On State-of-the-art of POS Tagger, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder for Indo-Aryan and Dravidian Languages

Hema Gaikwad[1], Jatinderkumar R. Saini[2]
Symbiosis Institute of Computer Studies and Research
Symbiosis International (Deemed University)
Pune, India

***Abstract*—Computational Linguistic refers to the development of the computer systems that deal with human languages. In this paper, different Computational Linguistic Techniques such as Parts of Speech (POS) tagger, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder were considered. After a thorough literature review, it was found that fifteen techniques were used for POS tagging, nine techniques were used for "Sandhi" splitting, one work is done for "Alankaar" finder and absolutely no techniques are available for "Samaas" finder for the Indo-Aryan as well as Dravidian languages. Analysis shows that Rule Based Approach (RBA) and Hidden Markov Model (HMM) are frequently used for POS tagging, RBA is most frequently used for "Sandhi" Splitter, the general Human Intelligence (HI) is used for "Alankaar" Finder and no "Samaas" finder technique is available for any Indian language.**

***Keywords*—"Alankaar"; "Samaas"; "Sandhi"; Parts of Speech tagger (POST)**

I. Introduction

Natural Language Processing (NLP) has two main branches comprising of Natural Language Understanding (NLU) and Natural Language Generation (NLG). Computational Linguistic is a part of NLP and it requires a good understanding of both programming as well as knowledge of the language. Computational Linguistic techniques include Machine Translation, Speech Recognition systems, Text-to-Speech Synthesizers, Interactive Voice Response systems, Search Engines, POST, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder.

Many languages are spoken in different parts of India. The Indian languages can be divided mainly into Indo-Aryan and Dravidian languages. Punjabi, Hindi, Gujarati, Marathi, etc. are the examples of Indo-Aryan languages while Malayalam, Telugu, Kannada, etc. are the examples of the Dravidian languages. Hindi, recognized as the official language of India, is one of the most common languages in India [1]. It alone has 38 million native speakers and happens to be the fourth most spoken language of the world [2]. Hindi also has various dialects. For instance, Awadhi which is one of its dialects is spoken in 20 districts of India and 08 districts of Nepal [3]. The prominent texts like "Ramcharitmanas", "Hanuman Chalisa" and "Padmavat" are written in Awadhi [4][5][6].

This paper presents a very thorough and exhaustive study of the various types of tools for the various Indian languages. The tools covered in this paper include POS tagger, "Sandhi" Splitter, "Alankaar" finder and "Samaas" finder. The best attempt has been made to present the research works done in the area till date.

The research work is segregated into various sections: Section II describes related work. Section III discusses the Analysis of NLP techniques for Indian Languages. Finally, the Section IV describes the Conclusion and Future work.

II. Related Work

Basit et al. [7] talked about Awadhi POS tagger and its tag set. For developing tag set authors referred Bureau of Indian standards (BIS) and used Feature Based Approaches (FBA). Various features like word level, tag level, character level and Boolean level are used for POS tagging. Ekbal et al. [8] developed Bengali POS tagger using Maximum Entropy Approach (MEA). They worked on 72,341 words and uses 26 tags. MEA is based on feature selection and it can be lexicon feature, name entity recognition, suffix and prefix of word, context free feature, digit feature etc. By using the above features, the system got 88.2% accuracy. Proisl et al. [9] experimented parts of speech tagging on Magahi and Bhojpuri by using SoMeWeTa, Bi-Long Short Term Memory (LSTM)+Conditional Random Field (CRF) and Standard tagger approach. SoMeWeTa tagger depends on average structure perceptron. Bi-LSTM uses character word embedding and support transfer learning. Standard tagger based on Maximum Entropy Cyclic Dependency Network (MECDN). After experimenting, authors achieved 90.70% for Magahi and 94.08% for Bhojpuri. Ojha et al. [10] used CRF and Support Vector Machine (SVM) for tagging the Indo Aryan Languages Specifically Hindi, Odia and Bhojpuri. 90K tokens were used for training the system and 2K tokens were used for testing purpose. 88% to 93.7% accuracy was achieved with SVM and 82% to 86.7% accuracy was achieved with CRF.

Singh et al. [11] presented Bhojpuri POS tagger developed by SVM with 87.3% to 88.6% accuracy and errors can be minimized by increasing the corpus size. Pandey et al. [12] developed Chhattisgarhi POS tagger using RBA. 40,000 words (taken from story books) and 30 tags were used for testing purpose and achieved 78% accuracy. Sinha et al. [13]

presented Chhattisgarhi language rules so that this could further be used for developing parser and translators for the Chhattisgarhi language. Reddy et al. [14] developed cross language POS tagger using HMM i.e. Kannada POS tagger using Telugu resources. Bhirud et al. [15] talked about the significance of various Computational Linguistics (CL) tools such as Grammar checker, POST, "Sandhi" Splitter and "Samaas" Finder.

Verma et al. [16] talked about the Lexical analysis or tokenization process. The authors used different religious text such as Bible, Gita, Guru Granth Sahib, Rigveda and Quran to perform the lexical analysis process. Bhatt et al. [17] checked the accuracy of Gujrati POS tagger. For this, the author worked on two different data sets and two different methods. The data sets were Sports information dataset and Amusement dataset. By using HMM 70% and 56% accuracy were gained for sports information dataset and Amusement dataset respectively. By using RBA model, authors got 76% and 80% accuracy for sports information dataset and amusement dataset respectively. Sharma et al. [18] stated that multiple techniques were used to perform POS tagging on Hindi text. The techniques either based on Rules or based on Statistics or based on both. The statistical model could be SVM, HMM, CRF and MEA.

Narayan et al. [19] developed Hindi POS tagger using Artificial Neural Network (ANN) and achieved 91.03% accuracy. Narayan et al. [20] developed Hindi POS tagger using Quantum Neural Network (QNN) and achieved 99.13% accuracy. Mohnot et al. [21] proposed Hindi POS tagger developed using Hybrid Approach (HA) and it could be the combination of RBA, CRF, HMM and so on. 80,000 words and seven types of tags were used for experiment purpose. Joshi et al. [22] stated that three approaches were very common for POS tagging, they are RBA, Statistical Approach (SA) and HA. Garg et al. [23] used RBA for Hindi POS tagger. In this paper authors referred news, essay and short stories and collected 26,149 words and used 30 different tags and achieved 87.55% accuracy. Shrivastava et al. [24] developed Hindi POS tagger using Longest Suffix Matching Approach of HMM and got 93.12% accuracy. Dalal et al. [25] stated that Maximum Entropy Markov Model (MEMM) is used for POS tagging and chunking. This model is having various features such as corpus based feature, word based feature, dictionary based feature and context based features. The first three features are used for POS tagging and last feature is used for chunking purpose.

Antony et al. [26] developed Kannada POS tagger using SVM. Authors himself developed his own corpus, and words are taken from Kannada newspaper and books. Initially the corpus size was 1000 words then 25,000 words and finally 54,000 words and 30 tags. Accordingly, authors gained 48%, 66% and 86% accuracy respectively. Priyadarshi et al. [27] proposed Maithili POS using CRF. Author himself annotated Maithili text and created a corpus which consisted of 52,190 words. 85.88% accuracy was achieved when experiment was performed on wikipedia dumps and other Maithili web resources. Mundotiya et al. [28] developed Maithili POS tagger using CRF and achieved 0.77% precision & recall, 0.78% F1 score and 0.77% accuracy. Jha et al. [29] discussed about the "Sandhi" rules and Machine Learning models for analyzing the word, generating multiple words, concatenation with root word to suffix or prefix. Singh et al. [30] developed morphology based Manipuri POS tagger. Authors used dictionaries for root word, prefix and suffix. System was tested on 3784 sentences that consist of 10,917 words. The result shows that 69% words were correctly tagged while 31% of them were incorrectly tagged (23% unknown words and 8% known words).

Patil et al. [31] developed Rule based Marathi POS tagger. The system is tested with small corpus size and achieved 78.82% accuracy. Authors stated that system's accuracy can be increased by increasing the corpus size. Singh et al. [32] presented N-gram HMM for POS tagger. Authors considered tourism domain and collected 1,95,647 words for experiment purpose. Kaur et al. [33] talked about Punjabi POS tagger developed using HMM with tag set of 630 tags. Large tag set creates the data sparseness problem and it could be resolved by reducing the tag set. In this paper author suggested the new tag set proposed by Technical department of Indian languages (TDIL) and it consist of only 36 tags instead of 630 tags. The accuracy with 36 tags and 630 tags were 92-95% and 85-87% respectively. Mittal et al. [34] described N-gram HMM model for Punjabi POS tagger. Result showed that N-gram model is not suitable for unknown words because of spelling mistake or foreign language words.

Sharma et al. [35] stated that correctness of POS tagger depends on how accurately tagger tags the words of a sentence. The problem with the existing tagger is that it fails to tag the compound words and complex sentences. Authors were interested to increase the efficiency of existing Punjabi POS tagger by implementing the Viterbi algorithm of Bi-gram HMM. Suresh et al. [36] developed Telugu POS tagger using HMM with 620 tags but TDIL proposed only 34 tags for Indian languages. After experimenting 92-95% and 85-87% accuracy achieved with 34 tags and 620 tags respectively. Jagadeesh et al. [37] used unsupervised learning algorithm and Deep Learning (DL) methods for developing Telugu POS. The Table I indicate approaches for POS tagger for Indian Languages.

Al Shamsi et al. [38] used HMM to develop Arabic POS tagger and got 97% accuracy. Demilie et al. [39] developed POS tagger for Awngi language using HMM. Authors used 23 tags and 188,760 words for training and testing purpose. 93.64% and 94.77% accuracy is achieved with Uni-gram and Bi-gram HMM respectively. Purnamasari et al. [40] talked about Indonesian rule based POS tagger and authors used KBBI (Indonesian large dictionary) and morphological rules for tagging purpose.

Wicaksono et al. [41] developed POS tagger for Indonesian language using HMM. Affix tree, succeeding POS tag and additional lexicon methods were used to improve the accuracy. The result stated that affix tree and additional lexicon methods are best to improve the accuracy of POS tagger than succeeding POS tag. Dibitso et al. [42] developed Setswana African Language POS using SVM. Authors reviewed POS taggers for different African languages and identified challenges and techniques. Table II shows approaches for POS tagger for International Languages from 2006 to 2019 duration.

TABLE I. APPROACHES FOR POS TAGGER FOR INDIAN LANGUAGES

S. No	Author(s)	Language	Year	Approach
1	Basit et al. [7]	Awadhi	2008	FBA
2	Ekbal et al. [8]	Bengali	2008	MEA
3	Proisl et al. [9]	Bhojpuri & Magadhi	2019	SoMeWeTa, MECDN,Bi-LSTM+CRF
4	Ojha et al.[10]	Bhojpuri	2015	SVM, CRF
5	Singh et al.[11]	Bhojpuri	2015	SVM
6	Pandey et al. [12]	Chhattisgarhi	2018	RBA
7	Sinha et al. [13]	Chhattisgarhi	2018	RBA
8	Reddy et al. [14]	Cross Language	2011	HMM
9	Bhirud et al. [15]	Generic	2017	CL
10	Verma et al. [16]	Generic	2017	ML
11	Bhatt et al. [17]	Gujrati	2019	HMM, RBA
12	Sharma et al. [18]	Hindi	2020	RBA, SA, HA
13	Narayan et al. [19]	Hindi	2014	ANN
14	Narayan et al. [20]	Hindi	2014	QNN
15	Mohnot et al. [21]	Hindi	2014	HA
16	Joshi et al. [22]	Hindi	2013	HMM
17	Garg et al. [23]	Hindi	2012	RBA
18	Shrivastava et al. [24]	Hindi	2008	HMM
19	Dalal et al. [25]	Hindi	2006	MEMM
20	Antony et al. [26]	Kannada	2010	SVM
21	Priyadarshi et al. [27]	Maithili	2020	CRF
22	Mundotiya et al. [28]	Maithili	2020	CRF
23	Jha et al. [29]	Maithili	2018	RBA
24	Singh et al. [30]	Manipuri	2008	MBA
25	Patil et al. [31]	Marathi	2014	RBA
26	Singh et al. [32]	Marathi	2013	N-gram HMM
27	Kaur et al. [33]	Punjabi	2015	HMM
28	Mittal et al. [34]	Punjabi	2014	HA
29	Sharma et al. [35]	Punjabi	2011	Bi-gram HMM
30	Suresh et al. [36]	Telugu	2019	HMM
31	Jagadeesh et al. [37]	Telugu	2016	DL

TABLE II. APPROACHES FOR POS TAGGER FOR INTERNATIONAL LANGUAGES

S.No	Author(s)	Language	Year	Approach
1	Al Shamsi et al. [38]	Arabic	2006	HMM
2	Demilie et al. [39]	Awangi	2019	HMM
3	Purnamasari et al. [40]	Indonesian	2018	RBA
4	Wicaksono et al. [41]	Indonesian	2010	HMM
5	Dibitso et al. [42]	Setswana African	2019	SVM

Kovida et al. [43] discussed General Approaches (GA) used for language independent "Sandhi" Splitter and the system has been tested on two languages Telugu and Malayalam. Devadath et al. [44] conducted "Sandhi" splitting experiment on Dravidian languages. Authors evaluated the performance of "Sandhi" splitting tool and analyzed error propagation rate. Joshi et al. [45] presented "Sandhi" viched ("Sandhi" Splitter) using different Hindi rules. They experimented their system on 847 Hindi compound words and got 75% accuracy. Gupta et al. [46] developed a Rule based "Sandhi" Viched system for Hindi Language. The authors tested the system on more than 200 words and got 60% to 80% accuracy. Deshmukh et al. [47] compared four "Sandhi" analyzer and "Sandhi" Splitter systems developed in Sanskrit, Marathi, Hindi and Malayalam and authors found that RBA was used for all four languages.

Murthy et al. [48] developed first "Sandhi" Splitter in Kannada using "Sandhi" Place Determination (SPD) and Prefix Suffix method (PSM). The experiment was performed on 7000 words in Kannada language and achieved 80% accuracy. Shashirekha et al. [49] presented RBA based agama "Sandhi" Splitter namely Yakaragama and Vakaragama. The experiment was tested on the words taken from Kannada newspaper and online resources. The developed system achieved 98.85% accuracy.

Shree et al. [50] proposed Kannada "Sandhi" Splitter using CRF method. Sebastian et al. [51] discussed the results and issues of Malayalam word Splitter developed using Machine Learning (ML) approaches. Premjith et al. [52] used DL methods such as RNN, LSTM and Gated Recurrent Units (GRU) for constructing and splitting the words and obtained 98.08%, 97.88% and 98.16% accuracy respectively. Nisha et al. [53] developed the Malayalam "Sandhi" Splitter using Memory Based Language Processing (MBLP) algorithm. This algorithm was based on suffix separation. Authors discussed three methods for suffix separation such as Root driven method, Affix stripping method and the Suffix stripping method. Devadath et al. [54] developed the Malayalam "Sandhi" Splitter using the HA and got 91.1% accuracy and authors stated that HA was better than RBA and SA, because it is faster and more accurate.

Das et al. [55] developed Malayalam "Sandhi" Splitter using HA and Malayalam characters were represented by unicode. Nair et al. [56] developed Malayalam "Sandhi" Splitter using RBA to split the compound words. The system was tested on 4000 compound words and got 90% accuracy. Authors stated that work can be extended to other Dravidian languages because they have structural similarity. Joshi et al. [57] developed Marathi "Sandhi" Splitter using RBA. The experiment was tested on 150 words and got 70-80% accuracy. Patil et al. [58] proposed "Sandhi" viched system for Sanskrit language using RBA.

Bhardwaj et al. [59] developed Sanskrit benchmark called "Sandhi"kosh. "Sandhi"kosh includes Rule based corpus, Literature corpus, Bhagavad Gita corpus, UoH corpus and Astaadhyaayi. In this paper authors presented three most popular "Sandhi" splitting tools such as JNU tool, UoH tool and INRIA tool. All these tools refer "Sandhi"kosh for

referring any rules. All these are openly available and can be used by anyone for validating their tools.

Hellwig et al. [60] introduced Convolution Neural Network (CNN) and RNN for splitting the Sanskrit compound words and this model is also suitable for German compound words. Hellwig et al. [61] developed "Sandhi" resolution and "Sandhi" splitting system using RNN. Natarajan et al. [62] used Bayesian Word Segmentation Method (BWSM) for Sanskrit "Sandhi" Splitter. Rao et al. [63] focused Consonant and Phrase based "Sandhi" splitting for Telugu language. Vempaty et al. [64] developed a "Sandhi" Splitter for Telugu language by using Finite State Automata (FSA). The corpus size is 158K words and authors got 80.30% accuracy on 500 words. Table III depicts approaches for "Sandhi" Splitter for Indian Languages.

Adhikari et al. [65] discussed the rules for improving the existing Nepali morphological analyzers. Paul et al. [66] discussed about the Nepali stemmer developed using an affix stripping technique and rule based technique. The system was tested on 1800 words of different domain. These domains include news on Economics, Health & Political in Nepali language, which are based on Devanagari Script. The overall accuracy of the designed system was 90.48%. Basapur et al. [67] stated that developing a "Sandhi" Splitter or "Sandhi" joiner for Pali language is bit difficult because the complex nature of grammar rules. The Table IV represents approaches for "Sandhi" Splitter for International Languages from 2014 to 2020.

Hemlata et al. [68] stated that translation is the process of changing the words from one language to the other language without altering the meaning. Translation is a difficult task because it involves large no. of Ras and Alankaar. These help to enhance the beauty of the literature. Ramcharitmanas is an Awadhi epic which has a tremendous usage of Alankaar. It can be translated through machine, but doing so will deplore the beauty of the epic. Authors did this work better with the help of Human Intelligence (HI).

Das et al. [69] stated parse structure and simple sentence generation algorithm are used to generate simple sentences from the complex or compound sentences. Sharma [70] stated two things. Firstly, sentence simplification methods are used to simplify compound sentences. Secondly the RBA, HMM POS tagger and lexicon based morph are used to identify syntactic errors. On testing, the system got 93.30% precision, 97.32% recall rate and 95.25% F measures. Garain et al. [71] stated that sentences can be simplified by preparing parse tree and their efficiency could be decided on the basis of parse tree's efficiency. Poornima et al. [72] defined the RBA for sentence simplification. It is a two-step process. In first step, split the sentence by seeing the delimiter and in second step again split the sentence by seeing the connectives. Zhu et al. [73] stated that sentence simplification process consists of source and target. Complex sentence and simple sentence could be source and target. Tree based simplification model is used for splitting, dropping, reordering and substitution.

As discussed above, although some papers on sentence simplification were found, no papers were found on "Samaas" Finder for any Indian language.

TABLE III. APPROACHES FOR "SANDHI" SPLITTER FOR INDIAN LANGUAGES

S.No	Author(s)	Language	Year	Approach
1	Kovida et al [43]	Agglutinative Language	2011	GA
2	Devadath et al [44]	Dravidian	2016	GA
3	Joshi et al [45]	Hindi	2016	RBA
4	Gupta et al [46]	Hindi	2009	RBA
5	Deshmukh et al [47]	Indian Language	2014	GA
6	Murthy et al[48]	Kannada	2017	SPD, PSM
7	Shashirekha et al[49]	Kannada	2016	RBA
8	Shree et al[50]	Kannada	2016	CRF
9	Sebastian et al [51]	Malayalam	2020	ML
10	Premjith et al [52]	Malayalam	2018	DL
11	Nisha et al [53]	Malayalam	2016	MBLP
12	Devadath et al [54]	Malayalam	2014	RBA, SA
13	Das et al [55]	Malayalam	2012	RBA,ML
14	Nair et al [56]	Malayalam	2011	RBA
15	Joshi et al [57]	Marathi	2012	RBA
16	Patil et al [58]	Sanskrit	2018	RBA
17	Bhardwaj et al [59]	Sanskrit	2018	RBA
18	Hellwig et al [60]	Sanskrit	2018	ANN, QNN
19	Hellwig et al [61]	Sanskrit	2015	RNN
20	Natarajan et al [62]	Sanskrit	2011	BWSM
21	Rao et al [63]	Telugu	2014	RBA
22	Vempaty et al [64]	Telugu	2011	FSA

TABLE IV. APPROACHES FOR "SANDHI" SPLITTER FOR INTERNATIONAL LANGUAGES

S.No	Year	Author(s)	Language	Approach
1	2020	Adhikari et al. [65]	Nepali	RBA
2	2014	Paul et al. [66]	Nepali	RBA
3	2019	Basapur et al. [67]	Pali	GA

III. ANALYSIS OF NLP TECHNIQUES FOR INDIAN LANGUAGES

After analyzing the contents of Table I and Table III, we find that fifteen techniques are used for POS tagging and nine techniques are used for "Sandhi" splitting for many Indian Languages. Very less work is done for "Alankaar" Finder and no work is done for "Samaas" finder. Table V indicates POS tagger approaches abbreviation, Table VI represents "Sandhi" Splitter approaches abbreviation and Table VII indicates "Alankaar" Finder approach abbreviation.

Various graphs have been prepared by considering the different parameters. Fig. 1 shows Language-wise available POS tagger. Fig. 2 is for No. of approaches used by POS tagger and Fig. 3 is year-wise POS tagger.

TABLE V. POS TAGGER APPROACHES AND ITS ABBREVIATION

S.No	Approach name	Abbreviation
1	Rule Based Approach	RBA
2	Stochastic Approach	SA
3	Hybrid Approach	HA
4	Artificial Neural Network	ANN
5	Quantum Neural Network	QNN
6	Hidden Markov Model	HMM
7	Maximum Entropy Markov Model	MEMM
8	N Gram Markov Model	NGMM
9	Feature Based Approach	FBA
10	Conditional Random Field	CRF
11	Support Vector Machine	SVM
12	Morphology Based Approach	MBA
13	Author has not provided the details	SoMeWeTa
14	Bi-Long Short Term Memory	Bi-LSTM
15	Maximum Entropy Cyclic Dependency Network	MECDN

TABLE VI. "SANDHI" SPLITTER APPROACHES AND ITS ABBREVIATION

S.No	Approach name	Abbreviation
1	Rule Based Approach	RBA
2	Deep Learning	DL
3	Machine Learning	ML
4	Conditional Random Field	CRF
5	Memory Based Language Processing	MBLP
6	Bayesian Word Segmentation Method	BWSM
7	Finite State Automata	FSA
8	"Sandhi" Place Determination	SPD
9	Prefix and Suffix Method	PSM

TABLE VII. "ANANKAAR" FINDER APPROACH AND ITS ABBREVIATION

S.No	Approach name	Abbreviation
1	Human Intelligence	HI

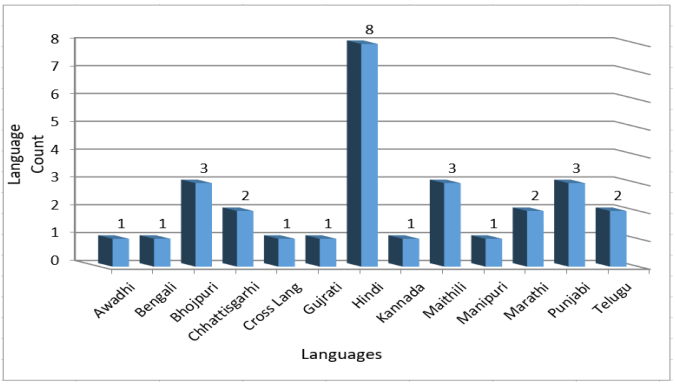

Fig. 1. Language-Wise Available POS Tagger.

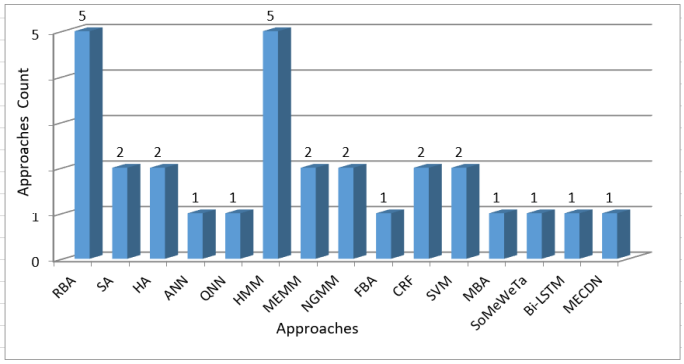

Fig. 2. No. of Approaches used by POS Taggers.

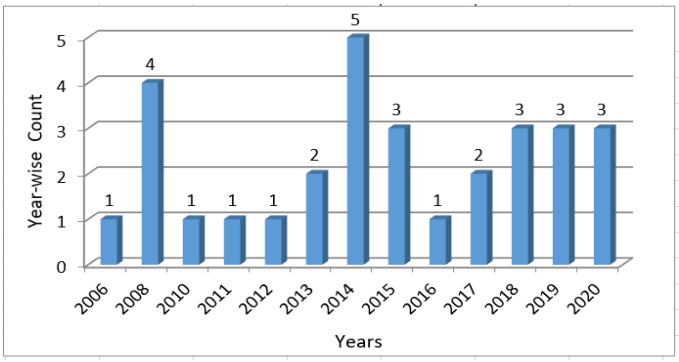

Fig. 3. Year-Wise POS Tagger.

Different graphs have been made for "Sandhi" Splitter. Fig. 4 represent language-wise available "Sandhi" Splitter. Fig. 5 shows the No. of approaches used by "Sandhi" Splitter and Fig. 6 is year-wise "Sandhi" Splitter.

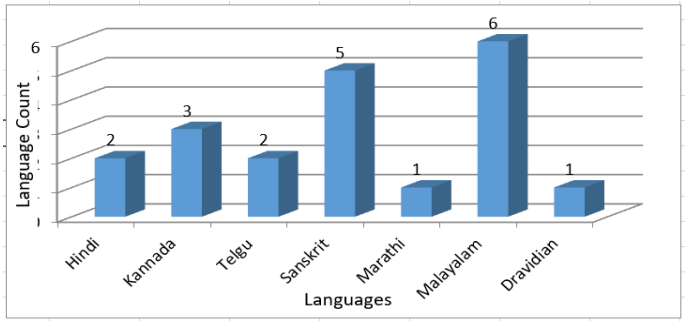

Fig. 4. Language-Wise Available "Sandhi" Splitter.

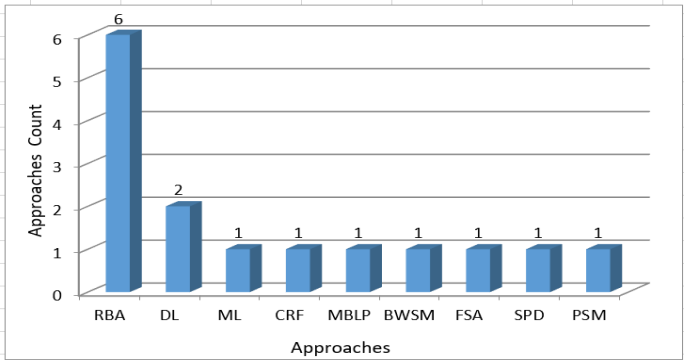

Fig. 5. No. of Approaches used by "Sandhi" Splitter.

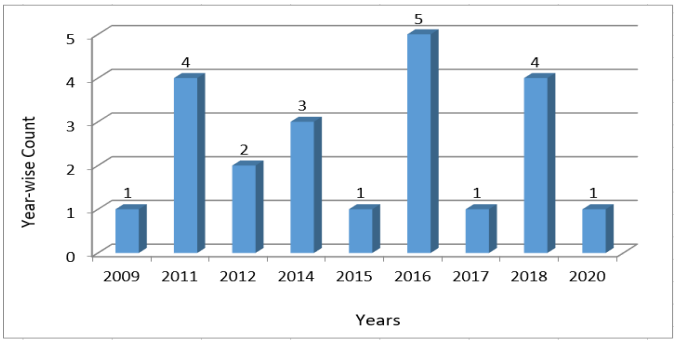

Fig. 6. Year-Wise "Sandhi" Splitter.

Fig. 7 shows the various approaches used by different Computational Linguistic tools.

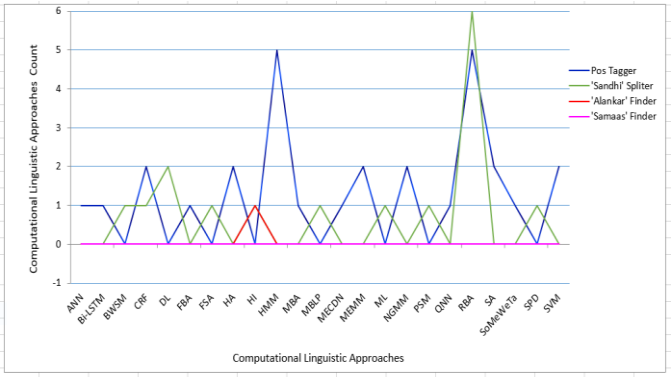

Fig. 7. Different Approaches used by POS Tagger, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder.

After reviewing all research papers, it is observed that most of the Computational Linguistics work is done in Maharashtra, Punjab, Telangana, Tamil Nadu and Uttar Pradesh. Table VIII depicts the state wise statistics.

Fig. 8 shows the Political map of India [74] and the state wise linguistic work are represented on the map.

TABLE VIII. COMPUTATIONAL LINGUISTICS STATISTICS STATE WISE

State	Count	State	Count
Andhra Pradesh	2	Madhya Pradesh	1
Assam	1	Maharashtra	8
Bihar	1	Meghalaya	1
Chhattisgarh	3	Punjab	7
Delhi	3	Rajasthan	2
Gujrat	1	Tamil Nadu	4
Haryana	1	Telangana	5
Jharkhand	1	Uttar Pradesh	4
Karnataka	3	West Bengal	3
Kerala	4		

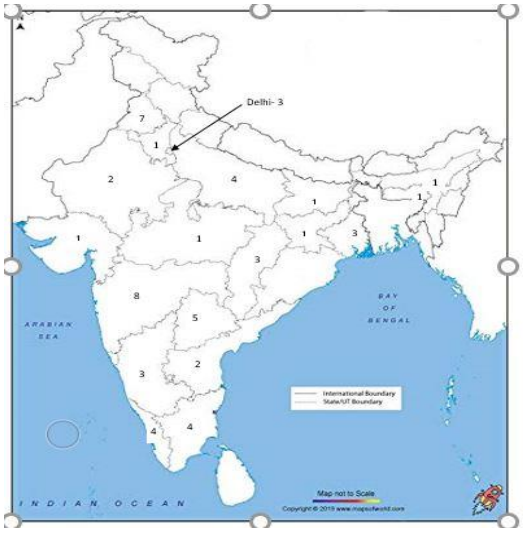

Fig. 8. State Wise Computational Linguistic Work.

IV. CONCLUSION AND FUTURE WORK

Linguistic techniques are helpful for understanding the natural languages. Four Computational Linguistic tools namely POS tagger, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder for Indo-Aryan and Dravidian languages have been considered. It is observed that POS tagger and "Sandhi" Splitter are available while "Alankaar" Finder and "Samaas" Finder are not. Most of the POS taggers are available only for Hindi language while "Sandhi" splitters are available mostly for Malayalam language. Fifteen techniques such as RBA, SA, HA, ANN, QNN, HMM, MEMM, N-gram HMM, FBA, CRF, SVM, MBA, Bi-LSTM and MECDN are suitable for POS tagging. It is observed that most of the Indian language POS taggers are built by using RBA and HMM.

Nine techniques namely RBA, DL, ML, CRF, MBLP, BWSM, FSA, SPD and PSM are appropriate for "Sandhi" Splitter. RBA is commonly used by researchers for developing "Sandhi" Splitter. The study shows that HI could be used for "Alankaar" Finder. But technique for "Samaas" Finder are unavailable yet.

As a future work, the authors would like to extend this work and use ML techniques for linguistic tools i.e. POS tagger, "Sandhi" Splitter, "Alankaar" Finder and "Samaas" Finder for Indo-Aryan and Dravidian languages.

REFERENCES

[1] The Constitution Of India, (2019). Government Of India Ministry Of Law and Justice Legislative Department.

[2] Information from Omniglot (2008), The online encyclopedia of writing systems and languages.

[3] Internet Archive, (2007), Linguistic Survey of India Vol. 6.

[4] Ramcharitmans, (2015), Tulsidas Ramcharitmanas.

[5] The Divine India, (2020), Hanuman Chalisa.

[6] Kavitakosh, (2006),Ramcharitmans/Tulsidas.

[7] Basit, A., & Kumar, R. (2019) Towards a Part-of-Speech Tagger for Awadhi: Corpus and Experiments.

[8] Ekbal, A., Haque, R., & Bandyopadhyay, S. (2008). Maximum entropy based Bengali part of speech tagging. Advances in Natural Language Processing and Applications, Research in Computing Science (RCS) Journal, 33, 67-78.

[9] Proisl, T., Uhrig, P., Blombach, A., Dykes, N., Heinrich, P., Kabashi, B., & Mammarella, S. (2019). The_Illiterati: Part-of-Speech Tagging for Magahi and Bhojpuri without even knowing the alphabet. In Proceedings of the First International Workshop on NLP Solutions for Under Resourced Languages (NSURL 2019) co-located with ICNLSP 2019-Short Papers (pp. 73-79).

[10] Ojha, A. K., Behera, P., Singh, S., & Jha, G. N. (2015). Training & evaluation of POS taggers in Indo-Aryan languages: a case of Hindi, Odia and Bhojpuri. In the proceedings of 7th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics (pp. 524-529).

[11] Singh, S., & Jha, G. N. (2015, August). Statistical tagger for Bhojpuri (employing support vector machine). In 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1524-1529). IEEE.

[12] Pandey, V., Padmavati, M. V., & Kumar, R. Rule Based Parts of Speech Tagger for Chhattisgarhi Language.

[13] Sinha, S.K. Sahu, & S ther (2018). Parts of speech tagging for Chhattisgarhi language. International journal of creative research thoughts (Volume 6, Issue 1 February 2018, ISSN: 2320-2882.

[14] Reddy, S., & Sharoff, S. (2011, November). Cross language POS taggers (and other tools) for Indian languages: An experiment with Kannada using Telugu resources. In Proceedings of the Fifth International Workshop On Cross Lingual Information Access (pp. 11-19).

[15] Bhirud, N. S., Bhavsar, R., & Pawar, B. (2017). Grammar checkers for natural languages: a review. International Journal on Natural Language Computing (IJNLC), 6(4), 51-62.

[16] Verma, M. (2017). Lexical analysis of religious texts using text mining and machine learning tools. International Journal of Computer Applications, 168(8), 39-45.

[17] Bhatt, P. M., & Ganatra, A. Analyzing& enhancing accuracy of part of speech tagger with the usage of mixed approaches for Gujarati. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277, 3878.

[18] Sharma, A., & Yadav, V. Approaches to Part of speech Tagging in Hindi Language: A Review.

[19] Narayan, R., Chakraverty, S., & Singh, V. P. (2014). Neural network based parts of speech tagger for Hindi. IFAC Proceedings Volumes, 47(1), 519-524.

[20] Narayan, R., Singh, V. P., & Chakraverty, S. (2014). Quantum neural network based parts of speech tagger for Hindi. International Journal of Advancements in Technology, 5(2), 137-152.

[21] Mohnot, K., Bansal, N., Singh, S. P., & Kumar, A. (2014). Hybrid approach for Part of Speech Tagger for Hindi language. International Journal of Computer Technology and Electronics Engineering (IJCTEE), 4(1), 25-30.

[22] Joshi, N., Darbari, H., & Mathur, I. (2013). HMM based POS tagger for Hindi. In Proceeding of 2013 International Conference on Artificial Intelligence, Soft Computing (AISC-2013) (pp. 341-349).

[23] Garg, N., Goyal, V., & Preet, S. (2012, December). Rule based Hindi part of speech tagger. In Proceedings of COLING 2012: Demonstration Papers (pp. 163-174).

[24] Shrivastava, M., & Bhattacharyya, P. (2008, December). Hindi POS tagger using naive stemming: harnessing morphological information without extensive linguistic knowledge. In International Conference on NLP (ICON08), Pune, India.

[25] Dalal, A., Nagaraj, K., Sawant, U., & Shelke, S. (2006). Hindi part-of-speech tagging and chunking: A maximum entropy approach. Proceeding of the NLPAI Machine Learning Competition.

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

[27] Priyadarshi, A., & Saha, S. K. (2020). Towards the first Maithili part of speech tagger: Resource creation and system development. Computer Speech & Language, 62, 101054.

[28] Mundotiya, R. K., Singh, M. K., Kapur, R., Mishra, S., & Singh, A. K. (2020). Basic Linguistic Resources and Baselines for Bhojpuri, Magahi and Maithili for Natural Language Processing. arXiv preprint arXiv:2004.13945.

[29] Jha, S. K., Singh, P. P., & Kaul, V. K. (2018). VEA Model in Word Formation Process of Maithili MT.

[30] Singh, T. D., & Bandyopadhyay, S. (2008). Morphology driven manipuripos tagger. In Proceedings of the IJCNLP-08 Workshop on NLP for less privileged languages.

[31] Patil, H. B., Patil, A. S., & Pawar, B. V. (2014). Part-of-Speech Tagger for Marathi Language using Limited Training Corpora. International Journal of Computer Applications, 975, 8887.

[32] Singh, J., Joshi, N., & Mathur, I. (2013, August). Development of Marathi part of speech tagger using statistical approach. In 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1554-1559). IEEE.

[33] Kaur, M., Aggerwal, M., & Sharma, S. K. (2014). Improving Punjabi Part of Speech Tagger by Using Reduced Tag Set. International Journal of Computer Applications & Information Technology, 7(2), 142.

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

[35] Sharma, S. K., & Lehal, G. S. (2011, June). Using hidden markov model to improve the accuracy of Punjabi pos tagger. In 2011 IEEE International Conference on Computer Science and Automation Engineering (Vol. 2, pp. 697-701). IEEE.

[36] Suresh, V. Reduced TagsetTo Improve Accuracy of HMM Based Parts of Speech Tagger in Telugu Language.

[37] Jagadeesh, M., Kumar, M. A., & Soman, K. P. (2016). Deep belief network based part-of-speech tagger for Telugu language. In Proceedings of the Second International Conference on Computer and Communication Technologies (pp. 75-84). Springer, New Delhi.

[38] Al Shamsi, F., & Guessoum, A. (2006, April). A hidden Markov model-based POS tagger for Arabic. In Proceeding of the 8th International Conference on the Statistical Analysis of Textual Data, France (pp. 31-42).

[39] Demilie, W. B. (2019, September) Parts of Speech Tagger for Awngi Language. International journal of Engineering Science and Computing (Vol. 9, Issue No 9).

[40] Purnamasari, K. K., & Suwardi, I. S. (2018, September). Rule based Part of Speech Tagger for Indonesian Language. In IOP Conference Series: Materials Science and Engineering (Vol. 407, No. 012151, pp. 1-4).

[41] Wicaksono, A. F., & Purwarianti, A. (2010, August). HMM Based part-of-speech tagger for bahasa Indonesia. In Fourth International MALINDO Workshop, Jakarta.

[42] Dibitso, M. A., Owolawi, P. A., & Ojo, S. O. (2019, November). Part of Speech Tagging for Setswana African Language. In 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC) (pp. 1-6). IEEE.

[43] Kovida, K. P. N., Sneha, N., & Mamidi, R. (2011). Statistical Sandhi Splitter For Agglutinative Languages.

[44] Devadath, V. V., & Sharma, D. M. (2016, August). Significance of an accurate sandhi-Splitter in shallow parsing of dravidian languages. In Proceedings of the ACL 2016 Student Research Workshop (pp. 37-42).

[45] Joshi, B. K., & Kushwah, K. K. (2016). Sandhi: the rule based word formation in Hindi. International Journal of Computer Science and Information Security, 14(12), 781.

[46] Gupta, P., & Goyal, V. (2009). Implementation of rule based algorithm for Sandhi-Viched of compound Hindi words. arXiv preprint arXiv:0909.2379.

[47] Deshmukh, R., Bhojane, V., & PIIT, N. P. (2014). Sandhi Splitting Techniques For Different Indian Languages. International Journal of Engineering Technology, Management and Applied Sciences (ijetmas), 2(7).

[48] Murthy, S. R., Akshatha, A. N., Upadhyaya, C. G., & Kumar, P. R. (2017, September). Kannada spell checker with sandhi Splitter. In 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 950-956). IEEE.

[49] Shashirekha, H. L., & Vanishree, K. S. (2016, September). Rule based Kannada Agama Sandhi Splitter. In 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 549-553). IEEE.

[50] Shree, M. R., Lakshmi, S., & Shambhavi, B. R. (2016, October). A novel approach to Sandhi splitting at character level for Kannada language. In 2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) (pp. 17-20). IEEE.

[51] Sebastian, M. P., & Kumar, G. S. (2020). Machine learning approach to suffix separation on a sandhi rule annotated malayalam data set. South Asia Research, 40(2), 231-249.

[52] Premjith, B., Soman, K. P., & Kumar, M. A. (2018). A deep learning approach for Malayalam morphological analysis at character level. Procedia computer science, 132, 47-54.

[53] Nisha, M., & Raj, P. R. (2016). Sandhi Splitter for malayalam using mblp approach. Procedia Technology, 24, 1522-1527.

[54] Devadath, V. V., Kurisinkel, L. J., Sharma, D. M., & Varma, V. (2014, December). A sandhi Splitter for malayalam. In Proceedings of the 11th International Conference on Natural Language Processing (pp. 156-161).

[55] Das, D., Radhika, K. T., Rajeev, R. R., & PC, R. R. (2012). Hybrid sandhi-Splitter for malayalam using unicode. In in proceedings of National Seminar on Relevance of Malayalam in Information Technology.

[56] Nair, L. R., & Peter, S. D. (2011, September). Development of a rule based learning system for splitting compound words in Malayalam language. In 2011 IEEE Recent Advances in Intelligent Computational Systems (pp. 751-755). IEEE.

[57] Joshi Shripad, S. (2012). Sandhi splitting of Marathi compound words. Int. J. on Adv. Computer Theory and Engg, 2(2).

[58] Patil, B., & Patil, M. (2018). Implementation of Sandhi Viccheda for Sanskrit Words/Sentences/Paragraphs.

[59] Bhardwaj, S., Gantayat, N., Chaturvedi, N., Garg, R., & Agarwal, S. (2018, May). Sandhikosh: A benchmark corpus for evaluating sanskrit sandhi tools. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

[60] Hellwig, O., & Nehrdich, S. (2018). Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (pp. 2754-2763).

[61] Hellwig, O. (2015, November). Using Recurrent Neural Networks for joint compound splitting and Sandhi resolution in Sanskrit. In 4th Biennial Workshop on Less-Resourced Languages.

[62] Natarajan, A., & Charniak, E. (2011, November). S3-Statistical Sandhi Splitting-Statistical Sandhi Splitting. In Proceedings of 5th International Joint Conference on Natural Language Processing (pp. 301-308).

[63] Rao, T. K., & Prasad, T. V. (2014). Telugu Bigram Splitting using Consonant-based and Phrase-based Splitting. Editorial Preface, 5(5), 122.

[64] Vempaty, P. C., & Nagalla, S. C. P. (2011). Automatic sandhi splitting method for Telugu, an Indian language. Procedia-Social and Behavioral Sciences, 27, 218-225.

[65] Adhikari, M., & Neupane, A. (2020). A vowel based word Splitter to improve performance of existing Nepali morphological analyzers on words borrowed from Sanskrit. Kathmandu University Journal of Science, Engineering and Technology, 14(1).

[66] Paul, A., Dey, A., & Purkayastha, B. S. (2014). An Affix Removal Stemmer for Natural Language Text in Nepali. International Journal of Computer Applications, 91(6).

[67] Basapur, S., Shivani, V., & Nair, S. (2019). Pāli Sandhi–A computational approach. In Proceedings of the 6th International Sanskrit Computational Linguistics Symposium (pp. 181-192).

[68] Hemlata, M. A., & Kalan, B. K. Distortion or Translation (2019). Studying Figures of Speech in Ramcharitmanasa.

[69] Das, B., Majumder, M., & Phadikar, S. (2018). A novel system for generating simple sentences from complex and compound sentences. *International Journal of Modern Education and Computer Science*, *12*(1), 57.

[70] Sharma, S. K. (2019). Sentence Reduction for Syntactic Analysis of Compound Sentences in Punjabi Language. EAI Endorsed Transactions on Scalable Information Systems, 6(20), e4.

[71] Garain, A., Basu, A., Dawn, R., & Naskar, S. K. (2019, November). Sentence simplification using syntactic parse trees. In 2019 4th International Conference on Information Systems and Computer Networks (ISCON) (pp. 672-676). IEEE.

[72] Poornima, C., Dhanalakshmi, V., Anand, K. M., & Soman, K. P. (2011). Rule based sentence simplification for english to tamil machine translation system. International Journal of Computer Applications, 25(8), 38-42.

[73] Zhu, Z., Bernhard, D., & Gurevych, I. (2010, August). A monolingual tree-based translation model for sentence simplification. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010) (pp. 1353-1361).

[74] Maps of Inida, (2020), India Outline Map with States and Union Territories.