Multi Task Learning Based Shallow Parsing for Indian Languages

PRUTHWIK MISHRA, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, India
VANDAN MUJADIA, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, India
DIPTI MISRA SHARMA, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, India

Shallow Parsing is an important step for many Natural Language Processing tasks. Although shallow parsing has a rich history for resource rich languages, it is not the case for most Indian languages. Shallow Parsing consists of POS Tagging and Chunking. Our study focuses on developing shallow parsers for Indian languages. As part of shallow parsing, we included morph analysis as well.

For the study, we first consolidated available shallow parsing corpora for **seven Indian Languages** (Hindi, Kannada, Bangla, Malayalam, Marathi, Urdu, Telugu) for which treebanks are publicly available. We then trained models to achieve state-of-the-art performance for shallow parsing in these languages for multiple domains. Since analyzing the performance of model predictions at sentence level is more realistic, we report the performance of these shallow parsers not only at the token level, but also at the sentence level. We also present machine learning techniques for multi-task shallow parsing. Our experiments show that fine-tuned contextual embedding with multi-task learning improves the performance of multiple as well as individual shallow parsing tasks across different domains. We show the transfer learning capability of these models by creating shallow parsers (only with POS and Chunk) for Gujarati, Odia, and Punjabi for which no treebanks are available.

As a part of this work, we will be releasing the Indian Languages Shallow Linguistic (ILSL) benchmarks for 10 Indian languages, including both the major language families Indo-Aryan and Dravidian as common building blocks that can be used to evaluate and understand various linguistic phenomena found in Indian languages and how well newer approaches can tackle them.

CCS Concepts: • **Computing methodologies** → **Natural language processing**; **Language resources**; **Neural networks;**

Additional Key Words and Phrases: Indic languages, shallow parsing, pos tagging, chunking, morph analysis, natural language processing (NLP)

ACM Reference Format:
Pruthwik Mishra, Vandan Mujadia, and Dipti Misra Sharma. 2024. Multi Task Learning Based Shallow Parsing for Indian Languages. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 23, 9, Article 131 (August 2024), 18 pages. https://doi.org/10.1145/3664620

Pruthwik Mishra and Vandan Mujadia contributed equally to this research.

We thank the Ministry of Electronics and Information Technology (MeitY), Government of India, for funding this work. We also express our gratitude to the developers of the treebanks that were developed by a consortium of institutes funded by MeitY, Government of India.

Authors' Contact Information: Pruthwik Mishra, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, Telangana, India; e-mail: pruthwik.mishra@research.iiit.ac.in; Vandan Mujadia, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, Telangana, India; e-mail: vmujadia@gmail.com; Dipti Misra Sharma, MT&NLP Lab,LTRC, IIIT Hyderabad, Gachibowli, Telangana, India; e-mail: dipti@iiit.ac.in.

ACM 2375-4699/2024/08-ART131
https://doi.org/10.1145/3664620

1 INTRODUCTION

Shallow Parsing, or Shallow Linguistic Analysis, is one of the important and basic components in NLP and usually acts as a proxy for Full Parsing (i.e., Deeper Syntactic parsing). Shallow Parsing [1] is an attempt to parse a shallow structure of a text to recover a limited but useful amount of syntactic information. For Indian languages, generally, this information includes Tokenization, POS tagging [47], and Chunking [43]. Morphological analysis [2] is also added as another component in shallow parsing.

Tokenization is a language dependent process of splitting a text into valid sentences and then a sentence into valid tokens. Morphological analysis includes the task of assigning a set of well-defined morphological lemma (root) to the tokens [31] and at least six universal morph tags that characterize the morphology of every word [50] mainly for Indian Languages. They are Coarse Parts-of-speech tag or Lexical Category (lcat), Gender (G), Number (N), Person (P), Case (C) for nouns, and Tense-Aspect-Modality (TAM) for verbs or Vibhakti markers for nouns as mentioned in Reference [44].

Earlier shallow parsing approaches for Indian languages followed a pipeline approach [46] as shown in Figure 1 and along with the final output, intermediate outputs of individual sub-modules are also made available as outputs.

A robust and accurate shallow parser substantially minimizes the complexity of many NLP tasks such as natural language parsing, **named entity recognition (NER)**, machine translation, question answering, dialogue system, evaluation of text generation, and so on. For low-resource machine translation, morphological analysis and shallow features (i.e., POS tags) help reduce sparsity by splitting words into linguistically informed roots and affixes. The translation framework then needs to translate only the lemmas and use the affixes to re-generate the exact inflection of the word in the target language (Sampark [17]). Recent **Neural Machine Translation (NMT)** techniques have shown that the use of shallow linguistic features improves overall translation quality and makes the NMT models converge faster in low resource settings [35, 45].

Following are the major contributions of this article:

- Collect, Analyze, and Consolidate available Shallow parsing corpora in one place for Indian Languages;
- Comparison of different Machine Learning Techniques for shallow parsing;
- **Robust state-of-the-art shallow parsers**[1] for 10 Indian languages that exploit multi-task learning using contextual embedding on multiple text domains;
- Release of **ILSL Benchmarks**[2] for Indic Languages.

2 RELATED WORK

POS tagging is one of the major tasks in shallow parsing. POS Tagging approaches can be classified into rule-based, empirical-based, and hybrid-based [3, 33]. For rule-based approaches, hand-written rules are used to distinguish the POS tag ambiguity. The empirical POS taggers are either example-based or stochastic-based. References [8, 10] presented HMM-based stochastic taggers. References [30, 34, 53, 54] are examples of supervised stochastic taggers where the POS tags are known at the training time. Reference [18] is an unsupervised stochastic tagger that uses a Bayesian approach for parameter estimation. **Conditional Random Field (CRF)** [26, 49] is the

[1]Will be made available.
[2]Benchmarking portal will be made available for ILSL.

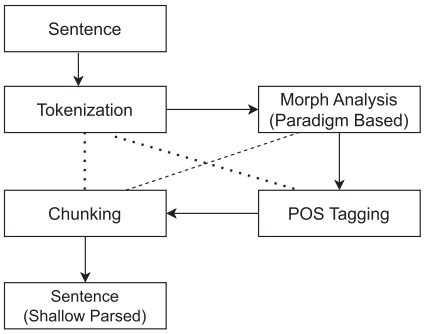

Fig. 1. Pipeline approach in shallow parsing.

most popular machine learning algorithm for POS tagging in Indian languages maximizing the probability of the whole sequence based on the provided features. The features are usually either morphological or context-based.

Though Paradigm-based morph approaches provide high precision and low recall, they suffer from lack of coverage. Rules and data are not exhaustive as well. Moreover, developing rules and data for these approaches requires expert knowledge and intense labor. Unlike paradigm-based morph segmentor, Morfessor [57] is a probabilistic machine learning method that provides morphological segmentation for words of a natural language based solely on provided training corpus efficiently and accurately. Other known statistical morph analyzers include SMA [29], SMA++ [48] for Indian languages. These algorithms predict the **gender, number, person, case (GNPC)**, and the **lemma (L)** of a given token using **Support Vector Machines (SVM)** [39, 55] as a linear classification problem.

Recent advances in deep learning, pre-trained language models such as ELMo [42], ULMFiT [21], OpenAI Transformer [27], BERT [16], and BART [52], and so on, have led to a massive leap in the state-of-the-art performance for many NLP tasks, i.e., text classification [12], natural language inference and question-answering, dialogue system [11], and so on. BERT [16] achieved state-of-the-art results as a pre-trained language model for a wide variety of NLP tasks. It is based on the Transformer [56] model, an attention mechanism that learns contextual relations between words for a given text using two separate mechanisms—an encoder that encodes the text input and a decoder that produces a prediction for the task. All these models are pre-trained on large corpora and then fine-tuned on different downstream tasks. In fine-tuning, the model is trained on an NLP task where it is fed a smaller task-specific dataset. For Indian Languages, MuRIL [25], IndicBERT [24], IndicBART [15] are some of the well-known pre-trained language models that have shown great results.

Following these trends, we explore different deep learning techniques in this context. We did not extend this work to adapters [36] for fine tuning, as its efficacy for natural language understanding is not extensively proven.

3 OUR APPROACH

As a first step towards developing shallow parsers in our study, we consolidated the existing annotated corpora for Indian languages. The corpora are annotated using different schemes. Thus, we developed tagset converters to create data under a single scheme. This provided larger quantities of data and enabled us to experiment with recent deep learning-based contextual learners where the performance directly depends on the amount of data.

Table 1. Details of Publicly Available Annotated Resources for Indian Languages

Lang	Code	Resource	#Sents	#Tokens	Annotation
Assamese	asm	ILCI	69,999	1,010,020	POS
Bangla	ban	ILCI	70,000	1,020,474	POS
		Treebank	15,722	190,859	POS, Chunk
Bodo	bod	ILCI	70,000	958,205	POS
Gujarati	guj	ILCI	70,000	1,062,421	POS
Hindi	hin	ILCI	69,999	1,261,050	POS
		Treebank	29,317	612,017	POS, Chunk, Morph
Kannada	kan	ILCI	45,000	583,869	POS
		Treebank	17,166	189,667	POS, Chunk, Morph
Konkani	kok	ILCI	70,092	982,080	POS
Malayalam	mal	ILCI	69,498	819,631	POS
		Treebank	15,596	176,307	POS, Chunk, Morph
Marathi	mar	ILCI	70,098	878,774	POS
		Treebank	14,299	201,041	POS, Chunk, Morph
Nepali	nep	ILCI	70,000	971,400	POS
Odia	ori	ILCI	24,000	325,351	POS
Punjabi	pan	ILCI	70,010	1,274,837	POS
Tamil	tam	ILCI	70,003	888,929	POS
Telugu	tel	ILCI	70,010	880,890	POS
		Treebank	3,223	17,032	POS, Chunk, Morph
Urdu	urd	ILCI	70,000	1,271,636	POS
		Treebank	7,075	199,211	POS, Chunk, Morph

3.1 Available Corpora

(1) **Treebanks**: Treebanks[3,4,5,6] are richly annotated corpora that represent full syntactic analysis of a sentence. Each sentence is annotated with POS, morph, chunk, named entities, and dependencies. The treebanks for ILs are based on Paninian Grammar formalism [4]. They are represented in **Shakti Standard Format (SSF)** [5].

(2) **POS Annotated ILCI (Indian Languages Corpora Initiative) Corpora**: The ILCI (Indian Languages Corpora Initiative)[7] corpora [22] were developed for 15 major Indian languages including English. The ILCI corpora were developed for domains such as Health, Tourism, Agriculture, and Entertainment.

The "Code" column in Table 1 refers to the ISO 639-2[8] codes defined to represent the names of the languages in column "Lang." Different linguistic resources across languages are distributed under Treebanks and ILCI. Fifteen languages are POS annotated in the ILCI corpora. Treebanks are available for seven languages, which are Bangla, Hindi [7], Kannada, Malayalam, Marathi, Telugu

[3] https://tdil-dc.in/
[4] https://ltrc.iiit.ac.in/showfile.php?filename=downloads/lingResources/newreleases.html
[5] http://ltrc.iiit.ac.in/showfile.php?filename=downloads/kolhi/
[6] http://ltrc.iiit.ac.in/treebank_H2014/
[7] http://tdil-dc.in/
[8] https://en.wikipedia.org/wiki/List_of_ISO_639-2_codes

[37], and Urdu [7]. For this work, we have not considered the annotated corpora, which is available under Universal Dependency Treebanks [38], as they are largely derived from treebanks [51] for considered Indian languages. As we are more interested in modeling shallow parsing tasks in multi-task setting, treebanks are chosen, as they have annotations at POS, Chunk, Morph levels.

3.2 Corpora Cleansing

The quality of annotation varies from language to language. Some of the corpora is also quite noisy. We noticed that the corpora suffers from incorrect, incomplete, duplicate, or other erroneous instances. Therefore, we decided to first fix these errors. We started this task by identifying common errors and then updated the annotations either by updating or replacing the tags. We removed the data where it is impossible to correct. No attempt was made to alter the sanctity of the annotated data; only minor fixes were made. After this step, we got noise-free annotated corpora.

3.3 Corpora Annotation Style

Resources such as the Hindi, Urdu treebanks followed the POS annotation guidelines as defined in ILMT [6], whereas newer treebanks (Bangla, Kannada, Malayalam, Marathi, Telugu) have followed BIS[9] guidelines for POS tagging. Therefore, to consolidate and utilize all available annotated corpora resources, we need to unify tagging standards and convert annotations from one standard/schema to another. This necessitates a tag-set conversion procedure. As BIS standards are defined for all Indian languages for POS tagging, we decided to follow and confine all POS annotations to it. We[10] designed a tagset converter for **ILMT schema to BIS schema** for POS tagging.

3.3.1 Tag-set Conversion: ILMT to BIS. ILMT tagset [6] was developed as a part of **Indian Language to Indian Language Machine Translation (IL-ILMT)** project consisting of 26 POS tags. However, **BIS (Bureau of Indian Standards)** POS Tagset is a hierarchical or layered POS tagset. It has 40+ POS tags that carry more linguistic information at tag level than the ILMT tagset. Hence, the BIS tagset is finer than the ILMT one. Each language uses a subset of the available BIS tags. Additionally, the number of tags in the BIS tagset for Dravidian languages[11] is higher than Indo Aryan languages,[12] as Dravidian languages are morphologically richer than their Indo Aryan counterparts.

Hence, this conversion process is challenging, as it involves a one-to-many mapping. To complete this task, we followed a strategy of creating a mapping between the tags using finite word lists.

Finite Word Lists. In this conversion process, we use two kinds of mapping.

One-to-one. A tag in ILMT tag-set maps to one and only one tag in BIS tag-set. For example: The tag for Common noun in ILMT is "NN," which directly maps to "N_NN" in BIS. The full list of mapping is available in the Appendix.

One-to-many. A tag in ILMT tag-set maps to more than one tag in BIS. For Hindi and Urdu, the closed class POS categories such as conjunctions and pronouns are finite. If a word in a sentence appears in the pre-defined list of words for a particular tag, then that word is assigned the corresponding tag.

[9]Bureau of Indian Standards for POS tagging: http://tdil-dc.in/tdildcMain/articles/134692Draft%20POS%20Tag%20standard.pdf
[10]Includes external contribution.
[11]https://www.britannica.com/topic/Dravidian-languages
[12]https://www.britannica.com/topic/Indo-Aryan-languages

Table 2. Data Splits for Treebanks Data

	ban	hin	kan	mal	mar	tel	urd
Train_Sents	12,576	15,087	13,732	12,476	11,429	2,577	5,659
Train_Toks	152,803	325,710	151,374	140,775	160,825	13,681	158,985
Dev_Sents	1,573	1,866	1,717	1,560	1,430	323	708
Dev_Tokss	19,061	40,762	18,949	18,321	19,640	1,650	20,085
Test_Sents	1,573	12,364	1,717	1,560	1,430	323	708
Test_Toks	18,995	245,545	19,344	17,211	20,576	1,701	20,141

Table 3. Shallow Parsing Sub-tasks Specific Categories

Task	#Categories	Categories
POS Tagging	24	DM_DMD, N_NNP, PSP, RP_INTF, JJ, N_NN, QT_QTC, V_VM, RD_PUNC, PR_PRP, V_VAUX, N_NST, CC_CCD, PR_PRL, RP_RPD, RD_SYM, QT_QTF, PR_PRF, PR_PRQ, RP_NEG, RB, QT_QTO, CC_CCS, RP_INJ
Chunking	11	NP, VGF, BLK, CCP, JJP, VGNN, FRAGP, VGNF, VGINF, RBP, NEGP
root	-	Root form of given word
lcat	11	pn, n, psp, avy, adj, num, v, punc, nst, adv, unk
gender	4	any, m, f, fn, fm, unk
number	4	sg, pl, any, unk
person	7	1, 2, 2h, 3, 3h, any, unk
case	4	d, o, any, unk
vib	-	Vibhakti for each word

For example: मैं (main), तू (Tu), आप (aap) belong to the word list of "PR_PRP" tag. These words will be tagged as "PR_PRP."
The word lists for conversion in both languages and the complete tool for the tag conversion can be found here.[13] After the conversion, we get the data as shown in Table 2.

4 MODEL TRAINING

We divided the shallow parsing task into multiple subtasks as listed in Table 3. Most of these subtasks come under the category of sequence to tag prediction (sequence labeling) except the lemma (root) word prediction. We experimented with different sequence labeling algorithms and present a comparative study about the performance of all the systems across different tasks. For lemma word prediction, we utilized transformer-based sequence to sequence algorithm. Following subsections describe each of these machine learning and deep learning approaches in detail. To the best of our knowledge, we are the first ones to model morph analysis as multiple classification tasks using deep learning techniques for Indian languages.

Table 3 shows the categories for different shallow parsing tasks that we extracted from the available treebanks for Indian languages. We have utilized BIS [40] POS tagsets for POS tagging. For morphological analysis, we used gender, number, person, case, and TAM categories for Hindi as defined in Reference [44]. For chunking, we followed Reference [6]'s categorization.

[13] https://github.com/ltrc/ILMT-To-BIS-Tagset-Converter-For-Hindi-and-Urdu

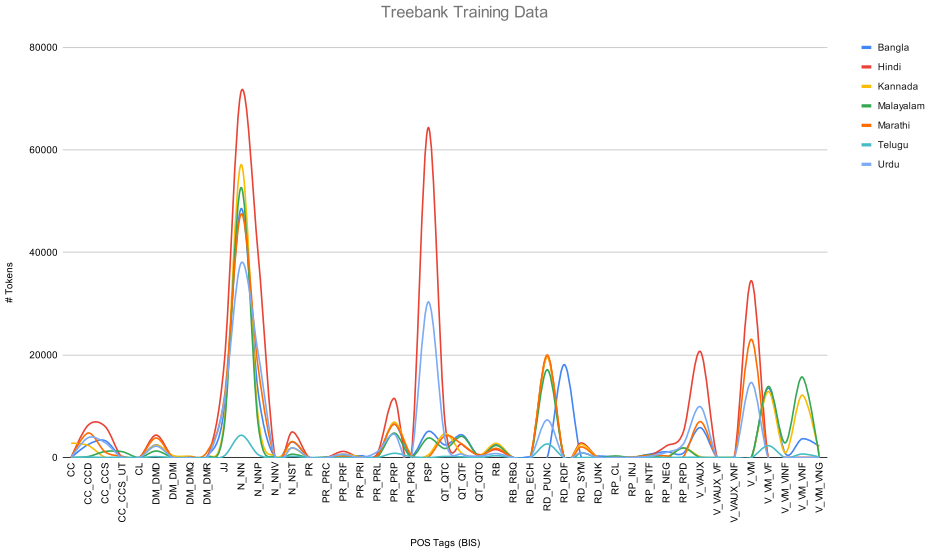

Fig. 2. Treebank training corpora POS tag distribution.

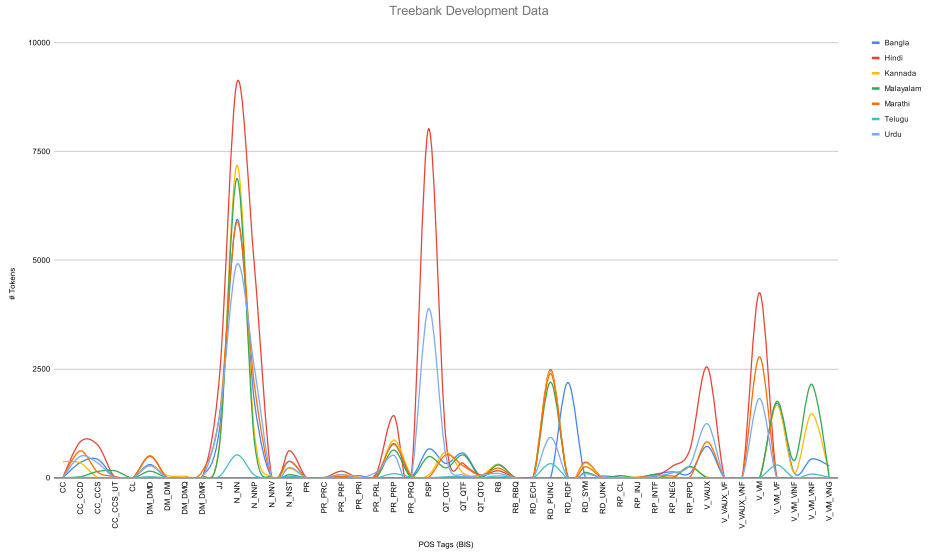

Fig. 3. Treebank development corpora POS tag distribution.

We used training data as mentioned in Table 2. We tested our approaches on news domain 2. We split the dataset into 80:10:10 ratio for training, development (dev), and testing (test). As the Hindi corpora has annotations from different domains, the test set size for Hindi is larger than the dev set. The distribution of POS tags used in our experiments are shown in Figures 2–4.

4.1 Lemma (root) Word Identification

To derive morphological root or lemma for all languages where morph annotations were available, we frame this task as a supervised learning task and applied a sequence-to-sequence model. For

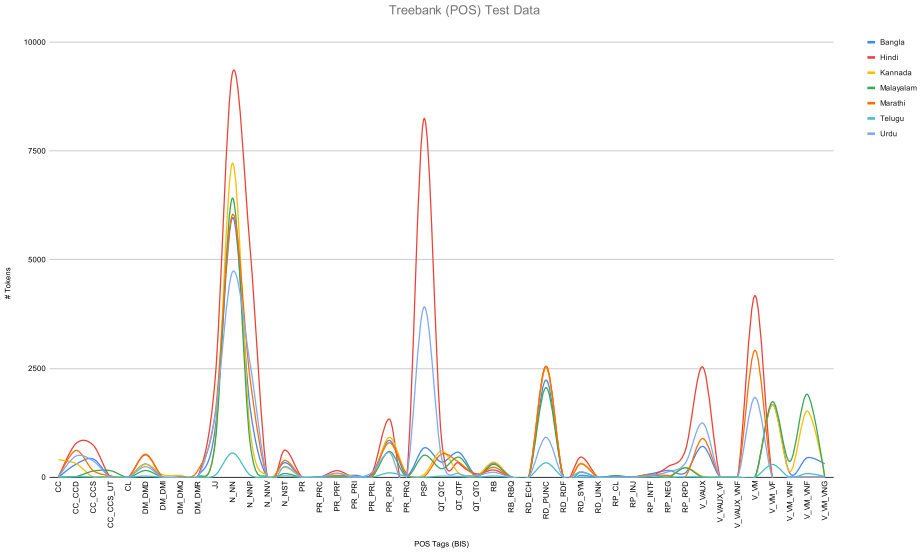

Fig. 4. Treebank test corpora POS tag distribution.

each given word, the main goal of this task is to derive its base form. For example, for a word, राज्यों (states) the task is to derive its root form राज्य (state). We applied character-level sequence-to-sequence learning to predict the correct root. We used the state-of-the-art transformer [56] to model this problem with the following configuration:

— Input : Character sequence
— Embedding size : 512
— Transformer Config : layers (6), hidden_size (2,048) attention heads (8)
— label smoothing : 1.0, dropout : 0.30, Optimizer : Adam
— max_character_sequence_length : 30
— Beam size : 4 (train) and 10 (test)

4.2 Sequence Labeling Approaches

For POS tagging, Chunking and five universal morph tags, i.e., **Gender (G), Number (N), Person (P), Case (C)**, and **Tense-Aspect-Modality (TAM)**, we applied sequence labeling algorithms and following subsections detail them.

4.2.1 Conditional Random Fields. Earlier literature in shallow parsing for Indian languages employed **Conditional Random Fields (CRF)**-based taggers to achieve state-of-the-art results. Hence, we started by training CRF taggers for all the seven languages to set the baseline.

For doing so, we extract specific features from each token in a sentence as detailed below. Except chunking, we used the same feature settings for all other tasks. The features used for modeling the CRF are divided into following two categories.

— For predicting the POS tags and morph features, two kinds of features are used
  — word related features (Prefix, Suffix, Length)
  — a context window of 5 for a word
— For chunking, only context features related to words and POS tags are considered, word and tag bigrams; no word-related features

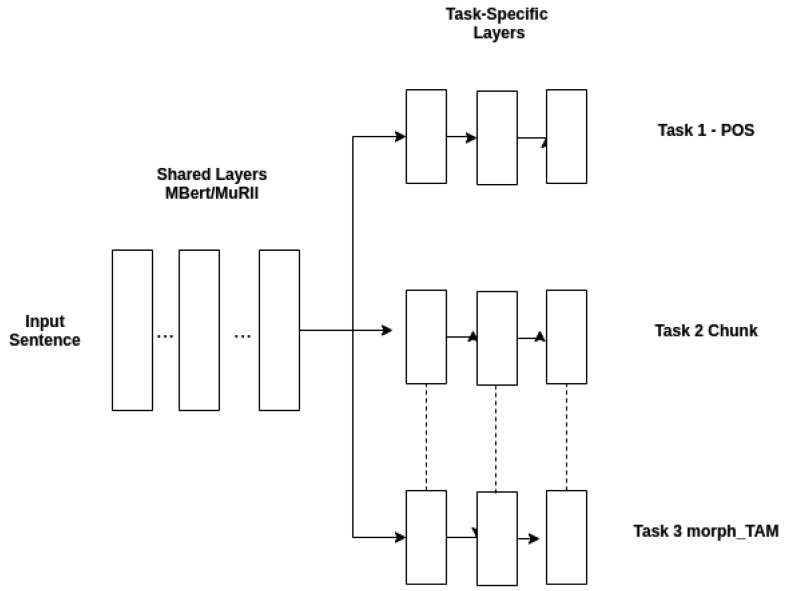

Fig. 5. Multi-task learning.

4.2.2 Neural Network-based Learning. We then trained different neural network architectures for all the tasks to see whether the performance of the shallow parsers can further be improved. We experimented with LSTM [20], **Contextual embedding-based Fine-Tuning with Multilingual Bert (MBERT)**, and MuRIL to see which of these would do the best prediction for each of the tasks. For **LSTM-based Sequence Prediction**, we used a bidirectional LSTM [19] network for predicting the output sequence. As an input to the network, we used pre-trained word2vec embedding [32] that was trained in house on Hindi news articles consisting of 5 million sentences. For other languages, we use the fasttext [9, 23] pre-trained word embeddings, which also uses character n-grams to enrich the word representations. The word2vec embeddings were of 200 dimensions, while the dimensions of pretrained fasttext[14] vectors were 300. For handling the **Out-Of-Vocabulary (OOV)** words, we sum the representation learned from the constituent characters present in a word, also known as C2W [28, 54] with the pre-trained word embedding. The representation is learned from the characters within a word after passing them through a bidirectional LSTM. For implementing this architecture, we used the Keras Deep learning framework [14].

Contextual embedding-based Fine-tuning with Multilingual BERT

For fine-tuning of shallow parsing tasks on contextual embeddings, we used multilingual BERT [16] cased (trained on 102 languages), with 12-layers, 768-hidden, 12-attention head size. For each of the shallow parsing tasks mentioned in Table 3, we applied sequence to tag or Softmax classifier on the pooled layer of multilingual BERT. For each task, the algorithm is back-propagating the gradients to MBERT layers along with task specific softmax layer.

Contextual Embedding-based Fine-tuning with MuRIL

Traditionally, word embeddings have been language-specific. These embeddings are trained for each language separately and these exist in entirely different vector spaces. In our shallow parsing experiments, we explore the use of MuRIL [25], which is a multilingual language model

[14] https://fasttext.cc/docs/en/crawl-vectors.html

Table 4. Token-level F1-scores for Shallow Parsing across Languages Using Single Task Learning

Task	Model	hin		ban		kan		mal		mar		tel		urd	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
POS	CRF	**0.96**	0.9	0.9	0.9	**0.92**	**0.92**	0.92	0.92	0.91	0.91	0.92	**0.93**	0.93	**0.94**
	BiLSTM	0.93	0.87	0.83	0.83	0.86	0.86	0.89	0.89	0.84	0.84	0.83	0.82	0.89	0.9
	MuRIL	**0.96**	**0.93**	**0.93**	**0.92**	**0.92**	**0.92**	**0.93**	**0.93**	**0.93**	**0.93**	**0.93**	**0.93**	**0.94**	**0.94**
	mBERT	**0.96**	0.92	**0.93**	0.91	0.9	0.9	0.91	0.91	0.91	0.9	0.86	0.87	0.93	0.92
Chunk	CRF	**0.96**	**0.9**	**0.93**	**0.92**	**0.92**	**0.91**	**0.94**	**0.94**	0.85	0.84	**0.92**	**0.92**	**0.93**	**0.93**
	BiLSTM	0.91	0.8	0.81	0.81	0.86	0.84	0.87	0.88	0.77	0.77	0.81	0.81	0.84	0.85
	MuRIL	**0.96**	0.89	0.9	0.9	0.9	0.89	0.92	0.92	**0.86**	**0.86**	0.88	0.87	**0.93**	0.9
	mBERT	0.95	0.87	0.86	0.87	0.87	0.85	0.91	0.91	0.85	0.85	0.86	0.86	0.91	0.9
lcat	CRF	**0.98**	0.95	–	–	**0.94**	0.93	0.92	0.93	0.93	0.93	**0.91**	0.9	0.93	0.93
	BiLSTM	0.97	0.94	–	–	0.93	0.93	0.95	0.94	0.91	0.91	0.84	0.84	0.94	0.94
	MuRIL	**0.98**	**0.97**	–	–	**0.95**	**0.94**	**0.95**	**0.94**	**0.94**	**0.94**	**0.91**	**0.91**	**0.94**	**0.94**
	mBERT	**0.98**	0.96	–	–	0.93	**0.94**	0.92	0.92	0.92	0.92	0.86	0.88	**0.94**	0.93
gender	CRF	0.96	0.93	–	–	0.96	0.95	0.95	0.95	0.87	0.87	**0.93**	0.93	0.92	0.92
	BiLSTM	0.93	0.88	–	–	0.94	0.93	0.95	0.95	0.76	0.76	0.87	0.88	0.9	0.9
	MuRIL	**0.97**	**0.94**	–	–	**0.97**	**0.95**	**0.96**	**0.96**	**0.89**	**0.89**	**0.93**	**0.94**	**0.93**	**0.93**
	mBERT	**0.97**	0.93	–	–	0.96	0.94	0.95	0.95	0.85	0.86	0.91	0.9	0.91	0.92
number	CRF	0.96	0.91	–	–	0.92	0.92	0.95	0.95	0.89	0.88	0.9	0.9	0.94	0.94
	BiLSTM	0.95	0.89	–	–	0.9	0.89	0.94	0.95	0.83	0.84	0.83	0.83	0.92	0.92
	MuRIL	**0.97**	**0.93**	–	–	**0.92**	**0.91**	**0.95**	**0.95**	**0.9**	**0.9**	**0.91**	**0.91**	**0.97**	**0.97**
	mBERT	**0.97**	0.92	–	–	0.91	**0.91**	0.94	0.94	0.88	0.88	0.89	0.9	0.95	0.94
person	CRF	0.97	0.94	–	–	**0.92**	0.92	0.95	0.95	0.97	0.97	0.93	0.93	0.96	0.96
	BiLSTM	0.9	0.87	–	–	0.9	0.9	0.96	0.96	0.97	0.97	0.9	0.89	0.94	0.94
	MuRIL	**0.98**	**0.95**	–	–	**0.92**	**0.91**	**0.96**	**0.97**	**0.98**	**0.98**	**0.94**	**0.95**	**0.97**	**0.97**
	mBERT	**0.98**	**0.95**	–	–	0.91	**0.91**	**0.96**	0.96	0.97	0.97	0.91	0.92	0.96	0.96
case	CRF	0.96	0.93	–	–	0.94	0.94	0.96	0.96	**0.91**	0.9	0.93	0.92	0.91	0.9
	BiLSTM	0.96	0.92	–	–	0.95	0.95	0.96	0.96	0.83	0.83	0.9	0.89	0.91	0.9
	MuRIL	**0.97**	**0.94**	–	–	**0.96**	**0.95**	**0.97**	**0.97**	**0.91**	**0.92**	**0.94**	**0.93**	**0.93**	**0.91**
	mBERT	**0.97**	0.93	–	–	0.95	0.95	0.96	0.96	0.89	0.88	0.92	0.91	0.91	**0.91**
vib	CRF	0.98	0.93	–	–	–	–	–	–	–	–	–	–	**0.93**	**0.94**
	BiLSTM	0.97	0.91	–	–	–	–	–	–	–	–	–	–	0.93	0.94
	MuRIL	**0.99**	**0.95**	–	–	–	–	–	–	–	–	–	–	**0.93**	**0.94**
	mBERT	**0.99**	0.94	–	–	–	–	–	–	–	–	–	–	0.91	0.92

specifically built for Indian languages. It is trained on significantly large amounts of Indian text corpora. MuRIL has been trained on 16 Indic languages and English for which monolingual data are publicly available. For fine-tuning each of the tasks mentioned in Table 3, we applied sequence to tag classifier on the pooled output of MuRIL. For each of our tasks, the algorithm is back-propagating the gradients through MuRIL layers along with task-specific output softmax layer.

The results for single task learning can be viewed from Tables 4 and 5.

5 MULTI-TASK LEARNING

Traditionally, shallow parsing is a pipeline approach [46], which makes all shallow parsing subtasks related and a particular single task information can be used in subsequent tasks. This makes the complete shallow parsing task a good candidate for **Multi-Task Learning (MTL)** setup. Instead of building a model for every task, MTL provides a framework to create a single model for all

Table 5. Sentence-level F1-scores for Shallow Parsing across Languages Using Single Task Learning

Task	Model	hin		ban		kan		mal		mar		tel		urd	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
POS	CRF	0.59	0.32	0.4	0.39	0.46	0.46	0.47	0.47	0.39	0.37	**0.64**	0.63	0.33	0.3
	BiLSTM	0.28	0.22	0.21	0.21	0.32	0.33	0.33	0.35	0.18	0.19	0.44	0.42	0.15	0.14
	MuRIL	**0.67**	**0.42**	**0.51**	**0.49**	**0.48**	**0.47**	**0.49**	**0.53**	**0.47**	**0.45**	**0.64**	**0.65**	**0.38**	**0.33**
	mBERT	0.65	0.41	0.5	0.48	0.45	0.46	0.46	0.5	0.43	0.43	0.6	0.61	0.35	0.31
Chunk	CRF	**0.8**	**0.56**	**0.62**	**0.61**	**0.67**	**0.63**	**0.7**	**0.73**	0.39	0.36	**0.79**	**0.74**	**0.58**	**0.57**
	BiLSTM	0.47	0.28	0.32	0.33	0.42	0.41	0.47	0.49	0.25	0.23	0.5	0.47	0.26	0.27
	MuRIL	**0.8**	0.48	0.57	0.57	0.55	0.53	0.6	0.63	**0.41**	**0.42**	0.65	0.62	0.54	0.51
	mBERT	0.77	0.46	0.54	0.53	0.52	0.51	0.58	0.6	0.4	0.39	0.61	0.62	0.5	0.5
lcat	CRF	0.72	0.51	–	–	0.62	0.61	0.55	0.54	0.47	0.47	0.62	0.62	0.3	0.28
	BiLSTM	0.59	0.39	–	–	0.5	0.5	**0.56**	**0.56**	0.34	0.33	0.47	0.43	**0.31**	**0.29**
	MuRIL	**0.77**	**0.57**	–	–	**0.63**	**0.62**	**0.56**	**0.56**	**0.51**	**0.51**	**0.63**	**0.64**	**0.31**	**0.29**
	mBERT	**0.77**	0.56	–	–	0.6	**0.62**	0.5	0.5	0.48	0.49	0.6	0.62	0.3	**0.29**
gender	CRF	0.49	0.37	–	–	**0.73**	0.72	0.66	0.69	0.24	0.24	0.77	0.75	0.2	0.18
	BiLSTM	0.29	0.21	–	–	0.58	0.58	0.61	0.63	0.08	0.08	0.61	0.6	0.13	0.1
	MuRIL	**0.55**	**0.43**	–	–	**0.73**	**0.74**	**0.67**	**0.7**	**0.32**	**0.31**	**0.78**	**0.76**	**0.29**	**0.21**
	mBERT	0.53	0.43	–	–	0.71	0.7	0.63	0.66	0.3	**0.31**	0.75	0.74	0.27	**0.21**
number	CRF	0.51	0.32	–	–	0.47	0.44	0.6	0.62	0.33	0.33	0.68	0.67	0.27	0.24
	BiLSTM	0.46	0.29	–	–	0.38	0.35	0.55	0.58	0.19	0.2	0.44	0.41	0.17	0.19
	MuRIL	**0.67**	**0.4**	–	–	**0.48**	**0.48**	**0.62**	**0.65**	**0.36**	**0.35**	**0.66**	**0.61**	**0.36**	**0.30**
	mBERT	0.65	**0.4**	–	–	0.44	0.43	0.61	0.62	**0.36**	0.33	0.63	0.59	0.31	0.27
person	CRF	0.59	0.42	–	–	0.48	0.47	0.71	0.72	0.77	0.76	0.78	**0.8**	0.43	0.36
	BiLSTM	0.09	0.13	–	–	0.41	0.4	0.64	0.68	0.73	0.7	0.63	0.62	0.27	0.25
	MuRIL	**0.73**	**0.49**	–	–	**0.49**	**0.5**	**0.72**	**0.75**	**0.78**	**0.79**	**0.79**	**0.8**	**0.44**	**0.48**
	mBERT	0.7	0.45	–	–	0.46	0.47	0.71	0.74	0.76	0.76	0.75	0.76	0.43	0.45
case	CRF	0.49	0.36	–	–	0.65	0.65	0.71	0.72	0.34	0.33	0.75	0.74	0.16	0.16
	BiLSTM	0.51	0.35	–	–	0.62	0.59	0.66	0.68	0.18	0.16	0.64	0.63	0.15	0.15
	MuRIL	**0.71**	**0.53**	–	–	**0.67**	**0.66**	**0.72**	**0.76**	**0.4**	**0.4**	**0.76**	**0.75**	**0.26**	**0.22**
	mBERT	0.68	0.52	–	–	0.66	**0.66**	0.7	0.72	0.38	0.37	0.74	**0.75**	0.24	0.19
vib	CRF	0.71	0.38	–	–	–	–	–	–	–	–	–	–	0.34	0.39
	BiLSTM	0.57	0.31	–	–	–	–	–	–	–	–	–	–	0.32	0.31
	MuRIL	**0.79**	**0.39**	–	–	–	–	–	–	–	–	–	–	**0.39**	**0.40**
	mBERT	0.77	0.36	–	–	–	–	–	–	–	–	–	–	0.39	0.39

the related tasks. We enabled multi-task learning for different neural networks: BiLSTM, MBERT, and MuRIL (presented in Figure 5) where contextual embeddings are shared across the subtasks. CRF does not facilitate the joint learning of multiple classification tasks at one go.

MTL aims to improve generalization, strengthens representations, and enables adaptation in machine learning [58] for related tasks. For any machine learning solution, one generally optimizes a particular metric by training a single model to perform the desired task and train the model until its convergence. If this convergence comes from the training signals of related tasks by sharing representations between the tasks, then it enables the model to generalize better on all the related tasks. MTL introduces additional training objectives (a combined training objective) to solve related tasks and to improve the performance on all related tasks [13] by learning training signals for each task. MTL also provides a scope of zero shot learning for never-seen languages. Hence, this architecture is explored to evaluate the performance. The datasets and models are available at https://github.com/ltrc/shallow_parsing_in_indian_languages.git.

Table 6. Token-level F1-scores for Shallow Parsing across Languages Using Multi-task Learning

Task	Model	hin		ban		kan		mal		mar		tel		urd	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
POS	BiLSTM	0.93	0.87	0.83	0.83	0.86	0.86	0.89	0.89	0.84	0.84	0.83	0.82	0.89	0.9
	MuRIL	**0.97**	**0.93**	**0.93**	**0.92**	**0.92**	**0.91**	**0.93**	**0.93**	**0.93**	**0.93**	**0.88**	**0.88**	**0.95**	**0.94**
	mBERT	**0.97**	0.92	**0.93**	0.91	0.9	0.9	0.91	0.91	0.91	0.9	0.86	0.87	0.93	0.92
Chunk	BiLSTM	0.91	0.8	0.81	0.81	0.85	0.84	0.88	0.89	0.78	0.77	0.81	0.81	0.85	0.85
	MuRIL	**0.97**	**0.91**	**0.9**	**0.9**	**0.9**	**0.89**	**0.92**	**0.92**	**0.86**	**0.87**	**0.88**	**0.87**	**0.93**	**0.92**
	mBERT	0.96	0.88	0.88	0.88	0.87	0.85	0.91	0.91	0.85	0.85	0.86	0.86	0.91	**0.92**
lcat	BiLSTM	0.97	0.94	–	–	0.93	0.93	0.95	0.94	0.91	0.91	0.84	0.84	0.95	0.95
	MuRIL	**0.99**	**0.97**	–	–	**0.95**	**0.94**	**0.95**	**0.94**	**0.94**	**0.94**	**0.91**	**0.91**	**0.95**	**0.95**
	mBERT	**0.99**	0.96	–	–	0.93	**0.94**	0.92	0.92	0.92	0.92	0.86	0.88	**0.95**	0.94
gender	BiLSTM	0.93	0.88	–	–	0.94	0.93	0.95	0.95	0.76	0.76	0.87	0.88	0.9	0.9
	MuRIL	**0.97**	**0.94**	–	–	**0.97**	**0.95**	**0.96**	**0.96**	**0.89**	**0.89**	**0.93**	**0.94**	**0.93**	**0.93**
	mBERT	**0.97**	0.93	–	–	0.96	0.94	0.95	0.95	0.85	0.86	0.91	0.9	0.91	0.92
number	BiLSTM	0.95	0.89	–	–	0.9	0.89	0.94	0.95	0.83	0.84	0.83	0.83	0.92	0.92
	MuRIL	**0.97**	**0.93**	–	–	**0.92**	**0.91**	**0.95**	**0.95**	**0.9**	**0.9**	**0.91**	**0.91**	**0.97**	**0.97**
	mBERT	**0.97**	0.92	–	–	0.91	**0.91**	0.94	0.94	0.88	0.88	0.89	0.9	0.95	0.94
person	BiLSTM	0.9	0.87	–	–	0.9	0.9	0.96	0.96	0.97	0.97	0.9	0.89	0.94	0.94
	MuRIL	**0.98**	**0.95**	–	–	**0.92**	**0.91**	**0.96**	**0.97**	**0.98**	**0.98**	**0.94**	**0.95**	**0.97**	**0.97**
	mBERT	**0.98**	**0.95**	–	–	0.91	**0.91**	**0.96**	0.96	0.97	0.97	0.91	0.92	0.96	0.96
case	BiLSTM	0.96	0.92	–	–	0.95	0.95	0.96	0.96	0.83	0.83	0.9	0.89	0.91	0.9
	MuRIL	**0.98**	**0.96**	–	–	**0.96**	**0.95**	**0.97**	**0.97**	**0.91**	**0.92**	**0.94**	**0.93**	**0.93**	**0.92**
	mBERT	**0.98**	0.95	–	–	0.95	0.95	0.96	0.96	0.89	0.88	0.92	0.91	0.91	**0.92**
vib	BiLSTM	0.97	0.91	–	–	–	–	–	–	–	–	–	–	0.93	0.94
	MuRIL	**0.99**	**0.95**	–	–	–	–	–	–	–	–	–	–	**0.94**	**0.95**
	mBERT	**0.99**	0.94	–	–	–	–	–	–	–	–	–	–	0.93	0.94

6 RESULT AND ANALYSIS

For the task of lemma (root) word prediction, we evaluated our transformer-based sequence-to-sequence prediction approach on BLEU score [41] (character level) as well as on word-level (binary evaluation, either entire correct or entire wrong). We achieved overall **94.76** BLEU score on all the listed domains and **92.3%** overall accuracy for the task. We carried out error analysis to understand the problem areas for the task of root word prediction. We found that our approach generalizes root prediction very well. However, sometimes it predicts wrong root form for certain uncommon adjectives and proper nouns. For example, राधे(rādhē), proper noun is predicted राध(rādha) as root word, since the algorithm erroneously generalized the proper noun as the plural form of a noun due to the presence of the suffix "ē."

For POS tagging, Chunking, and six morph-related subtasks, we evaluate our approaches on token as well as sentence level. Sentence-level evaluation becomes necessary when shallow parsing information is used in certain applications such as machine translation. For token-level evaluation, we used F1-score and accuracy. For sentence-level evaluation, we consider positive instance only when we get all token-level predictions correct for a given sentence for a particular task. For Hindi and Urdu, the prediction of TAM or vibhakti can be modeled as a sequence classification task, as the number of possible vibhaktis for these languages is limited (around 20 or so). We can see that for most of the tasks (except chunking) in most domains MuRIL-Multi-task approach is working well. For chunking, CRF works better than most other approaches mainly due to transition probabilities.[15] If we compare MuRIL-Singletask with MuRIL-Multi-task evaluation metrics, then we find

[15] http://aritter.github.io/courses/slides/CRF.pdf

Table 7. Sentence-level F1-scores for Shallow Parsing across Languages Using Multi-task Learning

Task	Model	hin		ban		kan		mal		mar		tel		urd	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
POS	BiLSTM	0.28	0.22	0.21	0.21	0.32	0.33	0.33	0.35	0.18	0.19	0.44	0.42	0.15	0.14
	MuRIL	**0.66**	**0.41**	**0.52**	**0.5**	**0.48**	**0.49**	**0.49**	**0.53**	**0.47**	**0.45**	**0.64**	**0.66**	**0.38**	**0.33**
	mBERT	0.65	0.4	0.5	0.49	0.45	0.47	0.46	0.5	0.43	0.43	0.6	0.61	0.35	0.31
Chunk	BiLSTM	0.47	0.28	0.32	0.33	0.42	0.41	0.47	0.49	0.25	0.23	0.5	0.47	0.26	0.27
	MuRIL	**0.81**	**0.48**	**0.57**	**0.57**	**0.55**	**0.53**	**0.6**	**0.64**	**0.42**	**0.43**	**0.65**	**0.62**	**0.54**	**0.51**
	mBERT	0.77	0.46	0.54	0.53	0.52	0.51	0.58	0.6	0.4	0.39	0.61	0.62	0.5	0.5
lcat	BiLSTM	0.59	0.39	–	–	0.5	0.5	**0.56**	**0.56**	0.34	0.33	0.47	0.43	**0.32**	**0.3**
	MuRIL	**0.78**	**0.57**	–	–	**0.63**	**0.62**	**0.56**	**0.56**	**0.51**	**0.52**	**0.63**	**0.64**	**0.32**	**0.3**
	mBERT	**0.78**	0.56	–	–	0.6	**0.62**	0.5	0.5	0.48	0.49	0.6	0.62	0.31	**0.3**
gender	BiLSTM	0.29	0.21	–	–	0.58	0.58	0.61	0.63	0.08	0.08	0.61	0.6	0.13	0.1
	MuRIL	**0.55**	**0.43**	–	–	**0.74**	**0.74**	**0.67**	**0.7**	**0.32**	**0.31**	**0.78**	**0.76**	**0.29**	**0.21**
	mBERT	0.53	0.43	–	–	0.71	0.7	0.63	0.66	0.3	**0.31**	0.75	0.74	0.27	**0.21**
number	BiLSTM	0.46	0.29	–	–	0.38	0.35	0.55	0.58	0.19	0.2	0.44	0.41	0.17	0.19
	MuRIL	**0.67**	**0.4**	–	–	**0.48**	**0.48**	**0.62**	**0.65**	**0.36**	0.35	**0.66**	**0.61**	**0.36**	**0.30**
	mBERT	0.65	**0.4**	–	–	0.44	0.43	0.61	0.62	**0.36**	0.33	0.63	0.59	0.32	0.28
person	BiLSTM	0.09	0.13	–	–	0.41	0.4	0.64	0.68	0.73	0.7	0.63	0.62	0.27	0.25
	MuRIL	**0.73**	**0.49**	–	–	**0.49**	**0.5**	**0.72**	**0.76**	**0.78**	**0.79**	**0.79**	**0.8**	**0.44**	**0.48**
	mBERT	0.7	0.45	–	–	0.46	0.47	0.71	0.74	0.76	0.76	0.75	0.76	0.43	0.45
case	BiLSTM	0.51	0.35	–	–	0.62	0.59	0.66	0.68	0.18	0.16	0.64	0.63	0.15	0.15
	MuRIL	**0.71**	**0.53**	–	–	**0.67**	**0.66**	**0.72**	**0.76**	**0.4**	**0.4**	**0.76**	**0.75**	**0.26**	**0.22**
	mBERT	0.68	0.52	–	–	0.66	**0.66**	0.7	0.72	0.38	0.37	0.74	**0.75**	0.24	0.19
vib	BiLSTM	0.57	0.31	–	–	–	–	–	–	–	–	–	–	0.32	0.31
	MuRIL	**0.79**	**0.39**	–	–	–	–	–	–	–	–	–	–	**0.40**	**0.41**
	mBERT	0.77	0.36	–	–	–	–	–	–	–	–	–	–	0.39	0.39

that the multi-task setup is working on par or better compared to single-task approach. This points out the effectiveness of multi-task learning for shallow parsing where sub-tasks are dependent on each other. One can also observe from Table 4 that MuRIL and CRF-based approaches work better than MBERT. This suggests that contextual embeddings learned from similar/close languages provide good training signal compared to contextual embeddings learned on all available languages. Similarly, contextual embeddings in MuRIL and MBERT are better in downstream tasks compared to word embeddings used in BiLSTM models. For all the languages, the fasttext word vectors used in BiLSTM (other than Hindi) are trained on limited data sizes of Wikipedia articles, which is the main reason for the subpar performance. From the above tables, we see that the performance of the shallow parsers for all the models on Hindi test set is poorer than the Hindi dev set. This is due to the fact that the test set in Hindi consists of multiple diverse domains such as news, agriculture, conversational, entertainment, judiciary, and tourism. Whereas the dev and test sets in other languages have a similar distribution with respect to domains. MuRIL models significantly outperform CRF on Hindi test set with diverse domains for all the tasks. We also found that a fraction of errors occurred due to the inconsistencies in treeBank annotation. Errors are observed when training and test corpora are annotated using different tagsets. Usually, new tags are added to an existing tagset in new domains for words that cannot be tagged according to the current scheme or guidelines of a tagset, e.g., "RD_BUL" is a new tag introduced for handling bullet points in the "Judiciary" domain that was not present in the news domain. We also observed that all the models perform

Table 8. Results for Vibhakti or Case Marker Prediction for Kannada (kan), Malayalam (mal), Marathi (mar), Telugu (tel)

Language	Type	Micro-F1	Macro-F1	%No Prediction Classes
kan	Dev	0.47	0.01	99
	Test	0.46	0.01	99
mal	Dev	0.83	0.07	89
	Test	0.83	0.08	88
mar	Dev	0.92	0.25	70
	Test	0.92	0.26	69
tel	Dev	0.87	0.35	58
	Test	0.84	0.39	54

significantly well on in-domain data, while their performances degrade marginally when tested on out-of-domain data.

All the morph feature prediction tasks are modeled as sequence tagging tasks. However, for morphologically rich languages such as Kannada, Telugu, Malayalam, and Marathi, the number of vibhakti markers is quite high (more than 100). Also, the data we have for these languages is relatively smaller in size. This makes it infeasible to model vibhakti prediction as a sequence classification task. Initial experiments for these languages show that many of the vibhakti classes are not getting predicted at all as shown in the column "%No Prediction Classes" in Table 8. Additionally, we can infer from this table that micro F1 scores for vibhakti prediction can be misleading, while macro F1 scores present a more realistic figure for this task. We can conclude from these observations that vibhakti prediction for morphologically richer languages cannot be modeled as a sequence tagging task.

Sentence-level results are presented in Tables 5 and 7. For all the tasks over all the domains, we see that MuRIL-multi-task performs significantly better than CRF and other models. The multi-task models either match the performance of the single-task models or outperform them. This indicates that contextual embedding provides better sentence-level representations for these shallow linguistic tasks for all languages. This particular observation also suggests that shallow linguistic tasks would serve as good benchmarks to evaluate learned representations, which led us to create ILSL benchmarks for Indian languages. Our results suggest that high F1 scores at token level do not guarantee similar performance at sentence level.

Morph-level annotations are not available in the Bangla Treebank. So, the experiments on morph features are not conducted for Bangla.

7 ZERO SHOT LEARNING

We extended this work to three Indian languages, Gujarati, Odia, and Punjabi, from Indo Aryan family, by evaluating the zero shot learning capabilities of our shallow parsers. These languages are selected as they show varying degree of richness in terms of morphology. Our multi-tasking-based shallow parsers are used for the predictions of multiple subtasks in these languages. For this task, we developed manually annotated evaluation test sets for Gujarati (guj), Odia (ori), and Punjabi (pan). Our evaluation included 200 POS and Chunk annotated datasets in these three languages.

Zero learning experiments on three languages provide baseline systems for these languages. Although Gujarati, Odia, Punjabi belong to the same language family, the POS tagset defined for them varies. Odia is morphologically richer compared to Gujarati and Punjabi. The tagset for Odia is similar to Dravidian languages. The tagsets for the other two languages resemble that of Hindi.

Table 9. F1-scores for Zero Shot Learning in Gujarati (guj), Odia (ori), and Punjabi (pan)

guj			ori			pan		
POS	Chunk	Model	POS	Chunk	Model	POS	Chunk	Model
0.72	**0.81**	**Hindi**	**0.68**	**0.58**	**Bangla**	**0.7**	**0.65**	**Hindi**
0.62	0.54	Marathi	0.61	0.52	Kannada	0.55	0.44	Marathi
0.58	0.55	Urdu	0.58	0.46	Malayalam	0.58	0.53	Urdu
–	–	–	0.58	0.41	Telugu	–	–	–

Based on the tagset that is used for annotation, we experimented with Kannada, Malayalam, Telugu, and Bangla shallow parsers for Odia POS, Chunk predictions, while the Hindi, Marathi, and Urdu shallow parsers were utilized for Gujarati and Punjabi. We observe that the F1-scores at token level for morphologically richer Odia is lower than Gujarati and Punjabi, which are morphologically closer to Hindi. Table 9 presents a clear picture of language similarities. The zero-shot results of Odia using the Bangla shallow parser outperforms the shallow parsers of Dravidian languages. A similar trend is also observed for Gujarati and Punjabi, where Hindi shallow parser performs the best. In all the languages, the sentence-level accuracies are poor (ranging from 5%–15% in POS and Chunk tasks) indicating that fine-tuning on language-specific data (as shown in Tables 4, 6, 5, 7) is superior to zero shot learning.

8 CONCLUSION AND FUTURE WORK

In this article, we demonstrated that contextual embedding-based fine-tuning on multi-task setup not only improves the performance of any shallow linguistic task on the same domain, but also for multiple tasks across multiple domains. We report overall token level as well as sentence-level improvement for multiple domains using MuRIL. As a future work, one can aim to try contextual embedding with CRF to improve overall Chunking results.

Starting with the languages (seven languages) for which treebanks are available and three resource-poor languages, we put out the **Indian Languages Shallow Linguistic (ILSL)** benchmarks as common building blocks and diagnostic gathered dataset. These datasets can be helpful to evaluate, understand, and analyze a wide range of linguistic phenomena found in Indian languages and how well newer deep learning-based approaches can handle them considering the morphological richness of Indian languages. Our shallow parsers can be used to bootstrap data for other resource-poor Indian languages, as shown by the zero shot learning experiments. As a continuous effort, we aim to include different task-specific datasets from multiple diverse Indian languages to this benchmark.

APPENDIX

A ONE-TO-ONE MAPPING FOR INDO-ARYAN LANGUAGES

Table 10 shows the mapping from ILMT to BIS tags used for Hindi and Urdu.

Table 10. One-to-one Mapping

Tag Name	ILMT Tag	BIS Tag
Common Noun	NN	N_NN
Locative and Temporal Noun	NST	N_NST
Proper Noun	NNP	N_NNP
Classifier	CL	RP_CL
Post position	PSP	PSP
Adverb	RB	RB
Adjective	JJ	JJ
Foreign Word	FW	RD_RDF
Demonstrative	DEM	DM_DMD
Particle	RP	RP_RPD
Finite/Main Verb	VM	V_VM
Auxiliary Verb	VAUX	V_VAUX
Question Word	WQ	PR_PRQ
Quantifier	QF	QT_QTF
Cardinal	QC	QT_QTC
Ordinal	QO	QT_QTO
Intensifier	INTF	RP_INTF
Interjection	INJ	RP_INJ
Negation	NEG	RP_NEG
Echo Word	ECH	RD_ECH
Unknown	UNK	RD_UNK

REFERENCES

[1] Steven P. Abney. 1992. Parsing by chunks. Principle-Based Parsing. *Studies in Linguistics and Philosophy*, Vol. 44, Springer, Dordrecht. https://doi.org/10.1007/978-94-011-3474-3_10

[2] Ankita Agarwal, Pramila, Shashi Pal Singh, Ajai Kumar, and Hemant Darbari. 2014. Morphological analyser for Hindi-A rule based implementation. *Int. J. Advan. Comput. Res.* 4, 1 (2014), 19.

[3] P. J. Antony and K. P. Soman. 2011. Parts of speech tagging for Indian languages: A literature survey. *Int. J. Comput. Applic.* 34, 8 (2011), 0975–8887.

[4] Rafiya Begum, Samar Husain, Arun Dhwaj, Dipti Misra Sharma, Lakshmi Bai, and Rajeev Sangal. 2008. Dependency annotation scheme for Indian languages. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing*.

[5] Akshar Bharati, Rajeev Sangal, and Dipti M. Sharma. 2007. SSF: Shakti standard format guide. *Lang. Technol. Res. Centre, Int. Instit. Inf. Technol., Hyderabad, India* (2007), 1–25.

[6] Akshar Bharati, Rajeev Sangal, Dipti Misra Sharma, and Lakshmi Bai. 2006. AnnCorra: Annotating corpora guidelines for POS and chunk annotation for Indian languages. *LTRC-TR31* (2006), 1–38.

[7] Riyaz Ahmad Bhat and Dipti Misra Sharma. 2012. Dependency treebank of Urdu and its evaluation. In *Proceedings of the 6th Linguistic Annotation Workshop*. 157–165.

[8] Phil Blunsom. 2004. Hidden Markov models. *Lecture Notes, August* 15, 18-19 (2004), 48.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Computat. Ling.* 5 (2017), 135–146.
[10] Thorsten Brants. 2000. TnT—A statistical part-of-speech tagger. *arXiv preprint cs/0003055* (2000).
[11] Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's GPT-2—How can I help you? Towards the use of pretrained language models for task-oriented dialogue systems. *arXiv preprint arXiv:1907.05774* (2019).
[12] Berfu Büyüköz, Ali Hürriyetoğlu, and Arzucan Özgür. 2020. Analyzing ELMo and DistilBERT on socio-political news classification. In *Proceedings of the Workshop on Automated Extraction of Socio-political Events from News 2020*. 9–18.
[13] Rich Caruana. 1997. Multitask learning. *Mach. Learn.* 28, 1 (1997), 41–75.
[14] François Chollet. 2015. Keras. Retrieved from https://keras.io
[15] Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. 2022. IndicBART: A pre-trained model for natural language generation of Indic languages. In *Findings of the Association for Computational Linguistics*.
[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[17] Sanjay K. Dwivedi and Pramod P. Sukhadeve. 2010. Machine translation system in indian perspectives. *J. Comput. Sci.* 6, 10 (2010), 1111.
[18] Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. 744–751.
[19] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional LSTM networks for improved phoneme classification and recognition. In *Proceedings of the International Conference on Artificial Neural Networks*. Springer, 799–804.
[20] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[21] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146* (2018).
[22] Girish Nath Jha. 2010. The TDIL program and the Indian Langauge Corpora Intitiative (ILCI). In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)*.
[23] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).
[24] Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N. C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of EMNLP*.
[25] Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. MuRIL: Multilingual representations for Indian languages. *arXiv preprint arXiv:2103.10730* (2021).
[26] John Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).
[27] Jieh-Sheng Lee and Jieh Hsiang. 2020. Patent claim generation by fine-tuning OpenAI GPT-2. *World Patent Inf.* 62 (2020), 101983.
[28] Wang Ling, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1520–1530. DOI: https://doi.org/10.18653/v1/D15-1176
[29] Deepak Kumar Malladi and Prashanth Mannem. 2013. Statistical morphological analyzer for Hindi. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*. 1007–1011.
[30] Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 171–189.
[31] Alec Marantz. 1997. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. *Univ. Pennsyl. Work. Papers Ling.* 4, 2 (1997), 14.
[32] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[33] Pruthwik Mishra, Vandan Mujadia, and Dipti Misra Sharma. 2017. POS tagging for resource poor languages through feature projection. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON'17)*. 50–55.
[34] Pruthwik Mishra and Dipti Misra Sharma. 2022. Building Odia shallow parser. *arXiv preprint arXiv:2204.08960* (2022).
[35] Vandan Mujadia and Dipti Sharma. 2020. NMT based similar language translation for Hindi–Marathi. In *Proceedings of the 5th Conference on Machine Translation*. Association for Computational Linguistics, Online, 414–417. Retrieved from https://www.aclweb.org/anthology/2020.wmt-1.48

[36] Nandini Mundra, Sumanth Doddapaneni, Raj Dabre, Anoop Kunchukuttan, Ratish Puduppully, and Mitesh M. Khapra. 2023. A comprehensive analysis of adapter efficiency. *arXiv preprint arXiv:2305.07491* (2023).
[37] Sneha Nallani, Manish Shrivastava, and Dipti Misra Sharma. 2020. A fully expanded dependency treebank for Telugu. In *Proceedings of the 5th Workshop on Indian Language Data: Resources and Evaluation (WILDRE'20)*. 39–44.
[38] Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16)*. 1659–1666.
[39] William S. Noble. 2006. What is a support vector machine? *Nat. Biotechnol.* 24, 12 (2006), 1565–1567.
[40] Atul Ku Ojha, Pitambar Behera, Srishti Singh, and Girish N. Jha. 2015. Training & evaluation of POS taggers in Indo-Aryan languages: A case of Hindi, Odia and Bhojpuri. In *Proceedings of 7th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*. 524–529.
[41] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 311–318.
[42] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.
[43] Avinesh P. V. S. and G. Karthik. 2007. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. *Shallow Pars. South Asian Lang.* 21 (2007), 21–24.
[44] Rajeev Sangal, Vineet Chaitanya, and Akshar Bharati. 1995. *Natural Language Processing: A Paninian Perspective*. PHI Learning Pvt. Ltd.
[45] Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. *arXiv preprint arXiv:1606.02892* (2016).
[46] Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Srivastava, Radhika Mamidi, and Dipti M. Sharma. 2016. Shallow parsing pipeline for Hindi-English code-mixed social media text. *arXiv preprint arXiv:1604.03136* (2016).
[47] Manish Shrivastava and Pushpak Bhattacharyya. 2008. Hindi POS tagger using naive stemming: Harnessing morphological information without extensive linguistic knowledge. In *Proceedings of the International Conference on NLP (ICON'08)*. Citeseer.
[48] Saikrishna Srirampur, Ravi Chandibhamar, and Radhika Mamidi. 2014. Statistical morph analyzer (SMA++) for Indian languages. In *Proceedings of the 1t Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*. 103–109.
[49] Charles Sutton and Andrew McCallum. 2012. An introduction to conditional random fields. *Found. Trends® Mach. Learn.* 4, 4 (2012), 267–373.
[50] John Sylak-Glassman, Christo Kirov, David Yarowsky, and Roger Que. 2015. A language-independent feature schema for inflectional morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 674–680.
[51] Juhi Tandon, Himani Chaudhry, Riyaz Ahmad Bhat, and Dipti Misra Sharma. 2016. Conversion from Paninian Karakas to universal dependencies for Hindi dependency treebank. In *Proceedings of the 10th Linguistic Annotation Workshop Held in Conjunction with ACL 2016 (LAW-X'16)*. 141–150.
[52] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. *arXiv preprint arXiv:1905.05950* (2019).
[53] Sajeetha Thavareesan and Sinnathamby Mahesan. 2020. Word embedding-based part of speech tagging in Tamil texts. In *Proceedings of the IEEE 15th International Conference on Industrial and Information Systems (ICIIS'20)*. IEEE, 478–482.
[54] Ketan Kumar Todi, Pruthwik Mishra, and Dipti Misra Sharma. 2018. Building a Kannada POS tagger using machine learning and neural network models. *arXiv preprint arXiv:1808.03175* (2018).
[55] Vladimir Vapnik, Isabel Guyon, and Trevor Hastie. 1995. Support vector machines. *Mach. Learn* 20, 3 (1995), 273–297.
[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).
[57] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. Aalto University publication series SCIENCE + TECHNOLOGY, 25/2013. Aalto University, Helsinki, (2013).
[58] Joseph Worsham and Jugal Kalita. 2020. Multi-task learning for natural language processing in the 2020s: Where are we going? *Pattern Recog. Lett.* 136 (2020), 120–126.

Received 30 May 2023; revised 24 December 2023; accepted 30 April 2024