

A comparative analysis of PoS tagging tools for Hindi and Marathi

Pratik Narayanrao Kalamkar, Prasadu Peddi, Yogesh Kumar Sharma

Department of Computer Science and Engineering, Shri Jagdishprasad Jhabarmal Tibrewala University, Jhunjhunu, India

Article Info

Article history:

Received Oct 4, 2024

Revised Jul 11, 2025

Accepted Oct 7, 2025

Keywords:

Computational linguistics

Machine learning

Natural language processing

Part-of-speech tagging

Text analytics

Tokenization

ABSTRACT

Many tools exist for performing parts of speech (PoS) data tagging in Hindi and Marathi. Still, no standard benchmark or performance evaluation data exists for these tools to help researchers choose the best according to their needs. This paper presents a performance comparison of different PoS taggers and widely available trained models for these two languages. We used different granularity data sets to compare the performance and precision of these tools with the Stanford PoS tagger. Since the tag sets used by these PoS taggers differ, we propose a mapping between different PoS tagsets to address this inherent challenge in tagger comparison. We tested our proposed PoS tag mappings on newly created Hindi and Marathi movie scripts and subtitle datasets since movie scripts are different in how they are formatted and structured. We shall be surveying and comparing five parts of speech taggers viz. IMLT Hindi rules-based PoS tagger, LTRC IIIT Hindi PoS tagger, CDAC Hindi PoS tagger, LTRC Marathi PoS tagger, CDAC Marathi PoS tagger. It would also help us evaluate how the Bureau of Indian Standards's (BIS) tag set of Indian languages compares to the Universal Dependency (UD) PoS tag set, as no studies have been conducted before to evaluate this aspect.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Pratik N. Kalamkar

Department of CSE, Shri Jagdishprasad Jhabarmal Tibrewala University

Jhunjhunu, Rajasthan, India

Email: pratik2kcn@gmail.com

1. INTRODUCTION

In the field of natural language processing (NLP), part-of-speech (PoS) tagging is an important task that significantly impacts later linguistic processing stages. It involves marking each word, punctuation, and symbol in a sentence with its corresponding PoS grammatical category—such as noun, verb, and adjective—based on its meaning and context. PoS tagging provides structural and contextual information that helps in understanding the text accurately in the later stages of syntactic parsing, NER, chunking, and semantic analysis. PoS tagging is particularly challenging for morphologically rich and varied languages spoken in India, like Hindi and Marathi, where words undergo complex inflections based on gender, number, and case. The task becomes even more challenging and complex due to the limited availability of ready-made annotated corpora and linguistic resources for these languages, compared to widely studied languages like English [1]. PoS tagging possibility is more complex than it looks in this situation because of code-switching, colloquialism, and lack of linguistic resources for this kind of data [2]. Code-switching occurs when text is switched from one language to another. Colloquialism refers to informal expressions, slang, and non-standard grammatical constructs that often differ from formal language norms. The lack of linguistic resources, including

scarce or non-existent PoS-tagged datasets for less-documented languages or dialects, significantly limits the availability of supervised training options for developing accurate models. This paper presents a comprehensive comparative performance and accuracy evaluation of several PoS taggers developed for Hindi and Marathi languages. The focus is on their performance across various dimensions, such as accuracy, speed, robustness, and the ability to handle different levels of PoS granularity. The taggers selected for examination are widely available and popular tools, including the IMLT Hindi rules-based PoS tagger, LTRC IIIT Hindi PoS tagger, CDAC Hindi PoS tagger, LTRC IIIT Marathi PoS tagger, and CDAC Marathi PoS tagger. These taggers adopt a diverse range of methodologies, from rule-based systems to machine learning (ML)-based models, thereby providing a comprehensive view of the research landscape.

A central aspect of this study is the evaluation of the taggers across different levels of PoS granularity, particularly when mapped to the Universal Dependencies (UD) PoS tag set. PoS tagging systems mostly use language-specific tag sets, which may vary in their level of granularity (e.g., differentiating between proper nouns and common nouns or grouping them under a general “noun” category) from tagger to tagger. By mapping these outputs of different taggers to the common UD tag set, we can perform a more uniform and significant comparison across taggers, even if their original tag sets differ in complexity detail or number. This analysis allows us to assess the taggers’ flexibility to adjust to different levels of linguistic abstraction, which is especially important in cross-linguistic and multilingual NLP applications. To further underscore the practical application of the research, we test these PoS taggers on movie scripts and subtitles, which present unique challenges not typically found in traditional, grammatically formal corpora. Movie scripts and subtitles contain informal dialogues, scene descriptions, quoted text, and action lines, which may pose additional difficulties for PoS taggers designed with more structured text in mind [3]. We use a movie script dataset for Hindi PoS tagger evaluation; this consists of scripts of 100 different Hindi movies released in 2018. For the Marathi PoS tagger evaluation, we use a subtitle dataset consisting of 100 subtitles belonging to 100 films released in 2010.

This paper aligns with the goals of advancing NLP techniques for underrepresented languages, specifically Hindi and Marathi, by providing a comparative analysis of existing PoS tagging tools. It bridges the gap between academic research and practical applications by focusing on linguistically rich and computationally challenging datasets like movie scripts and subtitles. This alignment ensures that the study contributes both to the theoretical understanding and practical enhancement of PoS tagging performance for Indian languages.

2. DATA PREPARATION AND PRE-PROCESSING

Following data pre-processing steps were followed for creating a dataset of Hindi movie scripts,

- Step 1: web crawling: websites such as filmcompanion.in and scribd.com were crawled using tools such as Open Web Scraper and Scrapy to collect Hindi movie scripts.
- Step 2: formatting issues: scripts were found in various formats, including Devanagari, Phonotonic Hindi, English, PDFs, and scanned copies, due to the lack of uniform scriptwriting standards in the Hindi movie industry.
- Step 3: manual typing: scanned copies of scripts were manually typed into Devanagari Hindi text.
- Step 4: spell checking: proper spell checks were performed on English and Hindi electronic texts using Microsoft Word macros to create a cleaner dataset.
- Step 5: language conversion: scripts in Phonotonic Hindi and English were converted into Devanagari Hindi using Google Translate.
- Step 6: UTF-8 conversion: all scripts were converted to Unicode UTF-8 format to ensure compatibility with the PoS taggers being evaluated.
- Step 7: text standardization: regular expressions were used in notepad++ to arrange the scripts in a uniform text format. This includes steps like removing unwanted characters and page numbers and converting each sentence to a new line, as shown in Figure 1.
- Step 8: dataset overview: the final dataset consists of 100 Hindi movie scripts across various genres since 2018, containing a total of 170,744 lines and 1,029,826 words, stored as UTF-8 formatted text files in Devanagari Hindi.
- Step 9: Python script for analysis: a Python script was written to calculate the total number of lines and words. Lines were counted by reading each file line by line, while words were counted by splitting each line based on whitespace.

For the Marathi language movies dataset, it was even more challenging to get a dataset, as no significant readily available scripts were present. Hence, in Step 1, we gathered subtitles (.srt) of Marathi movies in different languages and manually translated them into Marathi language using Google Translate and later verified with language experts. Language expert verification would further reduce the chances of errors after automatic translation. One hundred Marathi movies of different genres have been selected since 2010. Lahoti *et al.* [4] extensively surveyed Marathi language NLP tasks, especially for resources, tools, and state-of-the-art techniques. However, this survey showed a lack of research on Marathi subtitles.

Later, in Step 2, time stamps were removed from subtitle files. To remove time stamps, we processed text using regex (regular expressions) in Python by removing lines from subtitle files that have only numbers and symbols like : (colon) and (,) comma. We are left with only dialogues, sound effect words, time cues, background noise words, and non-verbal communication. We kept these words to improve the richness of our subtitles like a script. Step 3, since the original text we are using here was in subtitle format, the sentences (dialogues) were spread across different lines. To combine multiple lines of the same sentence into a single line, we used regex on text files. The use of Regex in Notepad++ would combine the sentences that are split across multiple lines into a single line by matching termination symbols viz. (dot), ? (question mark), ! (exclamation) which are similarly used in Marathi as in the English language. We created a dataset of 100 different Marathi movie subtitles, which finally is in the form of 100 text files in UTF-8 with Marathi text in Devnagari after pre-processing, as illustrated in Figure 2. These contain a total of 153,565 lines and 851,377 words. The number of lines and words is counted using Python script, just like we did for Hindi script files.

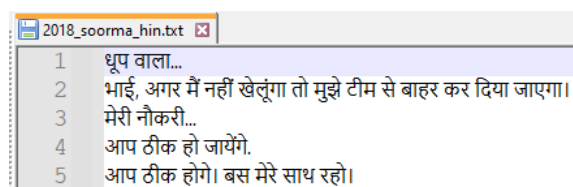


Figure 1. Sample snippet of Hindi movie script

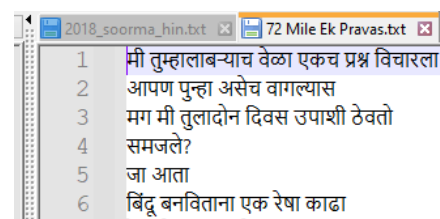


Figure 2. Sample snippet of Marathi movie script

3. RELATED WORK

Various studies have compared different PoS tagging techniques across domains, each emphasizing the performance of diverse algorithms. One of the earliest studies in PoS tagger comparison by Kumawat and Jain [5] talks about various developments in PoS taggers and PoS-tag-set for the Indian language; they talk about the application Trigram and hidden Markov models (HMM) methods on Hindi text to measure performance accuracy of different PoS taggers available. Chiplunkar *et al.* [1] conducted a comparative evaluation of PoS taggers using HMM across multilingual corpora consisting of Hindi and Marathi. Their study demonstrated the impact of linguistic diversity on tagging accuracy, emphasizing the varying performance of each model in different linguistic settings.

Horsmann *et al.* [6] conducted a comparative analysis of 22 PoS tagger models for English and German, sourced from nine different implementations. By evaluating these models on a variety of corpora spanning different domains, they simulate a “black-box” scenario in which researchers select a PoS tagger based on factors such as popularity or ease of use and subsequently apply it to diverse types of text. This approach focuses on assessing the performance of the taggers across various text types. Jacobsen *et al.* [7] investigate the trade-off between model size and performance in ML-based NLP, proposing methods to compare the two. Their case study on part-of-speech tagging across eight languages identifies classical taggers as optimal in balancing size and performance. In contrast, deep models, such studies lack Hindi and Marathi languages.

Specific to Indian languages, Talukdar and Sarma (2023) [8] traces the evolution of automatic PoS tagging for Indo-Aryan languages from dictionary-based and rule-based methods to ML and deep learning (DL) models. Their review highlights the superior performance of ML and DL-based taggers, with reported accuracies reaching up to 97%, and emphasizes the role of customized DL models and pre-processing methods in enhancing performance [8]. In their work on PoS tagging for the Khasi language, the authors employ the conditional random field (CRF) method, achieving a testing accuracy of 92.12% and an F1-score of 0.91.

This study contributes to the development of a Khasi PoS corpus, which is crucial for building lemmatizers and supporting various NLP applications for the language [9]. PoS tagging plays a pivotal role in many NLP applications. Still, while English has well-established taggers, no such resources exist for Marathi, a language with complex morphology and regional variations. The research provides a thorough exploration of different PoS tagging models for Marathi, addressing challenges such as ambiguity, inflectional structure, and free word order while proposing solutions to improve tagging accuracy [10].

PoS tagging for languages like Marathi, with complex morphology, presents significant challenges, which have been addressed through rule-based techniques, HMMs, and hybrid models. Despite the promise of ML and DL approaches, limited annotated data remains a key obstacle, as highlighted in recent studies [11]. Talukdar *et al.* [12] performed a critical review on PoS and Universal PoS (UPoS) in low-resource languages like Hindi. Computation for Indian Language Technology, Indian Institute of Technology, Bombay has created two taggers, a Hindi PoS tagger and a hybrid PoS tagger. The first one is the Hindi PoS tagger, which is a CRF-based PoS tagger for the Hindi language. This tagger uses CRF-based open-source tool kit CRF++. It is available only in Hindi. The post-tagged file contains the PoS-tagged text in Shakti standard format (SSF).

LTRC IIIT Hindi shallow parser, developed at machine translation and NLP lab at Language Technology Research Center, IIIT-Hyderabad. The center has undertaken work in various sub-areas of NLP, viz. Syntax, parsing, semantics, word sense disambiguation, discourse, tree banking, and machine translation. The center has developed a computational Paninian grammar (CPG) framework for Indian languages. The focus of research in the lab includes computational grammatical models, machine translation, parsing, semantics, and dialogue and discourse analysis. The critical component studied here is the working of a shallow parser for Hindi and Marathi languages. The shallow parsing here consists of a chunker and morphological analysis and includes SSF, as described earlier in the ILMT study. This one does not provide a direct PoS tagger; this is a shallow parser but is provided for both Marathi and Hindi. But this shallow parser provides debug and fast modes, where we can get intermediate outputs at various stages like tokenization and PoS tag. We used this feature to get PoS tags. Therefore, we shall refer to it as the LTRC IIIT Hindi/Marathi PoS tagger. The LTRC IIIT Hindi PoS tagger intermediate output is in wx format (format developed by IIT); hence, we have an extra step of converting that into meaningful UTF-8 encoding. We have commented out steps after the PoS tagger in this parser source code, as they are not required for analyzing the PoS tagger; this shall give us an accurate measurement of the PoS tagger provided by this shallow parser.

CDACM Hindi and Marathi PoS taggers, developed knowledge based computer systems division of CDAC Mumbai, provides PoS model for Hindi and Marathi languages, trained on Stanford parts of speech log-linear model. However, the model, although trained using Stanford PoS tagger, is trained on a tweet dataset of Hindi mixed with English [2]. PoS tool provides a Linux sh script that takes the input file and provides an output file with PoS tags. It is developed with pre-processing followed by plain training with Stanford PoS tagger. Using a maximum entropy method, this PoS tagger learns a log-linear conditional probability model from tagged text. The PoS tag of the input word is then decided by the model based on the context and surrounding tags.

Stanford provides a variety of NLP tools, including PoS tagger, which is a Java implementation of the log-linear part-of-speech taggers [13]. This PoS tagger comes with three trained models for English, two for Chinese, two for Arabic, and one for French, German, and Spanish. This tool has been distributed with a library named Stanza and has pre-trained language models for Hindi and Marathi. The Stanford Hindi and Marathi PoS taggers use Universal PoS tags. The English tagger here uses PennTreebank Tag set [14]. An attention-based model using transfer learning achieved 93.86% accuracy on the Hindi disease dataset, demonstrating the potential of domain adaptation for PoS tagging in low-resource domains [15].

This study offers a comprehensive comparative analysis of five PoS taggers for Hindi and Marathi, focusing on performance metrics such as accuracy, robustness, and speed on unique datasets, including Hindi movie scripts and Marathi subtitles. A standardized mapping between tag sets and the UD PoS tag set is proposed, enabling uniform comparisons across taggers. By evaluating these tools on informal, structurally diverse datasets, this work highlights their adaptability to real-world text. Additionally, insights into robustness across token sizes and computational trade-offs are provided, offering practical guidance for low-resource language NLP. The findings contribute significantly to advancing PoS tagging techniques for Indian languages.

4. METHOD AND RESULTS

We will see how these five mentioned parts of speech taggers Viz. IMLT Hindi rules based PoS tagger, LTRC IIIT Hindi PoS tagger, CDAC Hindi PoS tagger, LTRC IIIT Marathi PoS tagger, and CDAC Marathi PoS tagger compared with Stanford PoS tagger when we use pre-processed scripts from Hindi movies, and subtitles from Marathi movies as dataset.

We shall compare the speed, accuracy, and robustness of these PoS taggers with the Stanford PoS taggers. The reason for choosing the Stanford PoS tagger as the gold standard for comparing other PoS taggers is its wide acceptance and lesser granular tag set when compared to these other PoS taggers. This would help us to have common minimum tags when we compare.

Speed evaluation: for measuring the speed performance of these PoS taggers, we supplied these PoS taggers with our pre-processed dataset consisting of Hindi movie scripts and Marathi movie subtitles. Hindi dataset is processed on the IMLT Hindi rules-based PoS tagger, LTRC IT Hindi PoS tagger, CDAC Hindi PoS tagger, and Stanford Hindi PoS tagger (Stanza). Marathi dataset is processed on LTRC Marathi PoS tagger, CDAC Marathi PoS tagger, and Stanford Marathi PoS tagger (Stanza). Stanza, introduced by Manning *et al.* [16], supports a wide variety of NLP tasks for Indian languages, including PoS tagging for Hindi and Marathi. Various Linux shell and Python scripts were created that would suitably run these PoS taggers on batch files. In order to measure processing time and memory utilization, we used `/usr/bin/time` command. We run each of the PoS taggers for 10 iterations; this would help us eliminate random variation due to several factors, such as background processes, memory management, and CPU scheduling. This would also ensure that the final speed measurement is not skewed by an outlier, giving a better measure of consistency and stability of the PoS tagger. All these speed tests were performed on Ubuntu 20.04, on the same hardware, i.e., Core i7 processor with 12 GB RAM. Different speed metrics were measured. Processing time is calculated using the Lapsed wall clock time taken to finish PoS tagging of the given dataset. Memory utilization is measured in maximum resident set size (RSS), which indicates the peak physical memory used by a process during execution. It is critical for evaluating memory efficiency, identifying potential memory leaks, and optimizing resource allocation. High RSS values may cause performance degradation, especially on systems with limited RAM.

Figure 3 presents a detailed comparison of different metrics for four different Hindi PoS taggers: IMLT Hindi rules-based PoS tagger, LTRC IIIT Hindi PoS tagger, CDAC Hindi PoS tagger, and Stanford Hindi PoS Tagger (Stanza). The metrics include the processing time (in seconds) as shown in Figure 3(a), tokens per second in Figure 3(b), and memory usage (in kilobytes) in Figure 3(c). Each tagger's performance is evaluated over 10 iterations to assess average processing time, tokens per second, and memory utilization.

Table 1 presents the average performance and standard deviation for four Hindi PoS taggers evaluated over 10 iterations. The IMLT Hindi rules-based tagger demonstrated the highest token processing speed (1,137.4 tokens/sec) with low memory usage (676,614 kB) and minimal variation, making it the most efficient in terms of speed. In contrast, the LTRC IIIT Hindi tagger showed extremely low speed (9.12 tokens/sec) but used the least memory overall (16,276 kB). The CDAC Hindi tagger, however, offered a reassuringly balanced performance. It boasted a high processing speed (917.9 tokens/sec) and moderate memory usage (415,494 kB), providing a reliable option for PoS tagging tasks. The Stanford Hindi tagger (Stanza), while exhibiting relatively good speed (71.6 tokens/sec), consumed significantly higher memory (4.94 GB), indicating a trade-off between model complexity and resource consumption.

Table 2 summarizes the average performance and standard deviation of three Marathi PoS taggers across 10 iterations. The LTRC IIIT Marathi tagger recorded the slowest processing speed (1.59 tokens/sec) with the lowest memory usage (247,631 kB), indicating a lightweight but computationally intensive implementation. The CDAC Marathi tagger demonstrated significantly better efficiency, processing 436.3 tokens/sec with moderate memory consumption (431,050 kB), making it the most balanced performer in this group. In contrast, the Stanford Marathi tagger (Stanza) achieved a moderate speed of 28.16 tokens/sec. However, it required over 5.3 GB of memory, highlighting a substantial resource demand that underscores the need for optimization in DL architectures.

Figure 4 presents a detailed comparison of three different Marathi PoS taggers: LTRC IIIT Marathi PoS tagger, CDAC Marathi PoS tagger, and Stanford Marathi PoS tagger (Stanza) on 100 Marathi movies dataset. The metrics include processing time (in seconds) as shown in Figure 4(a), total tokens generated and tagged, tokens per second in Figure 4(b), and memory performance MSS (in kilobytes) in Figure 4(c). Each tagger's performance is evaluated over 10 iterations to assess average processing time, tokens per second, and memory utilization.

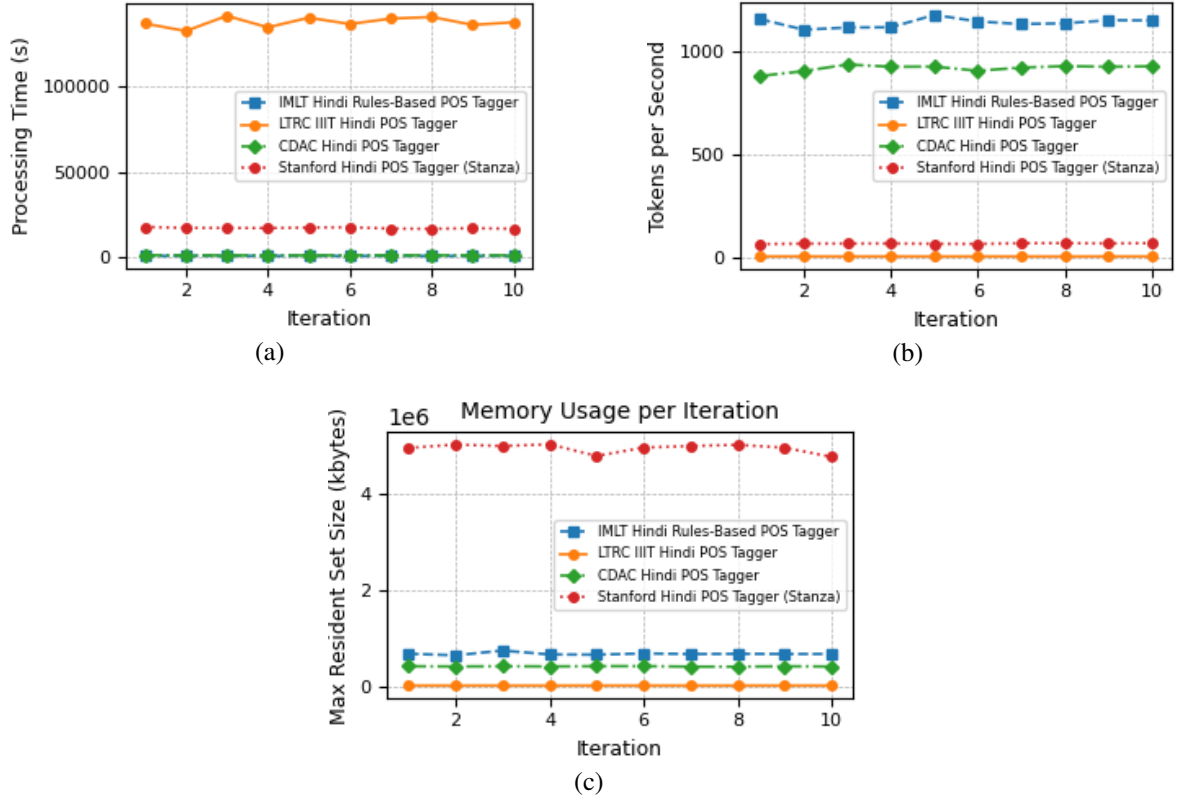


Figure 3. Performance metrics: (a) processing time per iteration, (b) tokens processed per second per iteration, and (c) maximum resident set size for four Hindi PoS taggers

Table 1. Average performance summary with standard deviation for Hindi PoS taggers over 10 iterations

Tagger	Time (s)	Tokens/sec	Max. Res. Memory (kB)
IMLT Hindi rules-based	926.4 \pm 17.6	1,137.4 \pm 21.7	676,614 \pm 26198
LTRC IIIT Hindi	137,590 \pm 2,883	9.12 \pm 0.19	16,276 \pm 41
CDAC Hindi	1273.2 \pm 23.3	917.9 \pm 16.5	415,494 \pm 3874
Stanford Hindi (Stanza)	17,251.9 \pm 367.4	71.6 \pm 1.5	4,943,015 \pm 94,292

Table 2. Average performance summary with standard deviation for Marathi PoS taggers over 10 iterations

Tagger	Time (s)	Tokens/sec	Max. Res. Memory (kB)
LTRC IIIT Marathi	615,433 \pm 2,429.8	1.59 \pm 0.01	247,631.2 \pm 946.2
CDAC Marathi	1951.9 \pm 36.5	436.3 \pm 7.9	431,049.6 \pm 3488.7
Stanford Marathi (Stanza)	38,324.4 \pm 286.3	28.16 \pm 0.21	5,314,620.7 \pm 8,055.5

There is a difference in the number of tokens processed across different PoS-taggers, even though the evaluation is based on the same set of 100 files for Hindi and 100 files for Marathi due to inconsistencies in how the PoS taggers handle tokenization. However, it must be noted that the dataset number of lines and words we are using for evaluating these PoS taggers are the same. Due to variations in tokenization methods, different PoS taggers generated differing token counts for the same dataset. We chose to retain the original tokenization of each system to preserve their design integrity. Drawing on the approach of Chiche and Yitagesu [17], who conducted a comprehensive analysis of PoS tagging systems, comparing various approaches on accuracy and speed, we calculated performance metrics such as accuracy and F1-score based on each tagger's tokenization.

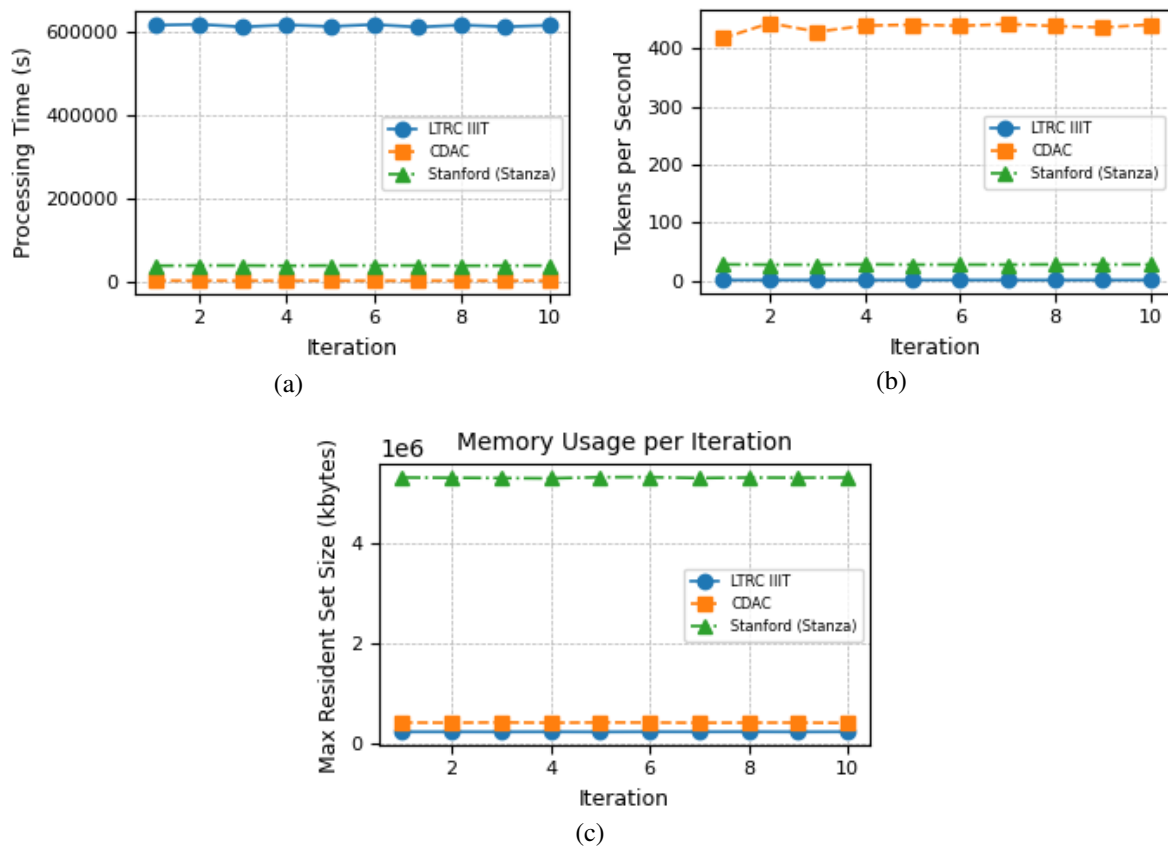


Figure 4. Performance metrics: (a) processing time per iteration, (b) tokens processed per second per iteration, and (c) maximum resident set size for three Marathi PoS taggers

Accuracy evaluation: as illustrated in Figure 5, the block diagram for measuring the accuracy of Hindi PoS taggers compares the PoS-tagged output from the IMLT Hindi rules-based PoS tagger, LTRC IIIT Hindi PoS tagger, and CDAC Hindi PoS tagger with the PoS-tagged production of the Stanford Hindi PoS tagger (Stanza). Hence, Stanford Hindi PoS tagger (Stanza) would act as a benchmark gold standard for evaluating the accuracy of the other three PoS taggers. We chose Stanford Hindi PoS tagger (Stanza) as our gold standard as it is a widely recognized PoS tagger with a coarser tag set than the other three. This would help us propose a meaningful mapping of tags when we compare the other three PoS taggers with Stanford Hindi PoS tagger (Stanza). Proposed mapping: as discussed previously, we are proposing the following tag mapping between different Hindi PoS taggers for an accurate comparison of Hindi PoS taggers.

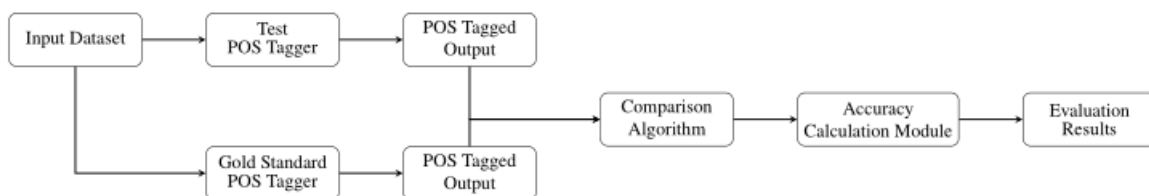


Figure 5. POS tagger evaluation workflow

Table 3 shows the mapping of equivalent linguistic categories from different PoS taggers of the Hindi language to a standardized set. In this case, the Stanford Hindi PoS tagger is the reference or standard tag. Original tags in different PoS taggers are replaced with their equivalent tag from Stanford Hindi PoS tagger for

accuracy measurement. The tags across different taggers use different notations or granular categorizations but often represent the same or similar linguistic categories. For example:

- ADJ (Stanford) represents adjectives. In other taggers, this is mapped to tags like JJ or QO (IMLT, LTRC, and CDAC), which also mark adjectives or related categories like quantifiers or ordinals.
- NOUN (Stanford) represents common nouns. It is mapped to tags like NN and NST in the other taggers, which are also noun-related categories.

The purpose of this mapping is to help in the fair analysis of PoS categories across different taggers by associating tags that describe the same parts of speech; this would help us handle granularity differences while computing accuracy. The first three Hindi PoS taggers that we are comparing with Stanford Hindi PoS tagger use the tag set published by the Bureau of Indian standards, “Linguistic resources — PoS tag set for Indian languages — guidelines for designing tag sets and specification,” [18]. However, each parser uses a different granularity of tagset from this standard. The Stanford uses universal dependencies PoS tags for both Hindi and Marathi languages. The first open-source treebank for Marathi, adhering to the UD syntactic annotation scheme, has been developed to provide a standardized resource for syntactic analysis of the language [19]. We propose a mapping, as shown in Table 4, between different Marathi PoS taggers to enable an accurate comparison with the Stanford Marathi PoS tagger (Stanza).

Table 3. Mapping of different Hindi PoS tagger’s tag set with Stanford Hindi PoS tagger’s tag set

IMLT Hindi	LTRC IIIT Hindi	CDAC Hindi	Stanford Hindi
JJ, QO	JJ, QO, JJC	JJ, QTO	ADJ
PSP	PSP	PSP	ADP
RB, INTF	RB, INTF, RBC	RB, INTF	ADV
VAUX	VAUX	VAUX	AUX
CC	CC, CCC	CCD	CCONJ
DEM, QF, CL	DEM, QF, CL, QFC	DM, DMD, DMR, DMQ, DMI, QT, QTF	DET
INJ	INJ, INJC	INJ	INTJ
NN, NST	NN, NST, NSTC	N, NN, NST	NOUN
QC	QC, QCC	QTC	NUM
RP, NEG	RP, NEG	RP, RPD, NEG	PART
PRP, WQ	PRP, WQ	PR, PRP, PRF, PRL, PRC, PRQ, PRI	PRON
NNP	NNP	NNP	PROPN
SYM	SYM	PUNC	PUNCT
UT	UT	CCS	SCONJ
VM	VM, VMC	V, VM	VERB
RDP, ECH, UNK	RDP, ECH, UNK	RD, RDF, UNK, ECH	X

In the case of LTRC IIIT Marathi PoS tagger, question words (e.g., “who,” “what”) WQ are matched as either PRON (pronouns) or DET (determiners) tags in Stanford Marathi PoS tagger, as shown in Table 4. Similarly, the CC (coordinating conjunction) tag is considered matched as either CCONJ (coordinating conjunction) or SCONJ (subordinating conjunction) tags in Stanford Marathi PoS tagger. The question words (WQ) in LTRC are mapped to either PRON or DET in Stanford based on their context—whether they function as pronouns or determiners. Similarly, the CC tag in LTRC can correspond to either CCONJ or SCONJ in Stanford, depending on its syntactic role. These mappings, like other mappings, reflect differences in granularity between the taggers. In Stanford, the broader LTRC tags are further classified for finer classification, with context guiding the exact match.

Despite the various formats from different PoS taggers, our standardization process simplifies the data into a common format: UTF-8 text files with two columns. The first column contains the token, and the next column contains the corresponding PoS tag of that token, separated by a tab. Each line has a token-PoS tag pair, as illustrated in Figure 6.

After we have the PoS-tagged dataset from each parser, we replace the original PoS tag given by that PoS parser with the corresponding Stanford PoS tag, as per the earlier tables for Hindi and Marathi PoS-tagged data. It would help us measure accuracy by having mapping of tags used by Hindi and Marathi PoS taggers, which are mainly PoS tags set for Indian languages given by the Bureau of Indian standards, with the corresponding Stanford PoS tagger’s tag set, which is primarily UD PoS tag set. We have written some Python scripts to perform this task.

For matching purposes, we match a word in the PoS-tagged file with the corresponding word in the Stanford PoS-tagged file. If we get the same PoS tag, the word is considered to be correctly matched. Since different PoS taggers have different tokenization approaches and hence the different tokens on the same data are generated, to have the best possible matching of words between the test file and the gold standard file, we use a matching algorithm using a Python script.

Table 4. Mapping of different Marathi PoS tagger's tag set with Stanford Marathi PoS tagger's tag set

LTRC IIIT Marathi	CDAC Marathi	Stanford Marathi
JJ, QO	JJ, QTO	ADJ
PSP	PSP	ADP
RB, INTF	RB, INTF	ADV
VAUX	VAUX	AUX
CC	CCD	CCONJ
DEM, QF, WQ	DM, DMD, DMR, DMQ, QT, QTF	DET
INJ	INJ	INTJ
NN, NST	N, NN, NST	NOUN
QC	QTC	NUM
RP, NEG	RP, RPD, NEG	PART
PRP, WQ	PR, PRP, PRF, PRL, PRC, PRQ	PRON
NNP	NNP	PROPN
SYM	PUNC	PUNCT
UT, CC	CCS, UT	SCONJ
VM	V, VM, VNF	VERB
CL, C, RDP, ECH, UNK	RD, RDF, UNK, ECH	X

धूप NOUN

वाला... VERB

भाई, NOUN

Figure 6. Token-PoS tag pair

The Python code compares PoS-tagged data from two files—one containing test data (other PoS taggers) and the other containing a gold standard (Stanford PoS taggers)—by first reading both files and extracting word-PoS pairs. It then uses the `diffib.SequenceMatcher` to find the best matching sequences of words between the two datasets. For each matched word pair, it checks whether the PoS tags are the same, recording true positives (correct matches), false positives (incorrect tags assigned in the test data), and false negatives (correct tags missed by the test data). The code calculates each tag's precision, recall, and F1 scores and produces overall accuracy metrics for different token size thresholds. Pan and Saha [20] evaluated PoS tagging for Bengali text. This work inspired our approach to calculating each tag's precision, recall, and F1 scores and generating overall accuracy metrics for different token size thresholds.

The Python library we use, `diffib.SequenceMatcher` is a Python module that compares sequences of any hashable types, typically for string or token matching. It uses a variant of the Ratcliff/Obershelp algorithm, also known as the “gestalt pattern matching” algorithm [21]. This algorithm compares sequences by finding the longest contiguous matching subsequence and recursively applying the process to the unmatched part. The Ratcliff/Obershelp algorithm is particularly efficient for sequences with large matching blocks, making it suitable for text comparison tasks where most of the text remains unchanged, which is ideal for our datasets. `SequenceMatcher` is easy for quick, approximate matching tasks, with high-level methods to compute similarity ratios and generate diffs. It balances efficiency and accuracy, making it a good choice for textual or sequence-matching tasks where the order and contiguous blocks matter and for cases where the sequences are mainly similar.

The support for each PoS tag is calculated by processing files from a gold-standard folder containing token-tag pairs. A Python script increments a counter for each tag encountered in individual files. These counts are subsequently aggregated across all files using a function that consolidates the tag occurrences into a master dictionary. The final output provides the total occurrences (support) of each PoS tag across all files in the gold-standard folder, effectively representing the frequency distribution of PoS tags in the dataset. The results of this analysis are presented in Table 5.

Table 5. Support table for each tag for gold standard data

POS tag	Support	POS tag	Support
NOUN	178,722	CCONJ	19,356
SCONJ	19,034	PRON	207,328
PART	42,892	VERB	152,316
ADP	100,208	AUX	116,814
PUNCT	231,186	ADJ	52,171
INTJ	1,788	PROPN	52,553
ADV	18,660	DET	24,182
NUM	15,863	X	829

Accuracy analysis is done on PoS taggers, using following different metrics,
Precision for each tag t:

$$\text{Precision}_t = \frac{TP_t}{TP_t + FP_t} \quad (1)$$

Recall for each tag t:

$$\text{Recall}_t = \frac{TP_t}{TP_t + FN_t} \quad (2)$$

F1 score for each tag t:

$$F1_t = 2 \times \frac{\text{Precision}_t \times \text{Recall}_t}{\text{Precision}_t + \text{Recall}_t} \quad (3)$$

Table 6 demonstrates IMLT Hindi rule-based PoS tagger's effectiveness across various parts of speech, with exceptionally high precision for categories such as PRON (0.98) and PUNCT (0.99). At the same time, challenges remain with tags like SCONJ and INTJ. Table 7 shows the LTRC IIIT Hindi PoS tagger performs exceptionally well for categories like PRON (precision: 0.98, recall: 0.94, F1: 0.96) and PUNCT (precision: 1.00, F1: 0.99), while challenges persist for tags like SCONJ and INTJ, with lower performance metrics due to fewer correct identifications.

Table 8 shows CDAC Hindi PoS tagger demonstrates high precision for categories such as PUNCT (1.00) and NUM (0.82) while maintaining consistent performance across most tags. However, challenges are noted for tags like SCONJ and INTJ, which show lower F1 scores due to lower precision and recall values. Table 9 shows LTRC IIIT Marathi PoS tagger performs exceptionally well in recognizing tags like PRON (F1: 0.85), PUNCT (F1: 0.98) and CCONJ (F1: 0.91). However, it struggles with tags like PROPN and INTJ, which have significantly lower F1 scores due to lower precision and recall. Table 10 shows CDAC Marathi PoS tagger performs well in categories like PRON (precision: 0.82, F1: 0.73), NOUN (F1: 0.73), and CCONJ (F1: 0.82), while lower performance is observed for tags such as PROPN and INTJ due to limited precision and recall.

Table 6. Performance metrics for the IMLT Hindi rule-based PoS tagger

PoS tag	TP	FN	FP	Precision	Recall	F1 score
NOUN	140,263	13,504	42,924	0.77	0.91	0.83
CCONJ	15,092	1,412	16,714	0.47	0.91	0.62
SCONJ	0	16,104	0	0.00	0.00	0.00
PRON	146,578	35,553	2,749	0.98	0.80	0.88
PART	31,039	5,557	1,430	0.96	0.85	0.90
VERB	111,263	17,017	34,694	0.76	0.87	0.81
ADP	77,684	11,407	2,521	0.97	0.87	0.92
AUX	63,674	29,576	12,699	0.83	0.68	0.75
PUNCT	139,920	4,466	1,221	0.99	0.97	0.98
ADJ	37,138	8,856	9,321	0.80	0.81	0.80
INTJ	279	973	1,233	0.18	0.22	0.20
PROPN	23,184	17,463	13,444	0.63	0.57	0.60
ADV	8,928	6,909	2,276	0.80	0.56	0.66
DET	16,766	4,431	11,671	0.59	0.79	0.68
NUM	11,246	878	664	0.94	0.93	0.94
X	0	744	0	0.00	0.00	0.00

Table 7. Performance metrics for the LTRC IIIT Hindi PoS tagger

PoS tag	TP	FN	FP	Precision	Recall	F1 score
NOUN	160,750	14,476	43,530	0.79	0.92	0.85
CCONJ	17,464	477	15,064	0.54	0.97	0.69
SCONJ	0	18,186	0	0.00	0.00	0.00
PRON	191,931	11,232	4,574	0.98	0.94	0.96
PART	40,451	1,797	4,945	0.89	0.96	0.92
VERB	132,680	17,114	43,109	0.75	0.89	0.82
ADP	89,316	10,141	2,312	0.97	0.90	0.93
AUX	76,688	37,607	13,379	0.85	0.67	0.75
PUNCT	114,241	1,932	65	1.00	0.98	0.99
ADJ	42,024	9,412	3,118	0.93	0.82	0.87
INTJ	759	649	3,365	0.18	0.54	0.27
PROPN	20,443	28,543	6,058	0.77	0.42	0.54
ADV	10,461	7,582	1,683	0.86	0.58	0.69
DET	18,625	5,239	5,942	0.76	0.78	0.77
NUM	13,157	369	891	0.94	0.97	0.95
X	191	632	285	0.40	0.23	0.29

Table 8. Performance metrics for the CDAC Hindi PoS tagger

PoS tag	TP	FN	FP	Precision	Recall	F1 score
NOUN	160,932	10,142	107,909	0.60	0.94	0.73
CCONJ	13,849	4,584	1,210	0.92	0.75	0.83
SCONJ	11,727	6,080	5,448	0.68	0.66	0.67
PRON	135,724	65,671	3,786	0.97	0.67	0.80
PART	33,150	7,610	4,451	0.88	0.81	0.85
VERB	111,866	30,719	22,543	0.83	0.78	0.81
ADP	85,738	12,723	3,398	0.96	0.87	0.91
AUX	86,834	16,469	17,110	0.84	0.84	0.84
PUNCT	115,519	6,641	0	1.00	0.95	0.97
ADJ	21,615	29,302	4,491	0.83	0.42	0.56
INTJ	248	1,138	464	0.35	0.18	0.24
PROPN	23,798	20,827	14,067	0.63	0.53	0.58
ADV	4,114	13,473	1,412	0.74	0.23	0.36
DET	16,269	7,393	37,955	0.30	0.69	0.42
NUM	14,575	799	3,133	0.82	0.95	0.88
X	0	824	118	0.00	0.00	0.00

Table 9. Performance metrics for the LTRC IIIT Marathi PoS tagger

PoS tag	TP	FN	FP	Precision	Recall	F1 score
PRON	107,302	7,260	30,350	0.78	0.94	0.85
NOUN	129,133	35,502	66,476	0.66	0.78	0.72
NUM	2,332	1,173	7,170	0.25	0.67	0.36
ADJ	20,062	23,508	20,758	0.49	0.46	0.48
VERB	111,579	40,528	56,540	0.66	0.73	0.70
ADV	7,023	69,034	2,028	0.78	0.09	0.17
AUX	40,813	42,486	10,202	0.80	0.49	0.61
PUNCT	145,874	4,182	832	0.99	0.97	0.98
DET	10,460	14,661	18,947	0.36	0.42	0.38
CCONJ	18,182	3,611	0	1.00	0.83	0.91
PROPN	1,890	3,764	24,695	0.07	0.33	0.12
SCONJ	7,933	5,160	757	0.91	0.61	0.73
PART	52	922	2,190	0.02	0.05	0.03
INTJ	162	15,628	2,383	0.06	0.01	0.02
ADP	150	3,862	152	0.50	0.04	0.07
X	0	12	18	0.00	0.00	0.00

Table 10. Performance metrics for the CDAC Marathi PoS tagger

PoS tag	TP	FN	FP	Precision	Recall	F1 score
PRON	70,651	37,414	15,172	0.82	0.65	0.73
NOUN	135,458	14,330	86,990	0.61	0.90	0.73
NUM	2,509	1,345	6,652	0.27	0.65	0.39
ADJ	17,925	23,835	14,198	0.56	0.43	0.49
VERB	90,965	24,069	32,378	0.74	0.79	0.76
ADV	11,168	55,307	5,965	0.65	0.17	0.27
DET	12,505	12,424	39,543	0.24	0.50	0.32
PROPN	955	3,187	10,906	0.08	0.23	0.12
CCONJ	16,354	5,176	1,872	0.90	0.76	0.82
AUX	22,705	24,995	8,686	0.72	0.48	0.57
SCONJ	7,039	5,062	3,647	0.66	0.58	0.62
INTJ	11	8,853	0	1.00	0.00	0.00
PUNCT	8,981	6,430	42	1.00	0.58	0.74
ADP	0	3,808	4	0.00	0.00	0.00
PART	11	935	851	0.01	0.01	0.01
X	0	11	1	0.00	0.00	0.00

Robustness: we further evaluate the robustness of various PoS taggers (IMLT Hindi rules-based, LTRC IIIT Hindi, CDAC Hindi, LTRC IIIT Marathi, and CDAC Marathi) by checking accuracy across different token sizes (10K, 50K, 100K, 250K, 500K, 1M, and the entire dataset). The evaluation focuses on key metrics such as macro precision, macro recall, macro F1 score, and overall accuracy. In this study, we utilize macro-averaging to assess the performance of each PoS tagger. Macro-averaging involves calculating each tag's precision, recall, and F1 score independently, followed by averaging these metrics across all tags. This approach mitigates the bias introduced by tags that occur more frequently, ensuring a balanced evaluation of the model's performance across all tags. Once the precision, recall, and F1 score are calculated for each tag, the macro-averaged values are computed by taking the mean across all tags:

Macro-averaged precision:

$$\text{Macro Precision} = \frac{1}{n} \sum_{t=1}^n \text{Precision}_t \quad (4)$$

Macro-averaged recall:

$$\text{Macro Recall} = \frac{1}{n} \sum_{t=1}^n \text{Recall}_t \quad (5)$$

Macro-averaged F1 score:

$$\text{Macro F1 Score} = \frac{1}{n} \sum_{t=1}^n \text{F1 Score}_t \quad (6)$$

where n is the total number of tags in tag set.

The overall accuracy of the model is computed as the ratio of the total number of correct predictions to the total number of words:

$$\text{Overall Accuracy} = \frac{\text{Total Correct}}{\text{Total Tokens}} \quad (7)$$

Table 11 presents the performance metrics, including macro precision, macro recall, macro F1 score, and overall accuracy, at varying token sizes ranging from 10K to 1M, for IMLT Hindi rules-based PoS tagger. The tagger demonstrates a consistent performance across all token sizes, with slight improvement observed in the 50K token size category. Table 12 shows the performance metrics evaluated over varying token sizes, including macro precision, macro recall, macro F1 score, and overall accuracy, for for LTRC IIIT Hindi PoS tagger. The results demonstrate consistent overall accuracy across all token sizes, with a slight variation in macro F1 score at higher token counts.

Table 11. Token-wise performance metrics for IMLT Hindi rules-based PoS tagger

Token size	Macro precision	Macro recall	Macro F1 score	Overall accuracy
10K	0.63	0.63	0.62	0.82
50K	0.64	0.64	0.63	0.83
100K	0.63	0.63	0.62	0.82
250K	0.63	0.63	0.62	0.82
500K	0.63	0.63	0.62	0.82
1M	0.63	0.63	0.62	0.82
All	0.63	0.63	0.62	0.82

Table 12. Token-wise performance metrics for LTRC IIIT Hindi PoS tagger

Token size	Macro precision	Macro recall	Macro F1 score	Overall accuracy
10K	0.63	0.64	0.62	0.84
50K	0.61	0.61	0.60	0.84
100K	0.61	0.61	0.59	0.84
250K	0.61	0.61	0.59	0.85
500K	0.61	0.61	0.59	0.85
1M	0.61	0.61	0.59	0.85
All	0.61	0.61	0.60	0.85

Table 13 presents macro precision, macro recall, macro F1 score, and overall accuracy measured over various token sizes, for CDAC Hindi PoS tagger. The tagger consistently achieves a 0.78 overall accuracy across all token sizes, with minimal variation in precision and F1 scores, highlighting stable performance irrespective of the token count. Table 14 shows macro precision, macro recall, macro F1 score, and overall accuracy for different token sizes for LTRC IIIT Marathi PoS tagger. The performance remains relatively consistent across all token sizes, with an overall accuracy stabilizing at 0.69 for larger token sets, reflecting modest precision and recall values. Table 15 presents the macro precision, macro recall, macro F1 score, and overall accuracy across varying token sizes for CDAC Marathi PoS tagger. The metrics demonstrate consistency as the token size increases, with an overall accuracy of 0.64 for larger token sets and a slight variation in precision and recall.

Macro-averaging is especially beneficial when dealing with imbalanced datasets where certain tags appear much more frequently than others. By averaging metrics across all tags, this method ensures that the evaluation is not biased toward the most frequent tags. It provides a more equitable measure of the model's performance across all parts of speech, including underrepresented tags. This is particularly important in applications where less common tags are of significant interest and must be evaluated with the same rigor as more frequent ones.

Table 13. Token-wise performance metrics for CDAC Hindi PoS tagger

Token size	<i>M</i> Precision	<i>M</i> Recall	<i>M</i> F1 score	Overall accuracy
10K	0.63	0.58	0.59	0.78
50K	0.64	0.57	0.58	0.78
100K	0.63	0.57	0.58	0.78
250K	0.63	0.57	0.58	0.78
500K	0.63	0.57	0.58	0.78
1M	0.63	0.57	0.58	0.78
All	0.63	0.57	0.58	0.78

Table 14. Token-wise performance metrics for LTRC IIIT Marathi PoS tagger

Token size	<i>M</i> Precision	<i>M</i> Recall	<i>M</i> F1 score	Overall accuracy
10K	0.40	0.37	0.36	0.66
50K	0.44	0.38	0.37	0.67
100K	0.44	0.39	0.37	0.68
250K	0.44	0.39	0.37	0.68
500K	0.42	0.37	0.36	0.69
1M	0.42	0.37	0.36	0.69
All	0.42	0.37	0.36	0.69

Table 15. Token-wise performance metrics for CDAC Marathi PoS tagger

Token size	M Precision	M Recall	M F1 score	Overall accuracy
10K	0.42	0.38	0.37	0.62
50K	0.46	0.36	0.36	0.62
100K	0.46	0.37	0.36	0.63
250K	0.43	0.35	0.34	0.63
500K	0.41	0.34	0.33	0.63
1M	0.41	0.34	0.33	0.64
All	0.41	0.34	0.33	0.64

Figure 7 presents the performance metrics (macro precision, macro recall, macro F1 score, and overall accuracy) for three Hindi PoS taggers: IMLT Hindi rules-based tagger, LTRC IIIT Hindi tagger, and CDAC Hindi tagger. Figure 8 presents the performance metrics (macro precision, macro recall, macro F1 score, and overall accuracy) for two Marathi PoS taggers: LTRC IIIT Marathi tagger and CDAC Marathi tagger. This comparison is crucial in understanding the performance variation in these PoS taggers, which could be attributed to the underlying algorithms (HMM, conditional random fields) used for PoS tagging, along with the way they handle steps before PoS tagging, like managing morphological complexity.

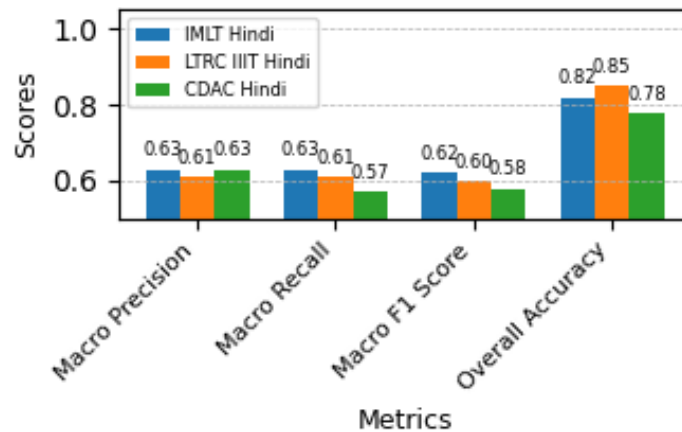


Figure 7. Performance comparison of Hindi PoS taggers

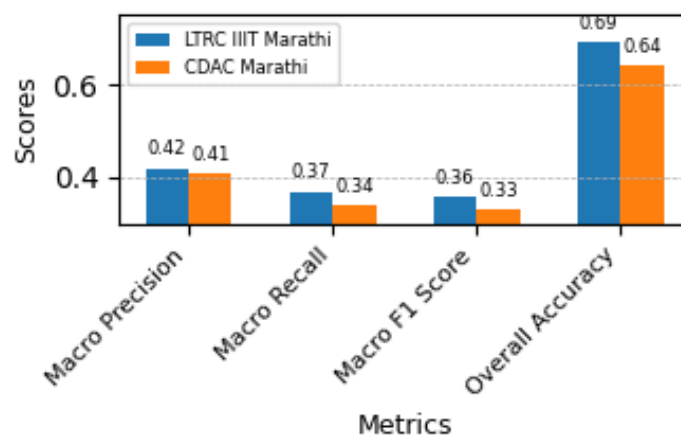


Figure 8. Performance comparison of Marathi PoS taggers

5. DISCUSSION AND FUTURE WORK

Executing and comparing the PoS taggers, especially in a Linux environment, was a challenging task. The lack of regular software updates from many of these taggers led to issues such as missing or incompatible libraries, which were difficult to resolve. Furthermore, these libraries had compatibility issues when switching from 32-bit to 64-bit modes, adding another layer of complexity to the process.

For Hindi PoS taggers, the IMLT rules-based PoS tagger processed 1,053,370 tokens with an average processing time of 926.4 seconds, with an average token processing rate of 1,137.43 tokens/second, and used approximately 676,614.4 kB of memory (MSS). Comparatively, the LTRC IIIT PoS tagger processed 1,254,066 tokens at a slower rate of 9.12 tokens/second, with a significantly lower memory footprint of 16,276.4 kB (MSS). Similarly, the CDAC Hindi PoS tagger performed faster than LTRC, processing 917.92 tokens/second with moderate memory usage. The Stanford Hindi PoS tagger (Stanza) had the highest memory surge of around 4,943,014.8 kB (MSS) but processed fewer tokens per second (71.55).

In the case of Marathi PoS taggers, the LTRC IIIT PoS tagger processed the least tokens per second (1.59) with an average memory usage of 247,631.2 kB (MSS). In contrast, the CDAC Marathi tagger showed a better balance, processing 436.3 tokens per second with moderate memory consumption. The Stanford Marathi tagger (Stanza) exhibited high memory usage of 5,314,620.7 kB (MSS) but processed only 28.16 tokens per second. While rule-based taggers like IMLT show efficiency in token processing, modern ML models (like Stanza) generally consume more memory, potentially signifying more complex processing capabilities, though at the cost of speed compared to traditional rule-based methods. These results highlight the trade-off between memory usage and processing speed in different tagging systems and suggest the need for further optimization based on specific resource constraints and performance needs. Baishya and Baruah [22] proposed highly efficient parts of speech tagging in low-resource languages, focusing on memory consumption and efficiency trade-offs, which provokes us further to evaluate the trade-off between memory usage and processing speed. Our results highlight the need for optimization based on resource constraints and performance needs.

Robustness was assessed regarding each tagger's performance consistency across altered token sizes, as shown in tables. The IMLT Hindi rules-based PoS tagger showed stable results across all token sizes, with only minimal accuracy and F1 score fluctuations. Although LTRC IIIT Hindi PoS tagger has higher accuracy (85%), it exhibited a slight drop in robustness when processing large datasets, as indicated by its marginal decrease in precision and recall for larger token sizes. On the Marathi side, both LTRC IIIT Marathi and CDAC Marathi PoS taggers maintained similar performance across different token sizes, with LTRC slightly outperforming CDAC in terms of overall robustness, particularly in handling tags with less frequency. Close to our method to check robustness using token size, Kumar *et al.* [23] explored tokenization in low-resource language settings and proposed novel techniques for improving tagger performance in bhojpuri. However, their work primarily focuses on other Indian languages like Angika, Magahi, Bhojpuri, and Hindi.

The accuracy results from comparing various PoS taggers emphasize differences in performance across Hindi and Marathi language PoS taggers with their datasets. The key metrics evaluated include macro precision, macro recall, macro F1 score, and overall accuracy across varying token sizes. Each PoS tagger demonstrates a degree of consistency in its performance with respect to differences that emerge, particularly in the handling of specific tags, such as PROP (proper nouns), INTJ (interjections), and SCONJ (subordinating conjunctions). These differences are most pronounced in the context of Hindi and Marathi languages, where linguistic structures can vary significantly.

The IMLT Hindi rules-based PoS tagger consistently showed steady results with an overall accuracy of 82%. However, this tagger struggled to identify less frequent parts of speech, such as interjections and subordinating conjunctions. Its performance plateaued as token size increased, which might indicate limited scalability. On the other hand, the LTRC IIIT Hindi PoS tagger outperformed other taggers in terms of overall accuracy, achieving 85%. Its high precision and recall for categories such as pronouns and punctuation indicate its robust handling of these tags. However, challenges remained in identifying subordinating conjunctions and interjections, a pattern seen as consistent across most taggers. The CDAC Hindi PoS tagger exhibited a slightly lower overall accuracy of 78%, with better performance on more frequent tags like nouns and punctuation. However, its F1 scores for less frequent tags, such as subordinating conjunctions and interjections, were lower, indicating room for improvement in handling these categories of PoS tags.

For the Marathi language, the LTRC IIIT Marathi PoS tagger demonstrated consistent accuracy (69%), but its F1 scores across most tags remained lower, particularly for proper nouns and interjections. The tagger performed well on high-frequency tags but struggled with tags with a lesser frequency in the dataset. The

CDAC Marathi PoS tagger, with an overall accuracy of 64%, showed consistency in results across various token sizes. Its precision and recall values for specific tags, such as determiners and pronouns, were lower, which affected the overall performance of this tagger. The LTRC IIIT Hindi PoS tagger, which achieved the highest accuracy (85%), demonstrated an effective balance between speed and accuracy. However, there was a trade-off, as it required more computational resources. Similarly, the CDAC Hindi PoS tagger was slightly less accurate (78%). Still, it proved to be faster and required less memory, making it a potentially better option for real-time PoS tagging applications.

Future work: future research should explore hybrid models combining the strengths of rule-based and statistical approaches to enhance the speed-accuracy trade-off. Additionally, extending Paul *et al.* [24] case study on pre-trained language models in Indian legal domains to Hindi and Marathi movie scripts presents promising directions. Hybrid methodologies could also improve accuracy for infrequent PoS tags, while incorporating domain-specific training may enhance robustness, especially for underrepresented tags. Further research opportunities include optimizing systems for low-resource scenarios through efficient algorithms that minimize memory usage and processing time without sacrificing accuracy. Moreover, extending the analysis to mixed-language datasets—such as Hindi-Marathi or English-Marathi—and pre-trained models for code-mixed Hindi-English text [25] could yield insightful results. Finally, evaluating these PoS taggers across diverse real-world applications like conversational artificial intelligence (AI) and social media text would provide valuable insights into their scalability and flexibility.

6. CONCLUSION

This study evaluates five PoS taggers across Hindi and Marathi using the movies dataset. Results show strong performance on frequent tags like nouns, verbs, and punctuation, but challenges remain with less common tags such as subordinating conjunctions (e.g., ‘because’) and interjections (e.g., ‘wow’). Accuracy plateaus as token size increases, highlighting scalability issues. For Hindi, the LTRC IIIT Hindi PoS tagger is the most robust, excelling in frequent tag handling but at a higher computational cost. The LTRC IIIT Marathi PoS tagger also performs well, but there is an urgent need for improvements in rare tag recognition. The CDAC PoS Taggers strike a balance between accuracy and efficiency, making them suitable for low-resource environments. This work lays the groundwork for improving PoS tagging tools for underrepresented Indian languages and other low-resource languages. While PoS taggers perform well on common parts of speech, gaps remain in handling less frequent tags like interjections and subordinating conjunctions.

FUNDING INFORMATION

This research received no external funding.

AUTHOR CONTRIBUTIONS STATEMENT

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Pratik Narayanrao Kalamkar	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Prasadu Peddi	✓									✓		✓		
Yogesh Kumar Sharma		✓				✓				✓		✓	✓	

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal Analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project Administration

Fu : Funding Acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.




DATA AVAILABILITY

- The data that support the findings of this study will be available in <https://github.com/pratik1986/Pos-Comparison.git>, <https://dx.doi.org/10.21227/2sb3-2643>, following 1 month embargo from the date of publication to allow for the commercialization of research findings.




REFERENCES

- [1] K. Chiplunkar, M. Kharche, T. Chaudhari, S. Shaligram, and S. Limkar, "Prediction of POS tagging for unknown words for specific Hindi and Marathi language," in *Intelligent Data Engineering and Analytics. Advances in Intelligent Systems and Computing*, vol. 1177. Springer, Singapore., 2020, pp. 133–143, doi: 10.1007/978-981-15-5679-1_13.
- [2] P. B. Pimpale and R. N. Patel, "Experiments with POS tagging code-mixed Indian social media text," *arXiv preprint arXiv:1610.09799*, 2016, doi: 10.48550/arXiv.1610.09799.
- [3] S.-B. Park, H.-N. Kim, H. Kim, and G.-S. Jo, "Exploiting script-subtitles alignment to scene boundary detection in movie," in *2010 IEEE International Symposium on Multimedia*, 2010, pp. 49–56, doi: 10.1109/ism.2010.17.
- [4] P. Lahoti, N. Mittal, and G. Singh, "A survey on NLP resources, tools, and techniques for Marathi language processing," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 22, pp. 1–34, 2022, doi: 10.1145/3548457.
- [5] D. Kumawat and V. Jain, "POS tagging approaches: a comparison," *International Journal of Computer Applications*, vol. 118, no. 6, 2015, doi: 10.5120/20752-3148.
- [6] T. Horsmann, N. Erbs, and T. Zesch, *Fast or accurate?—A comparative evaluation of PoS tagging models*. Universität Duisburg-Essen, 2015.
- [7] M. Jacobsen, M. H. Sørensen, and L. Derczynski, "Optimal Size-performance tradeoffs: weighing PoS tagger models," *arXiv preprint arXiv:2104.07951*, 2021, doi: 10.48550/arXiv.2104.07951.
- [8] K. Talukdar and S. K. Sarma, "Parts of speech taggers for Indo Aryan languages: a critical review of approaches and performances," in *2023 4th International Conference on Computing and Communication Systems (I3CS)*, 2023, pp. 1–6, doi: 10.1109/I3CS58314.2023.10127336.
- [9] S. Warjri, P. Pakray, S. A. Lyngdoh, and A. K. Maji, "Part-of-speech (PoS) tagging using conditional random field (CRF) model for Khasi corpora," *International Journal of Speech Technology*, vol. 24, no. 4, pp. 853–864, 2021, doi: 10.1007/s10772-021-09860-w.
- [10] I. Tambat and M. Narayankar, "A comparative survey on parts of speech taggers for the Marathi language," in *2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2022, pp. 1740–1747, doi: 10.1109/ICIRCA54612.2022.9985567.
- [11] P. Sarda, D. Kokare, R. Mokashi, T. Patkar, P. Deshpande, and S. Jahirabadkar, "'Shabda Anveshak': comparative review and analysis of Marathi part of speech tagging approaches," in *2025 1st International Conference on AIML-Applications for Engineering & Technology (ICAET)*, 2025, pp. 1–7, doi: 10.1109/ICAET63349.2025.10932205.
- [12] K. Talukdar, S. K. Sarma, and M. P. Bhuyan, "Parts of speech (PoS) and Universal parts of speech (UPoS) tagging: a critical review with special reference to low resource languages," in *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*, Dec. 2023, pp. 703–713, [Online]. Available: <https://aclanthology.org/2023.icon-1.70/>.
- [13] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, 2003, pp. 252–259, doi: 10.3115/1073445.1073478.
- [14] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Association for Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993, doi: 10.21236/ada273556.
- [15] R. K. Mundotiya, V. Kumar, A. Mehta, and A. K. Singh, "Attention-based domain adaption using transfer learning for part-of-speech tagging: an experiment on the Hindi language," 2020, [Online]. Available: <https://aclanthology.org/2020.paclic-1.54/>.
- [16] C. D. Manning, P. Qi, and Y. Zhang, "Stanza: a Python natural language processing toolkit for many human languages," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020, pp. 101–108, doi: 10.18653/v1/2020.acl-demos.14.
- [17] A. Chiche and B. Yitagesu, "Part of speech tagging: a systematic review of deep learning and machine learning approaches," *Journal of Big Data*, vol. 9, no. 1, p. 10, 2022, doi: 10.1186/s40537-022-00561-y.
- [18] Bureau of Indian Standards, "Linguistic resources - POS tag set for Indian languages - guidelines for designing tagsets and specification," 2021, [Online]. Available: <https://archive.org/details/gov.in.is.17627.2021>.
- [19] V. Ravishankar, "A universal dependencies Treebank for Marathi," in *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, 2017, pp. 190–200, [Online]. Available: <https://aclanthology.org/W17-7623/>.
- [20] S. Pan and D. Saha, "Performance evaluation of part-of-speech tagging for Bengali text," *Journal of The Institution of Engineers (India): Series B*, vol. 103, no. 2, pp. 577–589, 2022, doi: 10.1007/s40031-021-00630-5.
- [21] Python Software Foundation, "difflib — helpers for computing deltas." 2024, [Online]. Available: <https://docs.python.org/3/library/difflib.html>.
- [22] D. Baishya and R. Baruah, "Highly efficient parts of speech tagging in low resource languages with improved hidden Markov model and deep learning," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 10, 2021, doi: 10.14569/ijacsa.2021.0121011.
- [23] S. Kumar, P. Jyothi, and P. Bhattacharyya, "Part-of-speech tagging for extremely low-resource Indian languages," in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 14422–14431, doi: 10.18653/v1/2024.findings-acl.857.
- [24] S. Paul, A. Mandal, P. Goyal, and S. Ghosh, "Pre-trained language models for the legal domain: a case study on Indian law," in *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, 2023, pp. 187–196, doi: 10.1145/3594536.3595165.
- [25] R. Nayak and R. Joshi, "L3Cube-HingCorpus and HingBERT: a code mixed Hindi-English dataset and BERT language models," *arXiv preprint arXiv:2204.08398*, 2022, doi: 10.48550/arXiv.2204.08398.




BIOGRAPHIES OF AUTHORS

Pratik Narayanrao Kalamkar    received his first bachelor of engineering degree in computer science from the prestigious Savitribai Phule Pune University, India, in the year 2011. He also has a Master of Engineering (computer science) degree from Savitribai Phule Pune, India, in 2014. He is currently pursuing his research in the field of ML and NLP as a Ph.D. scholar from Shri Jagdishprasad Jhabarmal Tibrewala University, Jhunjhunu, India. He can be contacted at email: pratik2kcn@gmail.com.



Prasadu Peddi    currently serves as a research supervisor at Shri Jagdishprasad Jhabarmal Tibrewala University, Jhunjhunu, India. He has successfully guided 13 research scholars to the completion of their doctoral degrees. His scholarly contributions include the publication of 93 research papers in reputed international journals and the presentation of research at 16 national and international conferences. He can be contacted at email: peddiprasad37@gmail.com.



Yogesh Kumar Sharma    is an associate professor at Shri Jagdishprasad Jhabarmal Tibrewala University, Jhunjhunu, India and heads the Department of Computer Science and Engineering. He is also working as a research coordinator. Has over 20 years of experience in research publications. He can be contacted at email: dr.sharmayogeshkumar@gmail.com.