Morpheme based Language Model for Tamil Part-of-Speech Tagging

S. Lakshmana Pandian and T. V. Geetha

***Abstract*—The paper describes a Tamil Part of Speech (POS) tagging using a corpus-based approach by formulating a Language Model using morpheme components of words. Rule based tagging, Markov model taggers, Hidden Markov Model taggers and transformation-based learning tagger are some of the methods available for part of speech tagging. In this paper, we present a language model based on the information of the stem type, last morpheme, and previous to the last morpheme part of the word for categorizing its part of speech. For estimating the contribution factors of the model, we follow generalized iterative scaling technique. Presented model has the overall F-measure of 96%.**

***Index Terms*—Bayesian learning, language model, morpheme components, generalized iterative scaling.**

I. Introduction

Part-of-speech tagging, i.e., the process of assigning the part-of-speech label to words in a given text, is an important aspect of natural language processing. The first task of any POS tagging process is to choose various POS tags. A tag set is normally chosen based on the language technology application for which the POS tags are used. In this work, we have chosen a tag set of 35 categories for Tamil, keeping in mind applications like named entity recognition and question and answering systems. The major complexity in the POS tagging task is choosing the tag for the word by resolving ambiguity in cases where a word can occur with different POS tags in different contexts. Rule based approach, statistical approach and hybrid approaches combining both rule based and statistical based have been used for POS tagging. In this work, we have used a statistical language model for assigning part of speech tags. We have exploited the role of morphological context in choosing POS tags. The paper is organized as follows. Section 2 gives an overview of existing POS tagging approaches. The language characteristics used for POS tagging and list of POS tags used are described in section 3. Section 4 describes the effect of morphological context on tagging, while section 5 describes the design of the language model, and the section 6 contains evaluation and results.

II. Related Work

The earliest tagger used a rule based approach for assigning tags on the basis of word patterns and on the basis of tag assigned to the preceding and following words [8], [9]. Brill tagger used for English is a rule-based tagger, which uses hand written rules to distinguish tag entities [16]. It generates lexical rules automatically from input tokens that are annotated by the most likely tags and then employs these rules for identifying the tags of unknown words. Hidden Markov Model taggers [15] have been widely used for English POS tagging, where the main emphasis is on maximizing the product of word likelihood and tag sequence probability. In essence, these taggers exploit the fixed word order property of English to find the tag probabilities. TnT [14] is a stochastic Hidden Markov Model tagger, which estimates the lexical probability for unknown words based on its suffixes in comparison to suffixes of words in the training corpus. TnT has developed suffix based language models for German and English. However most Hidden Markov Models are based on sequence of words and are better suited for languages which have relatively fixed word order.

POS tagger for relatively free word order Indian languages needs more than word based POS sequences. POS tagger for Hindi, a partially free word order language, has been designed based on Hidden Markov Model framework proposed by Scutt and Brants [14]. This tagger chooses the best tag for a given word sequence. However, the language specific features and context has not been considered to tackle the partial free word order characteristics of Hindi. In the work of Aniket Dalal *et al*. [1], Hindi part of speech tagger using Maximum entropy Model has been described. In this system, the main POS tagging feature used are word based context, one level suffix and dictionary-based features. A word based hybrid model [7] for POS tagging has been used for Bengali, where the Hidden Markov Model probabilities of words are updated using both tagged as well as untagged corpus. In the case of untagged corpus the Expectation Maximization algorithm has been used to update the probabilities.

III. Parts of Speech in Tamil

Tamil is a morphologically rich language resulting in its relatively free word order characteristics. Normally most Tamil words take on more than one morphological suffix; often the number of suffixes is 3 with the maximum going up to 13. The role of the sequence of the morphological suffixes attached to a word in determining the part-of-speech tag is an interesting property of Tamil language. In this work, we have identified 79 morpheme components, which can be combined to form about 2,000 possible combinations of integrated suffixes. Two basic parts of speech, namely, noun and verb, are mutually distinguished by their grammatical inflections. In

Manuscript received May 12, 2008. Manuscript accepted for publication October 25, 2008.

S. Lakshmana Pandian and T. V. Geetha are with Department of Computer Science and Engineering, Anna University, Chennai, India (lpandian72@yahoo.com).

Tamil, noun grammatically marks number and cases. Tamil nouns basically take on eight cases. The normal morphological derivatives of Tamil nouns are as follows:

$Stem_{Noun}$ + *[Plural Marker]+[Oblique]+[Case Marker]*

The normal morphological derivative of Tamil Verb is as follows

$Stem_{Verb}$*+[Tense Marker]+[Verbal Participle Suffix]+[Auxiliary verb +[Tense Marker]+[Person, Number, Gender]*

In addition, adjective, adverb, pronoun, postposition are also some stems that take suffixes. In this work, we have used a tagged corpus of 4,70,910 words, which have been tagged with 35 POS categories in a semiautomatic manner using an available morphological analyzer [5], which separates stem and all morpheme components. It also provides the type of stem using lexicon. This tagged corpus is the basis of our language model.

IV. Effect of Morphological Information in Tagging

The relatively free word order of Tamil normally has the main verb in a terminating position and all other categories of words can occur in any position in the sentence. Pure word based approaches for POS tagging are not effective. As described in the previous section, Tamil has a rich multilevel morphology. This work exploits this multi-level morphology in determining the POS category of a word. The stem, the pre-final and final morpheme components attached to the word that is the words derivative form normally contribute to choosing the POS category of the word. Certain sequence of morpheme can be attached to the certain stem types. Moreover, the context in which morphological components occur and its combinations also contribute in choosing the POS tag for a word. In this work, language models have been used to determine the probabilities of a word derivative form functioning as a particular POS category.

TABLE I.
List of tags for Tamil and their description

	Tag	Description
1.	N	Noun
2.	NP	Noun Phrase
3.	NN	Noun + noun
4.	NNP	Noun + Noun Phrase
5.	IN	Interrogative noun
6.	INP	Interrogative noun phrase
7.	PN	Pronominal Noun
8.	PNP	Pronominal noun
9.	VN	Verbal Noun
10	VNP	Verbal Noun Phrase
11	Pn	Pronoun
12	PnP	Pronoun Phrase
13	Nn	Nominal noun
14	NnP	Nominal noun Phrase
15	V	Verb
16	VP	Verbal phrase
17	Vinf	Verb Infinitive
18	Vvp	Verb verbal participle
19	Vrp	Verbal Relative participle
20	AV	Auxiliary verb
21	FV	Finite Verb
22	NFV	Negative Finite Verb
23	Adv	Adverb
24	SP	Sub-ordinate clause conjunction Phrase
25	SCC	Sub-ordinate clause conjunction
26	Par	Particle
27	Adj	Adjective
28	Iadj	Interrogative adjective
29	Dadj	Demonstrative adjective
30	Inter	Intersection
31	Int	Intensifier
32	CNum	Character number
33	Num	Number
34	DT	Date time
35	PO	Post position

V. Language Model

Language models, in general, predict the occurrence of a unit based on the statistical information of unit's category and the context in which the unit occurs. Language models can differ in the basic units used for building the model, the features attached to the units and the length of the context used for prediction. The units used for building language models depend on the application for which the model is used. Bayes' theorem is usually used for posterior probability estimation from statistical information. Bayes' theorem relates the conditional and marginal probabilities of stochastic events *A* and *B* as follows

$$\Pr(A/B) = \frac{\Pr(B/A)\Pr(A)}{\Pr(B)} \quad \alpha \quad L(A/B)\Pr(A) \tag{5.1}$$

where *L(A|B)* is the likelihood of *A* given fixed *B.*

Each term in Bayes' theorem is described as follows.

Pr(A) is the *prior probability* or *marginal probability* of *A*. It is "prior" in the sense that it does not take into account any information about *B*.

Pr(B) is the prior or marginal probability of *B*, and acts as a *normalizing constant.*

Pr(A|B) is the *conditional probability* of *A*, given *B*. It is

also called the posterior probability because it is derived from or depends upon the specified value of *B*.

Pr(B|A) is the conditional probability of *B* given *A*.

In this paper, we have used a language model that considers the lexical category of the stem along with morphological components of a word in order to determine its POS tag.

In case the word is equal to a stem then its POS category is the same as the stem lexical category. In case the word consists of a stem and a single morphological component then the language model is designed by considering both the lexical category of the stem and the morphological component. It is given by the equation 5.2:

$$P(\frac{pos}{root\_type, e_l}) = \alpha_1 p(\frac{pos}{root\_type}) + \alpha_2 p(\frac{pos}{e_l}) \quad (5.2)$$

where $P(\frac{pos}{root\_type, e_l})$ gives the probability of the word being tagged with the particular pos tag given a particular stem type and a particular morphological ending e_l. The two factors used for this probability calculation are $p(\frac{pos}{root\_type})$, the prior probability of a particular pos tag given a particular stem type and $p(\frac{pos}{e_l})$, the prior probability of a particular pos tag given a particular morphological ending e_l. In addition, α_1 and α_2 are contribution factors and $\alpha_1 + \alpha_2 = 1$.

In order to calculate the prior probability $p(\frac{pos}{e_l})$ we use equation 5.3:

$$P(\frac{pos}{e_l}) = \frac{P(pos)\, p(\frac{e_l}{pos})}{P(e_l)} \quad (5.3)$$

In equation 5.3 $P(pos)$ is the posterior independent probability of the particular pos in the given corpus. $p(\frac{e_l}{pos})$ is the posterior probability of the final morphological component given the particular pos tag calculated from the tagged corpus. $P(e_l)$ is the posterior independent probability of final morphological component calculated from the tagged corpus. This probability is calculated using equation 5.4:

$$P(e_l) = \sum_{i=1}^{k} p(pos_i)\, p(\frac{e_l}{pos_i}) \quad (5.4)$$

$P(e_l)$ is calculated as a summation of all possible pos type k. $P(e_l)$, the product of $p(pos_i)$, the posterior probability of the given pos type *i* in the corpus and $p(\frac{e_l}{pos_i})$, the posterior probability of the final morpheme component given the pos tag *i*.

In case the word whose pos tag is to be determined consists of more than one morphological component, then the language model is designed by considering three factors: the lexical category of the stem and the pre-final and final morphological component and is given by equation 5.5:

$$P(\frac{pos}{root\_type, e_{l-1}, e_l}) = \alpha_1 p(\frac{pos}{root\_type}) + \alpha_2 p(\frac{pos}{e_{l-1}}) + \alpha_3 p(\frac{pos}{e_l}) \quad (5.5)$$

where $P(\frac{pos}{root\_type, e_{l-1}, e_l})$ gives the probability of the word being tagged with the particular pos tag given a particular stem type and a particular morphological pre-final and final components e_{l-1} and e_l. The three factors used for this probability calculation are $p(\frac{pos}{root\_type})$, the prior probability of a particular pos tag given a particular stem type and $p(\frac{pos}{e_l})$, the prior probability of a particular pos tag given a particular morphological component e_l. These two factors are similar to the equations 5.1 and 5.2. The second factor $p(\frac{pos}{e_{l-1}})$ is the prior probability of a particular pos tag given a particular morphological component e_{l-1}. In addition, α_1, α_2 and α_3 are contribution factors and $\alpha_1 + \alpha_2 + \alpha_3 = 1$. In order to calculate the prior probability $p(\frac{pos}{e_{l-1}})$ we use the equation:

$$P(\frac{pos}{e_{l-1}}) = \frac{P(pos)\, p(\frac{e_{l-1}}{pos})}{P(e_{l-1})} \quad (5.6)$$

In equation 5.6 $P(pos)$ is the posterior independent probability of the particular pos in the given corpus. $p(\frac{e_{l-1}}{pos})$ is the posterior probability of the final morphological component given the particular pos tag calculated from the tagged corpus. $P(e_{l-1})$ is the posterior independent probability of final morphological component calculated from the tagged corpus. This probability is calculated using equation 5.7.

$$P(e_{l-1}) = \sum_{i=1}^{k} p(pos_i)\, p(\frac{e_{l-1}}{pos_i}) \quad (5.7)$$

$P(e_{l-1})$ is calculated as a summation of all possible pos type *k*. $P(e_{l-1})$, the product of $p(pos_i)$, the posterior probability of the given pos type *i* in the corpus and $p(\frac{e_{l-1}}{pos_i})$, the posterior probability of the final morpheme component given the pos tag *i*. The next section describes the calculation of contribution factors.

VI. Estimation of Contribution Factors

Using generalized iterative scaling technique, the contribution factors α_1, α_2 and α_3 are calculated so as to maximize the likelihood of the rest of the corpus. Coarse and fine steps are used for finding the parameters. For the fixed

value of α_1, the $\alpha 2$ values are raised and simultaneously α_3 values are decreased, the correctness is showing the characteristic of inverted parabola in first quadrant of the graph. This inherent property is used for designing the following algorithm. Here, we are estimating the parameters in two steps, namely, coarse and fine steps. At coarse step, the value of increment and decrement is in the quantity of 0.1 for estimating coarse optimal value of the parameters. At fine step, the changes are in the quantity of .01 for estimating fine optimal value of parameters.

Algorithm for Estimating α_1, α_2 and α_3

Constraint $\alpha_1 + \alpha_2 + \alpha_3 = 1$
Max =0.0
Optimal _solution = set(0 , 0 , 1)
Coarse step
Initialize α_1=0.0
Step in increment α_1 by 0.1
Repeat the following step until α_1<=1.0
Initialize val =0.0; and α_2=0.0;
Step in increment α_2 by 0.1
Repeat the following step until α_2<=1.0- α_1;
set α_3=1.0-(α_1+α_2);
if the val is less than correctness (α_1 , α_2 , α_3)
assign the val = correctness (α_1 , α_2 , α_3)
else break the inner loop;
Loop
if the value of val greater than max
Max = val
Optimal _solution = set (α_1, α_2, α_3)
Loop
Fine Step
Initialize α_1= optimal(α_1);
Step in increment α_1 by 0.01
Repeat the following step until α_1<= optimal(α_1) +0.09
Assign val =0.0; and α_2= optimal(α_2);
Step in increment α_2 by 0.01
Repeat the following step until α_2<= optimal(α_2) +0.09- α_1
Assign α_3= 1.0-(α_1+α_2);
If the val is less than correctness (α_1 , α_2 , α_3))
then
val = correctness (α_1 , α_2 , α_3)
else
Break inner loop;
Loop
if the value of Max less than Val
Val = Max
Optimal _solution = set (α_1, α_2, α_3)
Loop
(α_1 , α_2 , α_3) ← value(Optimal _solution)

In this algorithm, correctness function will return the percentage of words correctly tagged by the language model with the corresponding contribution factors α_1 , α_2 and α_3. Optimal_solution is a set variable to store set of values of α_1 , α_2 and α_3. The function *optimal* with a variable as the parameter obtains the value of the corresponding parameter in the set *Optimal solution*.

Parameter α_1 , α_2 and α_3 for Language Model 1

TABLE II.
CONTRIBUTION FACTORS α1, α2, α3 FOR FINE STEP

α1	α2	α3	Correct (%)
0.30	0.21	0.49	86.67
0.31	0.21	0.48	86.64
0.32	0.22	0.46	86.82
0.33	0.23	0.44	86.55

After coarse step we got the values (0.3, 0.2, 0.5), thus, the estimated values of α1 = 0.31, α2 =0.21 and α3 =0.48

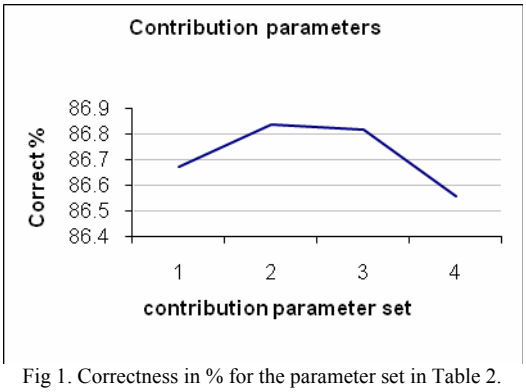

Fig 1. Correctness in % for the parameter set in Table 2.

Fig. 1 shows the graphical representation of Correctness for each set of parameter values as mentioned in Table II.

Parameter α_1 and α_2 for Language Model 2

TABLE III.
CONTRIBUTION FACTOR α1, α2 FOR COARSE STEP

	α1	α2	Correct (%)
1.	0.0	1.0	84.60
2.	0.4	0.6	93.06
3.	0.5	0.5	96.88
4.	0.6	0.4	98.37
5.	0.7	0.3	98.38
6.	0.8	0.2	99.37
7.	0.9	0.1	93.32
8.	1.0	0.0	90.00

There is no further improvement at fine step, so the estimated values are α1 = 0.8 and α2 =0.2 .

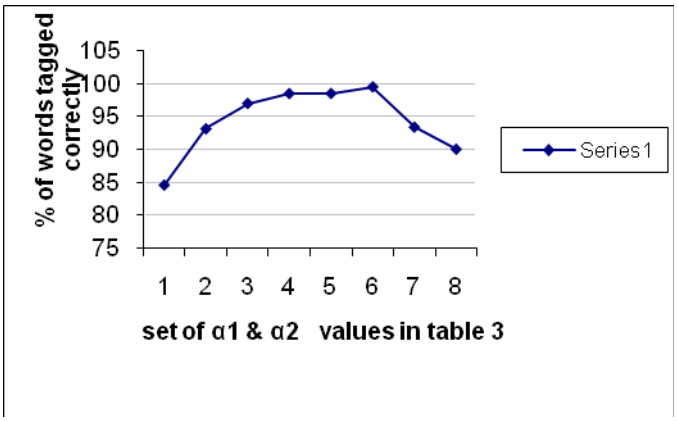

Fig. 2. Correctness in % for the parameter set from Table 3.

Fig. 2 shows the graphical representation of Correctness for each set of parameter values as mentioned in Table III.

VII. Evaluation and Results

The implemented system is evaluated as in Information Retrieval, which makes frequent use of precision and recall.

Precision is defined as a measure of the proportion of the selected items that the system got right.

$$precision = \frac{No\ .of\ items\ tagged\ correctly}{No\ .of\ items\ tagged} \quad (6.1)$$

Recall is defined as the proportion of the target items that the system selected.

$$\text{Re}\,call = \frac{No.of\ items\ tagged\ by\ the\ system}{No.of\ items\ to\ be\ tagged} \quad (6.2)$$

To combine precision and recall into a single measure of over all performance, the F-measure is defined as

$$F = \frac{1}{\alpha \frac{1}{p} + (1-\alpha)\frac{1}{R}} \quad (6.3)$$

where P is precision, R is recall and α is factor, which determine the weighting of precision and recall. A value of α is often chosen for equal weighting of precision and recall. With this α value the F-measure simplifies to

$$F = \frac{2PR}{P+R} \quad (6.4)$$

The perplexity is a useful metric for how well the language model matches with a test corpus. The perplexity is a variant of entropy. The entropy is measured by the following equation

$$H(tag\_type) = -\frac{1}{n}\sum_{i=1}^{n} \log(p(tag(w_i))) \quad (6.5)$$

$$perplexity(tag\_type) = 2^{H(tag\_type)} \quad (6.6)$$

The system is evaluated with a test corpus with 43,678 words in which 36,128 words are morphologically analyzed within our tag set. 6,123 words are named entity, while the remaining words are unidentified. The morphologically analyzed words are passed into tagger designed using our language model. The following table and graphs represents the results obtained by the language models for determining the tags.

TABLE IV.
Noun Categories

Postag	Recall	Precision	F-measure	Perplexity
<N>	0.98	0.99	0.99	1.91
<Pn>	0.98	0.98	0.98	2.61
<IN>	1.00	1.00	1.00	1.63
<NN>	0.48	0.97	0.64	8.58
<NP>	0.99	0.87	0.93	3.29
<PnP>	0.81	1.00	0.89	1.91
<INP>	0.69	0.47	0.56	9.18
<VnP>	0.01	1.00	0.02	1.57
<PN>	0.00	0.00	0.00	-
<NNP>	0.15	0.97	0.27	1.87
<NnP>	0.60	0.71	0.65	3.95
<PNP>	0.00	0.00	0.00	-
<Vn>	0.81	0.96	0.88	1.63
<Nn>	0.18	1.00	0.33	3.63

Table IV shows noun category type POS tags in which the tags <VNP>, <PN> and <PNP> are the noun category but its stem is of type verb. Due to this reason, the words of these tag types are wrongly tagged. These cases can be rectified by transformation based learning rules.

TABLE V.
Verb Categories

Postag	Recall	Precision	F-measure	Perplexity
<V>	0.99	0.93	0.95	3.88
<AV/V>	1.00	1.00	1.00	1.00
<FV>	1.00	0.99	0.99	1.00
<NFV>	1.00	0.90	0.95	1.00
<NFV/DT>	0.00	0.00	0.00	-
<Vvp>	0.93	0.92	0.92	2.74
<VP>	0.81	0.87	0.84	3.66
<Vinf>	0.71	0.88	0.79	3.98
<V/Vrp>	0.99	1.00	0.99	19.40
<Vrp>	0.98	0.78	0.87	1.81
<V/Vinf>	0.99	0.97	0.98	1.13
<VC>	0.42	0.91	0.57	3.47
<Vpost>	0.00	0.00	0.00	-

Table V shows verb category type POS tags. A word '*anRu*' in Tamil belongs to <NFV/DT> tag type. It has two type of context with the meaning of (i) *those day* (DT) and (ii) *not* (NFV). These cases have to be rectified by word sense

disambiguation rules. Words of Tag type <Vpost> are relatively very few. By considering further morpheme components, these tag types can be identified.

TABLE VI.
OTHER CATEGORIES

Postag	Recall	Precision	F-measure	Perplexity
<DT>	1.00	1.00	1.00	1.00
<cNum>	1.00	1.00	1.00	1.00
<Madj>	1.00	0.99	0.99	1.00
<adj>	0.98	0.99	0.99	3.57
<Dadj>	0.99	1.00	0.99	1.00
<Iadj>	1.00	0.97	0.99	1.00
<PO>	1.00	1.00	1.00	1.00
<conj>	0.96	1.00	0.98	1.00
<par>	1.00	1.00	1.00	1.02
<Int>	1.00	1.00	1.00	1.00
<adv>	0.92	0.97	0.94	3.80
<SP>	1.00	1.00	1.00	9.22
<postAdj>	1.00	0.86	0.92	9.63
<SCC>	1.00	1.00	1.00	1.00
<PostP>	0.97	1.00	0.99	9.07

Table VI shows POS tags of other category types. The occurrence of categories of this type is less as compared with noun type and verb type.

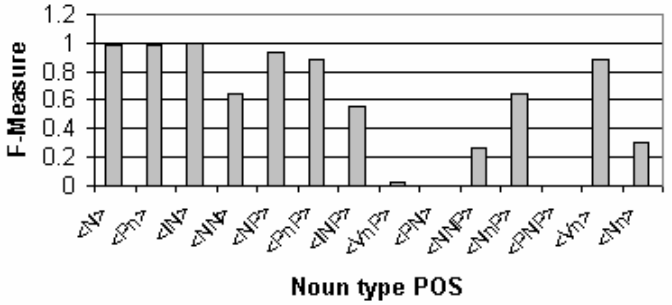

Fig 3. F-measure for POS tag of noun categories.

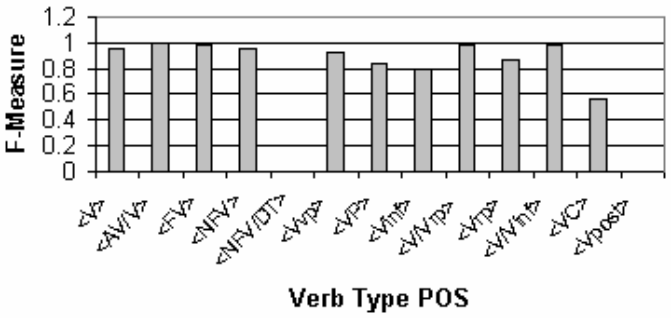

Fig 4. F-measure for POS tag of verb categories.

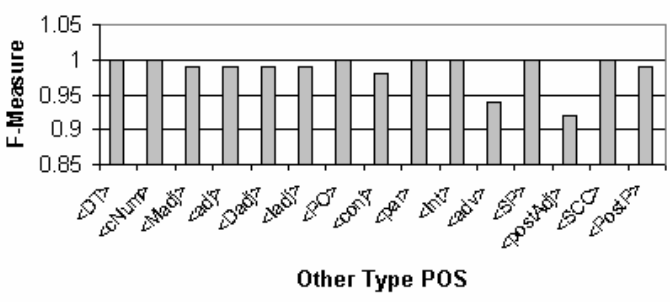

Fig 5. F-measure for POS tag of other categories.

The same test is repeated for another corpus with 62,765 words, in which the used analyzer identifies 52,889 words. 49,340 words are correctly tagged using the suggested language model. These two tests are tabulated in Table VII.

TABLE VII.
OVERALL ACCURACY AND PERPLEXITY.

	Words	Correct	Error	Accuracy (%)	Perplexity
T1	36,128	34,656	1,472	95.92	153.80
T2	36,889	34,840	2,049	94.45	153.96

As the perplexity values for both test cases are more or less equal, we can conclude that the formulated language model is independent of the type of untagged data.

VIII. CONCLUSION AND FUTURE WORK

We have presented a part-of-speech tagger for Tamil, which uses a specialized language model. The input of a tagger is a string of words and a specified tag set similar to the described one. The output is a single best tag for each word. The overall accuracy of this tagger is 95.92%. This system can be improved by adding a module with transformation based learning technique and word sense disambiguation. The same approach can be applied for any morphologically rich natural language. The morpheme based language model approach can be modified for chunking, named entity recognition and shallow parsing.

REFERENCES

[1] Aniket Dalal, Kumar Nagaraj, Uma Sawant, Sandeep Shelke , Hindi Part-of-Speech Tagging and Chunking : A Maximum Entropy Approach. In: Proceedings of the NLPAI Machine Learning Contest 2006 NLPAI, 2006.

[2] Nizar Habash , Owen Rambow ,Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in one Fell Swoop. In: Proceedings of the 43rd Annual Meeting of the ACL, pages 573–580, Association for Computational Linguistics, June 2005.

[3] D. Hiemstra. Using language models for information retrieval. PhD Thesis, University of Twente, 2001.

[4] S. Armstrong, G. Robert, and P. Bouillon. Building a Language Model for POS Tagging (unpublished), 1996. http://citeseer.ist.psu.edu/armstrong96building.html

[5] P. Anandan, K. Saravanan, Ranjani Parthasarathi and T. V. Geetha. Morphological Analyzer for Tamil. In: International Conference on Natural language Processing, 2002.

[6] Thomas Lehman. A grammar of modern Tamil, Pondicherry Institute of Linguistic and culture.

[7] Sandipan Dandapat, Sudeshna Sarkar and Anupam Basu. A Hybrid Model for Part-of-speech tagging and its application to Bengali. In: Transaction on Engineering, Computing and Technology VI December 2004.

[8] Barbara B. Greene and Gerald M. Rubin. Automated grammatical tagger of English. Department of Linguistics, Brown University, 1971.
[9] S. Klein and R. Simmons. A computational approach to grammatical coding of English words. JACM, 10:334-337, 1963.
[10] Theologos Athanaselies, Stelios Bakamidis and Ioannis Dologlou. Word reordering based on Statistical Language Model. In: Transaction Engineering, Computing and Technology, v. 12, March 2006.
[11] Sankaran Baskaran. Hindi POS tagging and Chunking. In: Proceedings of the NLPAI Machine Learning Contest, 2006.
[12] Lluís Márquez and Lluis Padró. A flexible pos tagger using an automatically acquired Language model. In: Proceedings of ACL/EACL'97.
[13] K. Rajan. Corpus analysis and tagging for Tamil. In: Proceeding of symposium on Translation support system STRANS-2002
[14] T. Brants. TnT - A Statistical Part-of-Speech Tagger. User manual, 2000.
[15] Scott M. Thede and Mary P. Harper. A second-order Hidden Markov Model for part-of-speech tagging. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, pages 175—182. Association for Computational Linguistics, June 20--26, 1999.
[16] Eric Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. Computation Linguistics, 21(4):543- 565, 1995.