TNT TAGGER FOR MALAYALAM WITH FUZZY RULE BASED LEARNING

Alen Jacob*[1], Amal Babu*[2], Rajeev R. R.[3] and P. C. Reghu Raj[4]

*[1] M. Tech, Computational Linguistics, Government Engineering College, Sreekrishnapuram, Palakkad, Kerala, India
alenjacob@outlook.com[1]
*[2] M. Tech, Computational Linguistics, Government Engineering College, Sreekrishnapuram, Palakkad, Kerala, India
amalbabuputhanpurayil@gmail.com[2]
[3] Research Officer –VRCLC, IIITM-K, Thiruvananthapuram, Kerala, India
rajeev@iiitmk.ac.in [3]
[4] Dept. of Computer Science and Engineering, Government Engineering College, Sreekrishnapuram, Palakkad, Kerala, India
pcreghu@gmail.com [4]

Abstract: TnT is an efficient statistical Parts-of-speech (POS) Tagger based on Hidden Markov Model. TnT performs well on known word sequences. But, the performance degrades with increase in the number of unknown words. In this paper, we propose a method to overcome this performance degradation using fuzzy rules. Fuzzy rule based model is designed to provide TnT with sufficient information about the tag of unknown words without degrading the performance of TnT. On processing an unknown word from the input, the TnT tagger relies on the probability distribution of words having the same suffix within the training corpus. In Indian languages like Malayalam, the POS tag of an unknown word depends not only on suffix. Due to high inflectional and free order nature, the dependency is rather complex than the one captured by suffix tag distribution probabilities. When TnT with fuzzy rule based learning encounters an unknown word, the TnT generates a set of possible tags for the given word based on the fuzzy rules matched by the word. If the word does not match any fuzzy rule then the model depends upon the probability distribution of the suffix. This approach guarantees that the performance of TnT will only be improved from its normal performance.

INTRODUCTION

Parts-of-speech (POS) tagging is the process of assigning grammatical categories to words with similar behavior. When the training corpus becomes small, the performance of TnT depends on the number of unknown words in the test document (Andrei Mikheev, 1997). A word is said to be un- known if it is not present in the training corpus, and hence not present in the lexicon. In this paper we presents TnT with fuzzy rule based learning to overcome the problems created by unknown words. In this model the TnT performs exactly the same way as an ordinary TnT till it encounters an unknown word. Rather than relying on the probability distribution of suffix of the unknown word, TnT make use of a set of fuzzy rules to predict the possible tags for the word. These fuzzy rules are learned from the training corpus. Rules provide a way to memorize the behavior of words in a given context. In our model, rules are converted to fuzzy rules for avoid over fitting.

Related work in this field are discussed in the following section.

RELATED WORKS

Fuzzy logic has not received much attention of Natural Language Processing research communities. A critical survey conducted by Joao P. Carvalho et al. (2012) shows that there is only a handful of papers that somehow uses fuzzy logic and techniques, as a scientific approach for solving a problem. Fuzzy network model is used by Kim and Chang Kim (1995) for POS tagging under small training data. In their work, Artificial Neural Network is used for learning the Fuzzy Con- textual Membership function. The unknown word problem is not pursued separately in this method. In Torsten Brants (2000) TnT tagger, the unknown words are tagged relying on the probability distribution of words with same suffix.

Cucerzan and D. Yarowsky (2000) presents a minimally supervised learning approach for unknown word tag prediction that uses paradigmatic similarity measure learned from large training data. Andrei Mikheev (1997) in his paper makes use of rule based approach for predicting the tag of unknown words. In his work, the tag prediction makes use of the starting and ending segment of the unknown word. He makes use of a general purpose lexicon and word frequencies de- rived from raw data to train the model. Orphanos and D. Christodoulakis (1999) makes use of decision tree based approach for solving unknown word problem for highly inflectional language.

TNT ARCHITECTURE

The Underlying Model (Torsten Brants, 2000)

TnT uses second order Markov models for part-of-speech tagging. The states of the model represent tags, outputs represent the words. Transition probabilities depend on the states, thus pairs of tags. Output probabilities depend only on the most recent category. To be explicit, we calculate:

$$\underset{t_1 \dots t_T}{arg\max} \left[\prod_{i=1}^{T} P(t_i|t_{i-1}, t_{i-2}) P(w_i|t_i) \right] P(t_{T+1}|t_T) \tag{1}$$

For a given sequence of words $w_1 \dots w_T$ of length T. $t_l \dots t_T$ are elements of the tagset, the additional tags t_{-1}, t_0, and t_{T+1} are beginning-of-sequence and end-of-sequence markers. Using these additional tags, even if they stem from rudimentary processing of punctuation marks, slightly improves tagging results.

TnT make use of the probability distribution of words having the same suffix to predict the tag of an unknown word. The term suffix as used here means "*final sequence of characters of a word*" which is not necessarily a linguistically meaningful suffix.

Issues associated with TnT while processing Malayalam documents are discussed in the following section.

Issues of TnT with Malayalam

Malayalam belongs to the Dravidian family of languages and shows very high agglutinative and free order nature. The context of a given word may not be completely defined by a trigram as in a TnT for other languages. Due to its agglutinative nature, a suffix of finite length might also fail for most of the unknown word tag prediction.

Tagging a given Malayalam document using a TnT would provide good efficiency, when the document consist only of known words in known context. When it comes to tagging of an unknown word in Malayalam, TnT depends solely on the probability distribution of words having same suffix. This suffix may not always be a valid suffix especially for highly inflected language like Malayalam. TnT then assumes that the unknown words are observed from a finite set of states, with observation probability found using the suffix probability distribution. As described, these finite states are selected based on incomplete and language independent information.

In the following section, we introduce a Fuzzy rule based model for predicting possible states or tag for a given unknown word in Malayalam. These predictions are governed by language dependent features learned by the machine. Section 5 deals with the integration of fuzzy rule based model with basic TnT architecture.

THE FUZZY RULE BASED MODEL

The Model

In fuzzy rule based model, we map the problem of POS tagging of unknown words to an instance of classification problem. The unknown words are taken as the input or observation that is to be classified into a predefined set of classes or tags. For implementing such a system, using fuzzy rules, we follow the approach described by [2]. The model consists of following steps:

Step 1 : Generates fuzzy rules from the data;
Step 2 : Transforms the input to feature vector;
Step 3 : Identifies the applicable fuzzy rules.

Step 1 : Generating Fuzzy Rules

A fuzzy rule help us to model the way in which a person would classify an unknown word. i.e., if a person is given an input word to classify, he identifies the affinity or similarity of the input to various classes. He classifies the input to a class that have strong similarity to it. When a human is classifying, it is not likely that he will follow uniform standards throughout the process.

In our study we made use of the simple triangular membership function. The fuzzy regions selected were labeled as *LOW₁*, *LOW₂*, *MEDIUM*, *STRONG₁*, and *STRONG₂*. These regions would model the human preference level towards a given tag. Our model make use of n-gram probabilities for finding the fuzzy membership value. These n-gram probabilities are calculated in the same way as in TnT [4].

Corpus

Malayalam corpus containing 12,870 tagged words from tourism and sports domain is used for training. The words are tagged based on BIS tagset. The n-gram probabilities and the word probabilities are estimated from this tagged corpus. As a first step, we use the maximum likelihood probabilities $\hat{P}$ which are derived from the relative frequencies f for all t_1, t_2, t_3 in the tagset and w_3 in the lexicon.

Unigrams: $$\hat{P}(t_3) = \frac{f(t_3)}{N} \qquad (2)$$

Bigrams: $$\hat{P}(t_3 | t_2) = \frac{f(t_2, t_3)}{f(t_2)} \qquad (3)$$

Trigrams: $$\hat{P}(t_3 | t_1, t_2) = \frac{f(t_1, t_2, t_3)}{f(t_1, t_2)} \qquad (4)$$

Lexical: $$\hat{P}(w_3 | t_3) = \frac{f(w_3, t_3)}{f(t_3)} \qquad (5)$$

N is the total number of tokens in the training corpus. As a second step, n-gram frequencies are smoothen and lexical frequencies are completed by handling words that are not in the lexicon (unknown words). Smoothing is performed to overcome sparse data problem. It is described in the following section.

Smoothing

Trigram probabilities generated from a corpus usually cannot directly be used because of the sparse data problem. This means that there are not enough instances for each trigram to reliably estimate the probability.

Linear interpolation is the smoothing technique used. Equation (6) shows the estimation of trigram probabilities.

$$P(t_3 | t_1, t_2) = \lambda_1 \hat{P}(t_3) + \lambda_2 \hat{P}(t_3 | t_2) + \lambda_3 \hat{P}(t_3 | t_1, t_2)$$

$\hat{P}$s are maximum likelihood estimates of the probabilities, and $\lambda_1 + \lambda_2 + \lambda_3 = 1$, so P again represent probability distribution.The values of λ_1 , λ_2 and λ_3 are estimated by deleted interpolation.

The learned probabilities are used to construct fuzzy rules. This is described in the following subsection.

Fuzzy Rule Generation

After training the probability model, we created a new training corpus, for fuzzy rule generation, consisting of words with its features. The features are to be carefully selected so as to better describe the word context. The features selected in this study are: tags of previously occurring words, the length of the word and its suffix. After creating the corpus most probable three tags for each words are predicted using the probability model. The fuzzy model then forms fuzzy rules of the following format:

IF: T_1 is B_1 and T2 is B_2 and T_3 is B_3 and suffix is S_1,
THEN: predict tag as C,

where, T_i's are all tags and B_i's are the fuzzy regions identified for each such tags. i.e., if the unknown word has a probability P_1 of being tagged with tag T_1, which belongs to one of the predefined fuzzy regions having membership values (μ_1 and μ_2) in at most two class. Then, the rule component identified is as, T_1 belongs to fuzzy class with maximum membership. For example, The rule component becomes: IF T_1 is Class $(\max(\mu_1, \mu_2))$. Three such components are built from most probable three class, to which the unknown word could be assigned.

From the training corpus, such fuzzy rules could be easily calculated by a machine. Such rules are combined together to form a fuzzy rule base. Such a rule base will have conflicting rules. The conflict between the rules are resolved by using, the degree of activation for a given rule as proposed by Johannes et.al. The degree of activation of the i^{th} rule is calculated as,

$$\beta_i\left(\bar{x}\right)=\prod_{j=1}^{n}\mu A_{ij}\left(x_j\right), \quad i=1,2,\ldots,M$$

where M is the size of the rule base and A_{ij} is the j^{th} labeled class of i^{th} rule. x_j denotes the j^{th} class label in the i^{th} rule.

Step 2 : Vectorization

Once the rules are learned, we are left with the task of converting the word into some feature vector representation. This representation would help us to select applicable fuzzy rules. Our model selects the tags of previous words and the suffix information as features. One could select any features as per the language. The previous tag information can be easily calculated from the intermediate output produced by a TnT. The suffix of the word can be extracted using a simple word splitter designed for Malayalam. We used the unicode chart for Malayalam for designing the word splitter. The word splitter takes as input a Malayalam word and returns a set consisting of all the alphabets in the word. Figure 1 shows the output of the Malayalam word splitter.

ആഫ്രിക്കൻ ⇒ ആ ഫ്രി ക്ക ൻ

Figure 1: The output of the Malayalam word splitter.

Information from a morphological analyzer at this stage would greatly improve the performance of the system. Any such features could be added inorder to improve the performance. Note that the complexity of such a rule using more features results in an exponential growth in running time. This is discussed further in the following section.

Step 3 : Rule Selection

Large number of features would increase the running time of the system. Features up to six would provide acceptable performance [3]. Classical fuzzy reasoning, also known as the *winner rule*(8) is used for selecting appropriate fuzzy rules.

$$C_{compat}\left(R_k,\bar{x}\right)=t\left(A_1\left(g_{p1}\right),A_2\left(g_{p2}\right),\ldots,A_l\left(g_{pl}\right)\right)$$

where $\vec{x}$ is the feature vector and $A_l(g_{pl})$ is the membership value corresponding to the fuzzy region labeled g_{pl}. Find the rule that has maximum compatibility value C_{compat}. Provide as output the class predicted by this rule.

The following section describes how the fuzzy rule based model can be integrated into the basic TnT architecture.

INTEGRATION

In Malayalam, due to its high inflectional nature, it is more often to come across an unknown word. For TnT every word is a sequence of characters. For two words to be similar, their character sequence should match exactly. Hence two different inflections of a given word-root are considered as two entirely different words. In a given Malayalam document, it is common to find more than one inflections for a given word in its discourse. In a training corpus, all the inflections of a given word are not expected to occur. Hence a known word, with previously not encountered inflection is considered to be unknown. Hence TnT would come across unknown words more often in Malayalam than in any other language with comparatively less inflections.

TnT strategy on encountering an unknown word is to initiate a certain set of procedures to best approximate the observation probabilities of the unknown word. TnT considers a word as unknown if there is no observation probability associated with that word from any of the states in the HMM, where each state corresponds to a pair of POS tags. The fuzzy model helps TnT to best approximate this observation probability compared to the built in strategy of TnT using suffix probability distribution alone.

The fuzzy rule based model is designed to predict the possible tags of an unknown word by taking as input the context in which the word occur and the suffix that the word takes. TnT on intercepting an unknown word, during its processing, would initiate the fuzzy rule based model by supplying contextual and suffix information. The fuzzy model predicts the tag of the word from previously encountered behaviors of words occurring in similar context with the provided suffix. Here the decision not only depends upon the suffix probability but also on the sequence of most probable tags.

In Malayalam it is not difficult for one to find examples, where words with same suffix have different POS tags. In TnT, we are interested in the most probable tag, a word ending with a given suffix takes. In fuzzy model we first makes a prediction of this manner, but the decision is not biased towards the most probable tag alone. But also to the sequence of most probable tags and the behaviors of words in the training corpus during such situations. ie, in its simplest form, if the word has suffix `s' and most probable predictions are `N',`PP' and `V', we consults the fuzzy rules for selecting one tag from them. Fuzzy rules records the observations made from the training corpus. If we have observed such a situation in the training corpus and at that time the selection of `PP' was appropriate, then we would have a rule that tells us to tag the word as `PP'. In our model the prediction rule not only depends on the tag patter (ie, `N, `PP' and `V') but also on the probability of these prediction.

Fuzzy model is not designed to predict the POS tag when the context of the word is not encountered previously in the training set. Hence, there might be situations where no rules in the rule base are matched. Here we make use of the free order nature of Malayalam. The model rearranges the tags of previous words (ie, rearranges the context words). This would result in a new input vector. Model then generates the predicted sequence and their probability values, for this new input vector. Fuzzy rule base is again searched for a possible match. The justification is that, in a training corpus we would have observed such an instance but not necessarily in the same order as we saw in the test document, this is due to the free order nature of Malayalam. Still if we are left with no match then, the fuzzy model would produce the same output as that of a TnT without a fuzzy rule based learning.

The fuzzy rule based model might providing more than one predictions, if each of the predictions are equally likely. In such situation, it is up to the TnT to predict the actual tag. TnT has a built in mechanism defined by the viterbi algorithm to handle such situations.

EXPERIMENTAL RESULTS AND ANALYSIS

The training corpus consist of 12870 words, using which three other corpora are generated, each with size 4052, 8046 and 12870 respectively. The evaluation is performed using 5-fold cross validation. Each of the three newly created training corpus is divided into five folds of almost equal size. The five-fold cross validation is then performed in three phases (one phase for each corpus). At each phase the model is trained with 4 folds of the training data and the remaining one fold is used to test and evaluate the model.

Table I. The 5-Fold cross validation result of TnT over the corpus of size 4052(number of words)

Corpus Size (No. of Words)	***Fold***	***Fold Size (No. of words)***	***TnT***				
			Words wrongly tagged	***Words correctly tagged***	***Unknown words***	***Unknown words wrongly tagged***	***Error to unknown words (%)***
4052	1	812	187	625	385	145	77.54
	2	806	164	642	336	125	76.22
	3	814	215	599	420	167	77.67
	4	811	186	625	308	146	78.50
	5	809	187	622	272	153	81.81
						Avg	78.35

Table II. The 5-Fold cross validation result of Fuzzy rule based model over the corpus of size 4052(number of words).

Fuzzy Model		
Total Input Size	***Correctly Tagged***	***Accuracy (0)***
145	54	6.6502
125	65	8.0645
167	74	9.0909
146	60	7.3983
153	53	6.5513
	Avg	7.5510

Table 1 and 2 show the 5-fold cross validation result of TnT and fuzzy rule based system. For tagging words belonging to fold 1 the model is trained using folds two to five, i.e., *fold 2, fold 3, fold 4* and *fold 5*. Among 812 words in the fold 1 only 625 words were correctly tagged. TnT failed to tag 187 words correctly of which 145 words were unknown. These 145 words were given as input to the fuzzy rule based model, among them 54 words were correctly tagged by the fuzzy model. This is justifiable since the TnT calls the fuzzy model only for unknown words. The performance of TnT can hence be improved by 6.65%. It is observed that, the fuzzy model improves the performance of TnT by an average of 7.55% with a standard deviation of 0.95%, for a corpus of size around 4000. We carried out the experiment over training corpora of different sizes. Figure 2 shows the performance improvement variation of TnT with fuzzy rule based model over corpora of different sizes. As the size of the training corpus increases the performance improvement decreases.

Comparison

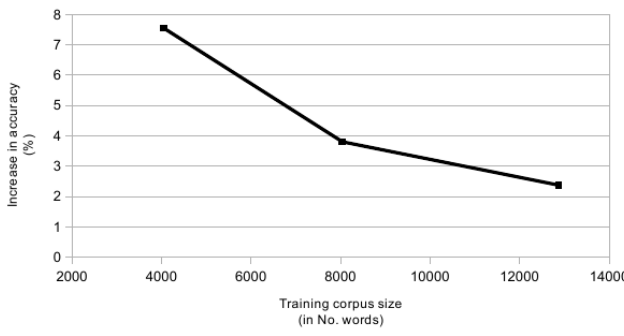

Figure 2: The 5-Fold cross validation result of TnT accuracy improvement, carried over corpora of different sizes

[9] using Decision tree base approach for predicting the unknown word tag, acquired an error rate of 16% for Greek language. They used a corpus of size 7624 (sentences), created from students writing literature, newspaper, technical papers, magazines etc. [8] used a rule based guising for predicting the tag of unknown word. He made use of three types of rules, prefix morphological rules, suffix morphological rules and ending-guising rules. Tagging accuracy obtained by cascading these guesser was 87.75 - 88.7%. This provided an overall accuracy of 93.6 - 94.3% in tagging, which is a 6\% improvement over the model without unknown word guising. [7] using their method acquired a 27% reduction in error rate for French and 12% for English. This resulted in a 7.8% increase in accuracy for the tagger for French and 7.6% improvement for English.

CONCLUSION AND FUTURE WORK

We proposed a fuzzy rule based model to improve the basic TnT performance by correctly predicting the tag of unknown words. The model performs well when the training data size is small.

Pursuing new features such as the tag of the following word, might improve the model. The use of Artificial Neural Network for approximating the fuzzy membership function is suggested as an extension.

REFERENCES

[1] Andrei Mikheev, “Automatic Rule Induction for Unknown-Word Guessing”, Computational Linguistics, 23(3), pages 405-423, 1997

[2] Cucerzan and D. Yarowsky, “Language Independent, Minimally Supervised Induction of Lexical Probabilities”, Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics(ACL-2000),pages 270-277, 2000.

[3] Daniel Jurafsky and James H. Martin, “Speech and Language Processing- an introduction to Natural Language Processing”, Computational Linguistics and Speech Recognition. Prentice Hall, 2014.

[4] Jae-Hoon Kim, Gil Chang Kim,“ Fuzzy Network Model for Part-of-Speech Tagging under Small Training Data”, Natural Language Engineering, Cambridge University Press, 1995.

[5] Joao P. Carvalho ,Fernando Batista, Luisa Coheur, “A Critical Survey on the use of Fuzzy Sets in Speech and Natural Language Processing”, WCCI 2012 IEEE World Congress on Computational Intelligence Brisbane, Australia, 2012.

[6] Li-Xin Wang and Jerry M. Mendel, “Generating fuzzy rules by learning from examples”, IEEE Transl. J. Magn. Japan, vol. 2, pp. 740-741, 1987.

[7] Orphanos and D. Christodoulakis, “POS Disambiguation and Unknown Word Guessing with Decision Trees”, In Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics(EACL-99), pages 134-141, 1999.

[8] Tatiane M. Nogueira, Heloisa A. Camargo and Solange O. Rezende, “Fuzzy rules for document classification to improve information retrieval”, International Journal for Computer Information Systems and Industrial Management Applications, Volume 3 pp. 210-217, 2011.

[9] Torsten Brants, “TnT- A statistical parts of speech tagger”, Association for Computational Linguistics, Proceedings of the sixth conference on Applied natural language processing, pages 224-231, 2000.