International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 4 | ISSN : 2456-3307

Parts of Speech Tagger for Pali Language

Yashodhara Haribhakta*[1], Laxmi Nadageri[2]

*[1,2]Departmentof Computer Engineering and IT, College of Engineering, Pune, Maharashtra, India

ABSTRACT

Parts of Speech tagging is the process of labelling the words in the text with their appropriate labels. The labels assigned are noun, verb, adjective, adverb, pronoun... etc. For performing natural language processing, Parts of Speech tagging is an essential requirement. It is very simple statistical model for many Natural Language Processing applications. In this paper, we propose a parts of speech tagger for Pali language. Pali though considered as extinct, has very rich literature comprising works on Logic, History, Medicine, Pharmacology etc. It is an Indo-Aryan language. The general approach used for development of Pali tagger is a Rule based approach. It also presents the tagset used for Pali language. The paper shows the performance of proposed Rule based tagger for a dataset up to 300 sentences / 1000 words. The learning algorithms Support Vector Machine and Decision Tree have been used for measuring the performance on Pali tagged corpus.

Keywords : Parts of Speech tagging, tagger, Rule based tagger, Pali language.

I. INTRODUCTION

Pali is an Indo-Aryan language which died out as a literary language in fourteenth century but survived somewhere as sacred language which is used mainly for religious purpose. The original teachings of Gautama Buddha were written into Pali language. The initial Pali text were found written on Palm leaves. Pali Text Society founded in 1881 took initiatives to scan such palm leaves and store Pali text into electronic form. Digitalization of such historical language is important to increase usage of its literature. The work presented here is an initiative step to create an annotated language resources to use in an applications of Natural Language Processing (NLP) like language learning, language teaching, text classification, question answering etc. The first attempt of creating an annotated language resource is Parts of speech (POS) tagger. POS tagging is a process of assigning a category to a word from a defined set of categories belonging to a language.

The main objective of proposed work is to build a POS tagger for Pali language. The approach used is Rule based. The Rule based tagger uses Pali grammar rules. Words of Pali text are labelled by looking at the affixes attached to each word. In case there is an ambiguity while assigning label, it is resolved looking at label of the context words. The Pali tagset comprises of 16 tags including unknown tag UN. Using this tagset the data from www.tipitaka.org has been used for manual labelling. So, a manually labelled dataset of 300 sentences has been generated.

The second objective is an evaluation of the performance of machine learning algorithms on labelled Pali dataset. Two popular learning methods are used: Support Vector Machine (SVM) and Decision Tree (DT). LibLinear and J48 are the implementations of SVM and DT respectively. They are used by optimizing their parameters to achieve the maximum accuracy. All the experiments done on learning algorithms and their results are discussed in this paper.

In section 2 shows related work done on POS taggers for Indian languages. In section 3 Pali language is introduced. Pali dataset is described in section 4. General introduction of POS taggers is given in section 5. Section 6 describes POS tagger for Pali language and results are discussed in section 7. Machine learning

algorithms are discussed in section 8. Section 9 shows different experiments conducted to achieve parameter optimization of learning algorithms along with attribute selection and handling unknown data and their results. Section 10 concludes the paper.

II. RELATED WORK

This section discusses the previous work done on POS taggers for many Indian languages. Here we are focusing mainly on rule based approach for POS tagging.

Different POS tagging approaches have been proposed for Hindi language. One of them is Morphology driven tagger [10]. This tagger uses only affix information of the word for POS tagging without any contextual information. However, to identify the main verb from verb group, the previous and the next word in verb group have been taken into consideration. To identify other POS categories, lexicon lookup has been used. The tagger has been evaluated using 4-fold cross validation and resulted in 93.45% accuracy.

Punjabi POS tagger was developed by using hand written grammar rules of Punjabi language [5]. Context information has been used to disambiguate the part of speech information. Total 630 tags have been proposed for Punjabi rule based POS tagger. This tagset consists of various word classes, word specific tags, tags for punctuation etc. Out of 630 proposed tags only 503 tags were used during POS tagging process. A disambiguation rules were stored in the database. Later the disambiguated text was passed for marking verbal operators to make structure of verb phrase more understandable. An average accuracy achieved by proposed Punjabi rule based tagger was 80.29% including unknown words and 88.86% excluding unknown words.

Different POS taggers were developed for Telugu language. One of them was rule based tagger [9]. The Telugu rule based tagger consists of series of different modules. Sentence tokenizer converts each sentence into tokens. Telugu morphological analyser gives all possible analysis of each word. Morph to POS translator converts all possible morph analysis into POS tag. POS disambiguator reduces the POS ambiguity of previous module by applying uni-gram and bi-gram rules Finally annotator produces tagged text. By applying rule based tagger to Telugu text 98% accuracy was obtained.

III. PALI LANGUAGE

A. Order of Pali Sentence

The order of the Pali sentence is flexible. In general, the order of simple Pali sentence is subject-object-predicate. Sometimes, other orders could also be found in literature and in colloquial practice. For example, Adjectives come before the subject/object and adverbs before the verb. However, adverbs of "time" always come first in the sentence.

B. Morphology

When suffixes attach to the root word, morphological changes take place.

For example,

√ lag + na = lagga: progressive assimilation i.e. initial consonant is assimilated to the final consonant of preceding word.

√ lip + ta = litta: regressive assimilation, final consonant of preceding word is assimilated to the initial consonant of following word.

C. Grammatical Gender

Pali has three genders: the Masculine, the Feminine, and the Neuter. However, Pali does not follow division of male, female etc. in assigning gender to nouns. Many masculine nouns in English are Feminine or Neuter in Pali. Some neuter nouns are masculine or feminine in Pali.

All nouns ending in a (ā is Latin letter; pronounced as ā in father) are feminine, but there are few masculine nouns ending in ā like,

Sā = dog,

Mā = moon

IV. THE DATASET

A small size corpus used in this experiment was collected from Pali Tipitaka (http://www.tipitaka.org/). The corpus consists of 300 sentences or approximately 1000 tokens. Each sentence consists of 3 to 4 words only. We selected simple sentences for corpus

generation for the sake of simpler POS tagging task. The collected data was not much noisier; hence it was processed just to arrange each sentence in separate line. Later, the data was tagged manually and used it as benchmark data for all next experiments discussed in subsequence sections. Final version of annotated Pali corpus consists of 547 nouns, 255 verbs and remaining other categories.

The dataset contains approximately 300 repeated words. That is, 70% of corpus contains all unknown words.

Pali Tagset: For tagging experiment, 15 different Pali tags are used (tagset shown in Appendix A). All the tags are derived from Duroiselle grammar. Tag sets consists of 5 basic tags (noun, adjective, adverb, verb, and pronoun) and remaining sub-tags (demonstrative pronoun, interrogative pronoun, verb basic, verb gerund, verb past participle etc.). In addition to these tags, UN (unknown) tag is used to annotate the words not belonging to set of 15 tags. There is only one word in an entire Pali corpus which got "UN" tag.

V. POS TAGGING

POS tagging is the process of assigning category to each word in the sentence. Categories include noun, verb, adjective, adverb and so on. POS taggers are the programs who does the job of assigning categories to each word. POS taggers fall mainly into two categories. 1. Rule based POS taggers and 2. Probabilistic POS taggers

A. **Rule based tagger**

Rule based taggers uses hand-written rules of specific language derived from sentence structure, word suffix/prefix, position of word in the sentence etc. Advantages of rule based taggers [1] over probabilistic taggers are that they require less stored information, simplified small set of rules, ease of implementation etc. [1] proposed 5.1% error rate on 5% of Brown corpus tagged using Rule based tagger.

B. Probabilistic tagger

Probabilistic POS taggers uses probability information of tag sequence to assign tag to each word. Probabilistic tagger builds tagging model using training corpus. Once the model is built the automatic tagging of sentences is done by selecting high probability tag sequence from the model. These taggers are more popular than the rule based taggers because of their tagging accuracy. Most accurate state of the art POS taggers are based on probabilistic methods. The size of training corpus plays important role in this type of taggers. Large the training corpus, more probability information can be derived.

VI. RULE BASED TAGGER FOR PALI LANGUAGE

The proposed Rule based tagger has been built for Pali language using only Pali sentence rules. The tagger is built using two types of rules: Open rules and Closed rules. Open rules include list of affixes that distinguish nouns, verbs, adjectives and adverbs. Closed rules are whole words that can be used in the sentence without any declension or an affix. Closed rules contain pronouns, numerical values, some adverbs which are not covered in open rules.

A. Open Rules

Following list shows affixes used in open rules.

1) Noun suffix: All declension of nouns whose nominative declension ends in a, ā, i, ī, u, ū, o, e etc. and other declensions of nouns are also considered.
2) Adjective suffix: Words ending with anta, antā, manta, mantā, vanta, vantā etc.
3) Adverb suffix: Words ending in atho, anto, adho, idāni, kho, vata etc.
4) Verb suffix: Words ending in ti, nti, si, esi, mi, mhi, tha, etha, hi etc.
5) Verb gerund suffix: Words ending in itvā, tvā, tvāna, tūna, tya etc.
6) Verb Aorist prefix: Words starting with a.

If any word having an affix matching with any of an open rule then the respective POS tag will be returned for that word.

B. Closed Rules

Following list will show few closed rules for each category.

i. Personal pronoun: "ahaṃ", "maṃ", "mamaṃ", "mayā", "me", etc.
ii. Demonstrative pronoun: "taṃ", "tena", "tassa", "tamhā", "tasmā", etc.

iii. Relative pronoun: “yo”, “yaṃ”, “yad”, “yena”, etc.
iv. Interrogative pronoun: “ko”, “kaṃ”, “kena”, “kassa”, “kissa”, “kasmā”, etc.
v. WH pronoun: “koci”, “kassaci”, “kañci”, “kiñci”, “kenaci”, etc.
vi. Numerical:“eka”, “paṭhama”, “dve”, “dutiya”, “tayo”, etc.

If any word is matching (whole word matching) with any of above closed rules then the respective POS tag will be returned for that word.

Resolving an ambiguity: Ambiguities occur if any word matching with two open rules. Consider following example to understand an ambiguity in Pali.

naro (a man) has suffix= “o” and POS= “noun”;
Pandito (wise) has suffix= “o” and POS= “adjective”.

Such ambiguities can be resolved using words position. Position of an adjective could be near the word it qualifies. Adjective qualifies noun or pronoun. It has been observed that many Pali sentences contains more than one consecutive nouns. Many nouns and adjectives share common suffix. Hence, the heuristic of word position for noun and adjective will fail in this case. For this reason, only few possible adjectives with specific suffixes have been handled using open rules described in open rule list.

However, following heuristics are used successfully to resolve an ambiguity.

1) Heuristic 1: Adverb comes before the verb. This heuristic was used to resolve the noun and adverb ambiguity.For example, the word ending with “aṃ” could be noun as well as an adverb. If the word passes this heuristic then it will be an adverb.

2) Heuristic 2: Some numerical words act as adjectives; those are handled by closed rules as well as open rules. For example, eka(one) is numerical word; aneka (many) is an adjective with prefix “an”.

3) Heuristic 3: Predicate comes last in the sentence. Pali sentences could be simple or complex. Simple sentence follows Subject-Object-Predicate structure. Complex sentences can be an exception. Sentence being simple or complex, predicate always comes last. This heuristic is used to assign verb tag.

Here, the Rule Based algorithm is explained.

Algorithm 1 Rule Based POS tagger algorithm

```
while not end of the file do
   Read each sentence of the untagged Pali file
for each word in the sentence do

for i < length(ClosedRules) do
if word.equals(ClosedRules[i]) then
return respective POS tag
end If
end for

for j < length(OpenRules) do
if wordSuffix.equals(OpenRules[j]) then
return respective POS tag
else if ambiguity then
apply heuristic rules
else
return UN tag
end If
end for

end for
end while
```

VII. RESULT OF RULE BASED PALI POS TAGGER

Rule based PALI POS tagger performance can be observed from Table I. It has achieved better accuracies for noun, adverb, verb represented by open rules. For tags represented by closed rules such as pronouns, particles, coordinating conjunction; proposed tagger has achieved 100% average accuracy. NaN in F-measure column means respective tag was not found.
Sample examples for evaluation of Rule based tagger.

Example 1:
Untagged sentences: dānaṃ datva garuṃ karoti
Manually tagged: dānaṃ_NN datva_RB garuṃ_NN karoti_VB
Rule based tagger: dānaṃ_NN datva_RB garuṃ_NN karoti_VB

Example 2:
Untagged sentence: ramo dhaññaṃ vikiṇṇati
Manually tagged: ramo_NN dhaññaṃ_NN vikiṇṇati_VB

Rule based tagger: ramo_NN dhaññaṃ_RB vikiṇṇati_VB

Example 3:
Untagged sentence: vijjayā abhāvate jivā jarāmaranayo cakke baddhate
Manually tagged: vijjayā_NN abhāvate_NN jivā_NN jarāmaranayo_JJ cake_NN baddhate_VB
Rule based tagger: vijjayā_NN abhāvate_NN jivā_JJ jarāmaranayo_NN cake_NN baddhate_NN

The input to the Rule based tagger is Pali text and the output is labelled Pali text. The given output is compared against manually tagged sentences. It is observed that the first example has been tagged successfully by our proposed rule based tagger.

TABLE I
RULE BASED POS TAGGER RESULT

POS tag	F-Measure
Noun	0.875121
Adjective	0.46153843
Adverb	0.824
Verb	0.9475982
Verb, Past Participle	0.46511623
Verb Gerund	0.7515152
Personal Pronoun	1.0
Demonstrative Pronoun	0.9629629
Interrogative Pronoun	NaN
WH Pronoun	1.0
Relative Pronoun	1.0
Particles	0.9642857
Coordinating Conjunction	1.0
Unknown	1.0

The sentence consists of all unambiguous words. The word "*datva*" has been identified as adverb using the suffix attached to it. Other words are also identified as noun and verb by using suffix information. For the second example, it is observed that the proposed POS tagger incorrectly identifies the word "*dhaññaṃ*" as an adverb which is manually tagged as noun. This results due to the heuristic no. 1. Similar is the case for the third sentence where the word "*jivā*" has labelled as an adjective instead of noun. Similarly, for the third sentence the last word of the sentence has been labelled as noun instead of verb. This incorrect labelling is because of the suffix attached to the word. In Pali, some verb groups do not consist of supporting verb explicitly. Word translation of "*baddhate*" in third sentence in English is "is stuck". But there is no supporting verb "*atthi/bhavati*" to represent the word "is". Hence, it is tagged as verb in manual tagging. However, because of suffix rules, our proposed tagger identifies it as noun.

VIII. PERFORMANCE EVALUATION OF RULE BASED TAGGER USING LEARNING ALGORITHMS

Machine Learning is a field of Computer science in which computers can learn without explicit programming. There are two types of machine learning algorithms. Supervised and Unsupervised.

Supervised learning algorithms trains a computer using training examples consists of inputs and target output. After sufficient learning, it can find out target output for any new input. Classification and Regression problems come into supervised learning.

Unsupervised learning algorithms trains a computer using input examples only. Example of unsupervised learning is clustering. Computer creates different clusters of input training examples and able to put new inputs in appropriate cluster.

Supervised machine learning algorithms were used to evaluate the performance of Rule based tagger by observing the learning accuracy of an algorithm. Pali corpus tagged by proposed Rule based tagger was used as an input training data for learning algorithms. Machine learning algorithms available in Weka tool [12] were used for this experiment. Two popular machine learning methods (Decision tree and Support Vector Machines (SVM)) were selected for this purpose. LibLinear [3] is one of the fastest and accurate algorithm among all SVM algorithms. Hence, we selected LibLinear for application of SVM on Pali data. To select best decision tree algorithm, we applied all four algorithms on Pali data. It can be observed from table II that J48 yields better result among all algorithms. Therefore, J48 and LibLinear were selected and used for further experiments.

TABLE III
EVALUATION OF DIFFERENT DECISION TREE ALGORITHMS

Algorithm	Accuracy
DecisionStump	62.7574%
RandomTree	69.8299%
J48	**80.752%**
REPTree	67.8603%

A. **J48**

J48 is an implementation of C4.5 algorithm [8]; the most popular among the decision tree algorithms in machine learning. C4.5 uses the concept of information entropy to build the decision tree from a set of samples in training data. At each node of the tree, C4.5 chooses the attribute with highest information gain for splitting. Information gain for attribute A is calculated as below:

Sample set S is divided into subsets by attribute A. Information gain is the measure of the amount of uncertainty in S reduced after splitting by attribute A.

$$I(A) = H(S) - \sum_{T \in T} P(T)H(T)$$

where,
H(S)- Entropy of sample S
T- Subset created after splitting S by attribute A
p(t)- number of elements in t / number of elements in S
H(t)- Entropy of t

Gebeyehu [4] used J48 algorithm for Amharic language POS tagging. He achieved accuracy of 84.9% by performing different experiments on Amharic dataset and J48 parameter optimization. Solorio and Liu [11] used J48 for POS tagging of English-Spanish code-switched text with 91.11% accuracy. Code-switched text is the mixing of languages in same conversational event. They experimented on 922 sentences collected from conversation of three bilingual persons. Patil et. al. [7] performed classification experiment on bank database using J48 and Naive Bayes algorithms. Both algorithms were evaluated on different attributes and accuracy, cost were analysed.

B. **LibLinear**

LibLinear is an open source library for linear SVM. It has inherited many features of LibSVM; a popular SVM library. It is much faster than LibSVM and SMO. LibLinear has shown more accurate results than popular state of the art taggers; Stanford tagger and SVMTool in the research of Choi and Palmer [2]. They created two tagger models by changing the parameters of LibLinear and applied both models to Wall Street Journal for training purpose and evaluated on different corpora. Dynamic selection of either model shows maximum accuracy of 97.46% on WSJ newswire corpus. For unknown data, it shows 88.40% accuracy on the same corpus.

IX. EXPERIMENTATION ON LEARNING ALGORITHMS

A. Parameter Optimization of J48

Binary Splitting of J48 tree has been used successfully in previous work on POS tagging [6]. Multi split tree creates more complex tree that do not generalize the data well, called over fitting. In contrast, binary tree splits the node based on the information gain coming from splitting on an attribute value. Result of Binary split for 10-fold CV are shown in Table III. Execution with binary split shows good results, hence next successive experiments on J48 used binary-split value "true".

TABLE IIIII
BINARY SPLIT RESULTS

Binary Split	Size of Tree	Accuracy
True	135	82.3635%
False	2024	78.4244%

B. Attribute Selection

Feature attributes are used to model the input data for any classifier. Feature attributes consist of contextual information such as previous word, next word, their POS tags, suffix of current word etc. All this information is important for learning algorithms to learn the tag sequence. In this experiment, we used 13 attributes to model the Pali input data. Features included in training data are:

current word, current word-tag, previous 3 word-tags, next 3 word-tags, suffix of current word, First character, Last character, isBeginning of Sentence, hasNumbers

Above attributes divided into Context and Current word attributes as follows.

Context attributes: next 3-word tags, previous 3-word tags

Current word attributes: suffix, First character, Last character, isBeginning of Sentence, hasNumbers

All 13 attributes used to evaluate both algorithms. Table IV shows results of J48 and LibLinear. It can be observed that LibLinear is more accurate than J48. Hence, we selected LibLinear for further experiments of attribute selection.

TABLE IVV
POS TAGGING ACCURACY WITH J48 AND LIBLINEAR

Algorithm	**Accuracy**
J48	83.0239%
LibLinear	**86.8258%**

Not all attributes are equally important. We need to select attributes with higher information gain. Though Weka tool has the facility to get an information gain of each attribute and select high gain attributes; we too have evaluated attributes by performing different operations on them and got better results than Weka's selected attributes. Operations were: adding or removing attributes, checking accuracy of tagger with different attribute combinations. Previous 3-word tags are represented by Tag-1, Tag-2, Tag-3 and Tag1, Tag2, Tag3 represents next 3-word tags. To find out most important contextual attributes, we experimented by adding and removing each context tag one at a time. Table V shows detailed experimentation of addition and removal of tags and results also. It can be observed that the previous word tag (Tag-1) and next word tag (Tag1) gives most accurate result. Hence, we selected Tag-1 and Tag1 as context attributes for further experiments.

TABLE V
BOTH SIDE FEATURES AND ACCURACY

Both side Features	**Accuracy**
Tag-3 Tag-2 Tag-1 Tag1 Tag2 Tag3	49.8674%
Tag-3 Tag-2 Tag-1 Tag1 Tag2	50.4863%
Tag-2 Tag-1 Tag1 Tag2	50.6631%
Tag-1 Tag1 Tag2	51.2821%
Tag-1 Tag1	**52.0778%**
Tag1	51.6357%
Left Side Features	**Accuracy**
Tag-1	46.6552%
Tag-1Tag-2	47.5685%
Tag-1Tag-2 Tag-3	47.5685%
Right Side Features	**Accuracy**
Tag1	51.6357%
Tag1 Tag2	51.5473%
Tag1 Tag2 Tag3	50.1326%

The Next experiment was conducted to decide the better combination of current word attributes with Tag-1 and Tag1. Table VI shows learning accuracies after removing one current word attribute at a time. It can be observed that better accuracy was achieved after removing first character of current word. This can be because of less information achieved from attribute first character. Therefore, most accurate combination of attributes is:

current word, current word-tag, previous word tag, next word tag, suffix of current word, last character of current word, is beginning of sentence.

TABLE VI
COMBINATION OF ATTRIBUTES

Attributes	**Accuracy**
Tag-1 Tag1 + All remaining attributes	87.0911%
Removed hasNumbers	87.0911%
Removed isBeginingofSen	87.0027%
Removed Last-char	86.8258%
Removed First-char	**87.1795%**
Removed Suffix	82.847%

C. Cost parameter of LibLinear

Cost parameter C was changed to see an improvement in total accuracy of LibLinear. Cost parameter is used to avoid the misclassification of training examples. Large the value of C causes small margin for classification and vice versa. LibLinear was evaluated by changing cost parameter value and for cost=2, LibLinear shows highest accuracy of 87.6216%.

D. Evaluation for Unknown data

Evaluation of learning algorithms for unknown data is important to find out the reliability of algorithm. For testing LibLinear with unknown data, original data was split into 85% training and 15% testing data manually; such that 15% testing data is totally unseen to 85% training data. Execution using LibLinear and parameter optimization, we got 76.3314% accuracy for unknown data (Table VII).

TABLE VII
TAGGING RESULTS FOR KNOWN AND UNKNOWN DATA

Type of Data	Accuracy
Known Data	100.00%
Unknown Data	76.3314%

X. CONCLUSION

The proposed Rule based tagger has shown better performance for noun, verb, and adverb with average accuracy of 90%. However, it has achieved low accuracy for adjectives. All other tags distinguished by closed rules shows around 100% accuracy. Rule based tagger can be enhanced by adding more refined rules for those tags showing low accuracy.

Performance of both machine learning algorithms were evaluated on labelled dataset and it has seen that LibLinear is the best learner as compared to J48. LibLinear has shown 87.6216%. accuracy and J48 has shown 83.0239% accuracy with all possible parameter optimization. However, the performance of J48 was not much discouraging compared to LibLinear. The training and testing speed of LibLinear was much better than J48.

The performance of tagger as well the learning algorithms can be increased by adding knowledge source such as machine-readable dictionary. However, Pali has less available resources which could be increased by engaging engineers in collaboration with Pali lexicographers.

XI. REFERENCES

[1] Eric Brill. A simple rule-based part of speech tagger. In Proceedings of the workshop on Speech and Natural Language, pages 112–116. Association for Computational Linguistics, 1992.

[2] Jinho D Choi and Martha Palmer. Fast and robust part-of-speech tagging using dynamic model selection. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2, pages 363–367. Association for Computational Linguistics, 2012.

[3] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. Journal of machine learning research, 9(Aug):1871–1874, 2008.

[4] K Gebeyehu. The application of decision tree for part of speech (pos) tagging for amharic. PhD thesis, Master Thesis, 2009.

[5] Mandeep Singh Gill, Gurpreet Singh Lehal, and Shiv Sharma Joshi. Part of speech tagging for grammar checking of punjabi. The Linguistic Journal, 4(1):6–21, 2009.

[6] Brian R Hirshman. Training set properties and decision-tree taggers: A closer look, 2009.

[7] Tina R Patil and SS Sherekar. Performance analysis of naive bayes and j48 classification algorithm for data classification. International Journal of Computer Science and Applications, 6(2):256–261, 2013.

[8] J Ross Quinlan. C4. 5: Programming for machine learning. Morgan Kauffmann, 38, 1993.

[9] RJ RamaSree and P Kusuma Kumari. Combining pos taggers for improved accuracy to create telugu annotated texts for information retrieval. Dept. of Telugu Studies, Tirupathi, India, 2007.

[10] Smriti Singh, Kuhoo Gupta, Manish Shrivastava, and Pushpak Bhattacharyya. Morphological richness offsets resource demand-experiences in constructing a pos tagger for hindi. In Proceedings of the COLING/ACL on Main conference poster sessions, pages 779– 786. Association for Computational Linguistics, 2006.

[11] Thamar Solorio and Yang Liu. Part-of-speech tagging for english-spanish code-switched text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1051– 1060. Association for Computational Linguistics, 2008.

[12] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016

APPENDIX A

PALI TAGSET

Sr. no.	Tag	Label

1	Noun	NN
2	Adjective	JJ
3	Adverb	RB
4	Verb	VB
5	Verb, Past Participle	VBN
6	Verb Gerund	VBG
7	Verb Aorist	VA
8	Particles	RP
9	Demonstrative Pronoun	PRD
10	Interrogative Pronoun	PRI
11	Personal Pronoun	PRP
12	Relative Pronoun	PRR
13	WH-Pronoun	WP
14	Coordinating Conjunction	CC
15	Cardinal Number	CD
16	Unknown	UN