where one model outperforms the other. It is important to consider the specific task requirements and the importance of each tag category when deciding which model to use.

VI. CONCLUSION AND FUTURE WORK

The experimental results indicated that the architecture utilizing a linear layer with pre-trained embeddings exhibited superior performance compared to the BiLSTM-linear layer model in Punjabi POS tagging. This suggests that a simpler architecture with pre-trained embeddings effectively captured the linguistic features of Punjabi words for POS tagging. These findings provide valuable insights into the optimal strategies for achieving accurate and contextually relevant POS tagging in Punjabi. Specifically, the advantage of mono-lingual fine-tuning further contributes to our understanding of the effectiveness of the linear layer architecture. The improved understanding of these findings facilitates the development of accurate and contextually relevant POS tagging systems in Punjabi, benefiting various natural language processing applications.

In the future, it is envisioned that the proposed approach could be extended to other low-resource languages, particularly ethnic minority languages like Assamese and Oriya. Furthermore, there are plans to adapt the architecture to address additional NLP tasks, including entity extraction and sentiment analysis. The aim is to inspire further advancements in low-resource agglutinative languages in the field of NLP.

REFERENCES

[1] Mittal, S., Sethi, N. S., & Sharma, S. K. (2014). Part of speech tagging of Punjabi language using N gram model. *International Journal of Computer Applications*, *100*(19).

[2] A. Basu P. R. Ray, V. Harish and S. Sarkar(2003), "Part of speech tagging and local word grouping techniques for natural language parsing in Hindi", Proceedings of the International Conference on Natural Language Processing (ICON 2003).

[3] S. Singh M. Shrivastava, N. Agrawal and P. Bhattacharya (2005), "Harnessing morphological analysis in pos tagging task", Proceedings of the International Conference on Natural Language Processing (ICON 2005).

[**4]** Smriti Singh, Kuhoo Gupta, Manish Shrivastava, and Pushpak Bhattacharyya (2006),"Morphological richness offsets resource demand – experiences in constructing a pos tagger for Hindi", Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, Sydney, Australia, pp. 779–786.

[5] Sankaran Baskaran (2006), "Hindi POS tagging and Chunking", Proceedings of the NLPAI ML contest workshop, National Workshop on Artificial Intelligence.

[6] Pranjal Awasthi, Delip Rao, Balaraman Ravindran (2006), "Part Of Speech Tagging and Chunking with HMM and CRF", Proceedings of the NLPAI ML contest workshop, National Workshop on Artificial Intelligence.

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

[8] Aniket Dalal, Kumar Nagaraj, Uma Sawant, Sandeep Shelke (2006), "Hindi Part-of-Speech Tagging and Chunking: A Maximum Entropy Approach" Proceedings of the NLPAI ML contest workshop, National Workshop on Artificial Intelligence.

[9] Ankur Parikh (2009), "Part-Of-Speech Tagging using Neural network", Proceedings of ICON-2009: 7th International Conference on Natural Language Processing.

[10] Ekbal, Asif, Mondal, S., and S. Bandyopadhyay (2007) "POS Tagging using HMM and Rule-based Chunking", In Proceedings of SPSAL-2007, IJCAI-07, pp. 25-28.

[11] A. Ekbal, R. Haque and S. Bandyopadhyay (2008), "Maximum Entropy Based Bengali Part of Speech Tagging", Advances in Natural Language Processing and Applications, Research in Computing Science (RCS) Journal, Vol. (33), pp. 67-78.

[12] A. Ekbal, R. Haque and S. Bandyopadhyay (2007), "Bengali Part of Speech Tagging using Conditional Random Field", Proceedings of the 7th International Symposium on Natural Language Processing (SNLP-07), Thailand, pp.131-136.

[13] A. Ekbal and S. Bandyopadhyay (2008), "Part of Speech Tagging in Bengali using Support Vector Machine", Proceedings of the International Conference on Information Technology (ICIT 2008), pp.106-111, IEEE.

[14] Manish Shrivastava, Pushpak Bhattacharyya (2008), "Hindi POS Tagger Using Naive Stemming: Harnessing Morphological Information Without Extensive Linguistic Knowledge", Proceedings of ICON-2008: 6th International Conference on Natural Language Processing.

[15] Cicero Dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In International Conference on Machine Learning. PMLR, 1818–1826.

[16] Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. arXiv preprint arXiv:1508.02096 (2015).

[17] Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. arXiv preprint arXiv:1604.05529 (2016).

[18] Gurpreet Singh, "Development of Punjabi Grammar Checker, Phd. Dissertation, 2008

[19] Kanwar, S., Ravishankar, M., & Sharma, S.K. (2011). POS TAGGING OF PUNJABI LANGUAGE USING HIDDEN MARKOV MODEL.

[20] Kaur, G., & Sharma, D. (2021). Development of Stochastic Part Of Speech Tagger for Morphologically Rich Languages. *International Journal of Research in Engineering and Science (IJRES)*.

[21] Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. arXiv 2018, arXiv:1802.05365.

[22] Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv 2018, arXiv:1810.04805.

[23] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. arXiv 2017, arXiv:1706.03762.

[24] Kakwani, D., Kunchukuttan, A., Golla, S., Gokul, N. C., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020, November). IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 4948-4961).

[25] Kudo, T., & Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.
[26] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.
[**27]** F. Phoneme, "Framewise Phoneme Classification with Bidirectional LSTM Networks," Training, pp. 2047–2052, 2005.
[28] Nitish Chandra, Sudhakar Kumawat, and Vinayak Srivastava. 2014. Various tagsets for indian languages and their performance in part of speech tagging Proceedings of 5 th IRF International Conference. Chennai, 23rd March (2014).
[29] Ruslan Mitkov. 2022. The Oxford handbook of computational linguistics. Oxford University Press.