Hybrid Neuro and Rule-Based Part of Speech Taggers

Qing Ma, Masaki Murata, Kiyotaka Uchimoto, Hitoshi Isahara
Communications Research Laboratory
Ministry of Posts and Telecommunications
588-2, Iwaoka, Nishi-ku, Kobe 651-2492, Japan
{qma, murata, uchimoto, isahara}@crl.go.jp

Abstract

A hybrid system for tagging part of speech is described that consists of a neuro tagger and a rule-based corrector. The neuro tagger is an initial-state annotator that uses different lengths of contexts based on longest context priority. Its inputs are weighted by information gains that are obtained by information maximization. The rule-based corrector is constructed by a set of transformation rules to make up for the shortcomings of the neuro tagger. Computer experiments show that almost 20% of the errors made by the neuro tagger are corrected by these transformation rules, so that the hybrid system can reach an accuracy of 95.5% counting only the ambiguous words and 99.1% counting all words when a small Thai corpus with 22,311 ambiguous words is used for training. This accuracy is far higher than that using an HMM and is also higher than that using a rule-based model.

1 Introduction

Many part of speech (POS) taggers proposed so far (e.g., Brill, 1994; Merialdo, 1994; Daelemans, et al., 1996; and Schmid, 1994) have achieved a high accuracy partly because a very large amount of data was used to train them (e.g., on the order of 1,000,000 words for English). For many other languages (e.g., Thai, which we treat in this paper), however, it is not as easy to create large corpora from which large amounts of training data can be extracted. It is therefore desirable to construct a practical tagger that needs as little training data as possible. A multi-neuro tagger (Ma and Isahara, 1998) and its slimmed-down version called the elastic neuro tagger (Ma, et al., 1999), which have high generalizing ability and therefore are good at dealing with the problems of data sparseness, were proposed to satisfy this requirement. These taggers perform POS tagging using different lengths of contexts based on longest context priority, and each element of the input is weighted with information gains (Quinlan, 1993) for reflecting that the elements of the input have different relevances in tagging. They had a tagging accuracy of 94.4% (counting only the ambiguous words in part of speech) in computer experiments when a small Thai corpus with 22,311 ambiguous words was used for training. This accuracy is far higher than that using the hidden Markov model (HMM), the main approach to part of speech tagging, and is also higher than that using a rule-based model.

Neuro taggers, however, have several crucial shortcomings. First, even in the case where the POS of a word is uniquely determined by the word on its left, for example, a neural net will also try to perform tagging based on the complete context. As a result, even for when the word on the left is the same, the tagging results will be different if the complete contexts are different. That is, the neuro tagger can hardly acquire the rules with single inputs. Furthermore, although lexical information is very important in tagging, it is difficult for neural nets to use it because doing so would make the network enormous. That is, the neuro tagger cannot acquire the rules with lexical information. Additionally, because of convergence and over-training problems, it is impossible and also not advisable to train neural nets to an accuracy of 100%. The training should be stopped at an appropriate level of accuracy. Consequently, neural nets may not acquire some useful rules.

To make up for these shortcomings of the neuro tagger, we introduce in this paper a rule-based corrector as the post-processor and construct a hybrid system. The rule-based corrector is constructed by a set of transformation rules, which is acquired by transformation-based error-driven learning (Brill, 1994) from training corpus using a set of templates. The templates are designed to supply the rules that the neuro tagger can hardly acquire. Actually, by examining the transformation rules acquired in the computer experiments, the 99.9% of them are exactly those that the neuro tagger can hardly acquire, even when using a template set including those for generating the rules that the neuro tagger can easily acquire. This reinforces our expectation that the rule-based approach is a well-suited method to cope with the shortcomings of the neuro tagger. Computer experiments shows that about 20% of errors made by the neuro tagger can be corrected by using these rules and that the hybrid system can reach an accuracy of 95.5% counting only the ambiguous words and 99.1% counting all words in the testing corpus, when the same corpus described above is used for training.

2 POS Tagging Problems

In this paper, suppose there is a lexicon V, where the POSs that can be served by each word are listed, and there is a set of POSs, Γ. That is, unknown words that do not exist in the lexicon are not dealt with. The POS tagging problem is thus to find a string of POSs $T = \tau_1\tau_2\cdots\tau_s$ ($\tau_i \in \Gamma$, $i = 1, \cdots, s$) by following procedure φ when sentence $W = w_1w_2\cdots w_s$ ($w_i \in V$, $i = 1, \cdots, s$) is given.

$$\varphi : W^t \to \tau_t, \qquad (1)$$

where t is the index of the target word (the word to be tagged), and W^t is a word sequence with length $l + 1 + r$ centered on the target word:

$$W^t = w_{t-l}\cdots w_t \cdots w_{t+r}, \qquad (2)$$

where $t - l \geq 1$, $t + r \leq s$. Tagging can thus be regarded as a classification problem by replacing the POS with class and can therefore be handled by using neural nets.

3 Hybrid System

Our hybrid system (Fig. 1) consists of a neuro tagger, which is used as an initial-state annotator, and a rule-based corrector, which corrects the outputs of the neuro tagger. When a word sequence W^t [see Eq. (2)] is given, the neuro tagger output a tagging result $\tau_N(w_t)$ for the target word w_t at first. The rule-based corrector then corrects the output of the neuro tagger as a fine tuner and gives the final tagging result $\tau_R(w_t)$.

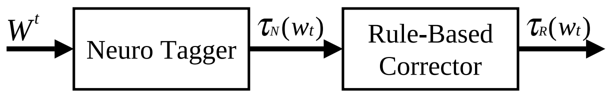

Figure 1: Hybrid neuro and rule-based tagger.

3.1 Neuro tagger

As shown in Fig. 2, the neuro tagger consists of a three-layer perceptron with elastic input. This section mainly describes the construction of input and output of the neuro tagger, and the elasticity by which it becomes possible to use variable length of context for tagging. For details of the architecture of perceptron see e.g., Haykin, 1994 and for details of the features of the neuro tagger see Ma and Isahara, 1998 and Ma, et al., 1999.

Input IPT is constructed from word sequence W^t [Eq. (2)], which is centered on target word w_t and has length $l + 1 + r$:

$$IPT = (ipt_{t-l}, \cdots, ipt_t, \cdots, ipt_{t+r}), \qquad (3)$$

provided that input length $l+1+r$ has elasticity, as described at the end of this section. When word w is given in position x ($x = t-l, \cdots, t+r$),

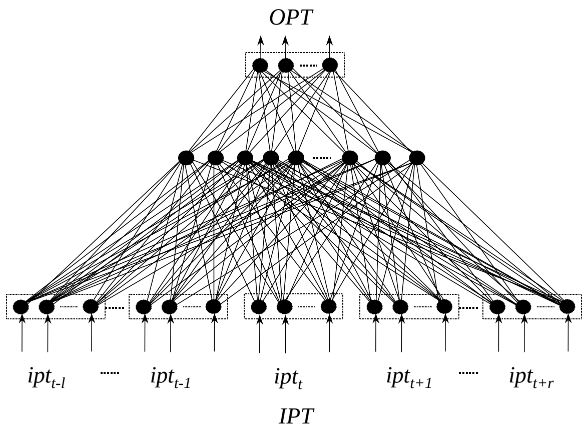

Figure 2: Neuro tagger.

element ipt_x of input IPT is a weighted pattern, defined as

$$ipt_x = g_x \cdot (e_{w1}, e_{w2}, \cdots, e_{w\gamma}), \qquad (4)$$

where g_x is the information gain which can be obtained using information theory (for details see Ma and Isahara, 1998) and γ is the number of types of POSs. If w is a word that appears in the training data, then each bit e_{wi} can be obtained:

$$e_{wi} = Prob(\tau^i|w), \qquad (5)$$

where $Prob(\tau^i|w)$ is a prior probability of τ^i that the word w can take. It is estimated from the training data:

$$Prob(\tau^i|w) = \frac{C(\tau^i, w)}{C(w)}, \qquad (6)$$

where $C(\tau^i, w)$ is the number of times both τ^i and w appear, and $C(w)$ is the number of times w appears in the training data. If w is a word that does not appear in the training data, then each bit e_{wi} is obtained:

$$e_{wi} = \begin{cases} \frac{1}{\gamma_w} & \text{if } \tau^i \text{ is a candidate} \\ 0 & \text{otherwise,} \end{cases} \qquad (7)$$

where γ_w is the number of POSs that the word w can take. Output OPT is defined as

$$OPT = (O_1, O_2, \cdots, O_\gamma), \qquad (8)$$

provided that the output OPT is decoded as

$$\tau_N(w_t) = \begin{cases} \tau^i & \text{if } O_i = 1 \;\&\; O_j = 0 \text{ for } j \neq i \\ Unknown & \text{otherwise,} \end{cases} \qquad (9)$$

where $\tau_N(w_t)$ is the tagging result obtained by the neuro tagger.

There is more information available for constructing the input for words on the left, because they have already been tagged. In the tagging phase, instead of using (4)-(6), the input can be constructed simply as

$$ipt_{t-i} = g_{t-i} \cdot OPT(-i), \qquad (10)$$

where $i = 1, \cdots, l$, and $OPT(-i)$ means the output of the tagger for the ith word before the target word. However, in the training process, the output of the tagger is not always correct and cannot be fed back to the inputs directly. Instead, a weighted average of the actual output and the desired output is used:

$$ipt_{t-i} = g_{t-i} \cdot (w_{OPT} \cdot OPT(-i) + w_{DES} \cdot DES), \qquad (11)$$

where DES is the desired output,

$$DES = (D_1, D_2, \cdots, D_\gamma), \qquad (12)$$

whose bits are defined as

$$D_i = \begin{cases} 1 & \text{if } \tau^i \text{ is a desired answer} \\ 0 & \text{otherwise,} \end{cases} \qquad (13)$$

and w_{OPT} and w_{DES} are respectively defined as

$$w_{OPT} = \frac{E_{OBJ}}{E_{ACT}} \qquad (14)$$

and

$$w_{DES} = 1 - w_{OPT}, \qquad (15)$$

where E_{OBJ} and E_{ACT} are the objective and actual errors. Thus, at the beginning of training, the weighting of the desired output is large. It decreases to zero during training.

Elastic inputs are used in the neuro tagger so that the length of context is variable in tagging based on longest context priority. In detail, (l, r) is initially set as large as possible for tagging. If $\tau_N(w_t) = Unknown$, then (l, r) is reduced by some constant interval. This process is repeated until $\tau_N(w_t) \neq Unknown$ or $(l, r) = (0, 0)$. On the other hand, to make the same set of connection weights of the neuro tagger with the largest (l, r) available as much as possible when using short inputs for tagging, in training phase the neuro tagger is regarded as a neural network that has gradually grown from small one. The training is therefore performed step by step from small networks to large ones (for details see Ma, et al. 1999).

3.2 Rule-based corrector

Even when the POS of a word can be determined with certainty by only the word on the left, for example, the neuro tagger still tries to tag based on the complete context. That is, in general, what the neuro tagger can easily acquire by learning is the rules whose conditional parts are constructed by all inputs ipt_x $(x = t - l, \cdots, t + r)$ that are joined with an AND logical operator, i.e., $(ipt_{t-l}\ \&\ \cdots\ ipt_t\ \&\ \cdots\ ipt_{t+r} \rightarrow OPT)$. In other words, it is difficult for the neuro tagger to learn rules whose conditional parts are constructed by only a single input like $(ipt_x \rightarrow OPT)$[1]. Also, although lexical information is very important in tagging, it is difficult for the neuro tagger to use it, because doing so would make the network enormous. That is, the neuro tagger cannot acquire rules whose conditional parts consist of lexical information like $(w \rightarrow OPT)$, $(w\&\tau \rightarrow OPT)$, and $(w_1\&w_2 \rightarrow OPT)$, where w, w_1, and w_2 are words and τ is the POS. Furthermore, because of convergence and over-training problems, it is impossible and also not advisable to train neural nets to an accuracy of 100%. The training should be stopped at an appropriate level of accuracy. Thus, neural net may not acquire some useful rules.

The transformation rule-based corrector makes up for these crucial shortcomings. The rules are acquired from a training corpus using a set of transformation templates by transformation-based error-driven learning (Brill, 1994). The templates are constructed using only those that supply the rules that the neuro tagger can hardly acquire, i.e., are those for acquiring the rules with single input, with lexical information, and with AND logical input of POSs and lexical information. The set of templates is shown in Table 1[2].

According to the learning procedure shown in Table 2, an ordered list of transformation rules are acquired by applying the template set to a training corpus, which had already been tagged by the neuro tagger. After the transformation rules are acquired, a corpus is tagged as follows. It is first tagged by the neuro tagger. The tagged corpus is then corrected by using the ordered list of transformation rules. The correction is a repetitive process applying the rules in order to the corpus, which is then updated, until all rules have been applied.

4 Experimental Results

Data: For our computer experiments, we used the same Thai corpus used by Ma et al. (1999). Its 10,452 sentences were randomly divided into two sets: one with 8,322 sentences for training and the other with 2,130 sentences for testing. The training set contained 124,331 words, of which 22,311 were ambiguous; the testing set contained 34,544 words, of which 6,717 were ambiguous. For training the neuro tagger, only the ambiguous words in the training set were used. For training the HMM, all the words in the training set were used. In both cases, all the words in the training set were used to estimate $Prob(\tau^i|w)$, the probability of τ^i that word w can be (for details on the HMM, see Ma, et al., 1999). In the corpus, 47 types of POSs are defined (Charoenporn et al., 1997); i.e., $\gamma = 47$.

Neuro tagger: The neuro tagger was constructed by a three-layer perceptron whose input-middle-output layers had $p - \frac{p}{2} - \gamma$ units, respectively, where $p = \gamma \times (l + 1 + r)$. The $(l + 1 + r)$ had the following elasticity. In training, the (l, r) was increased step by step as (1,1) $\rightarrow$ (2,1) $\rightarrow$ (2,2) $\rightarrow$ (3,2) $\rightarrow$ (3,3) and gradual training from a small to a large network was performed. In tagging, on the other hand, the

[1]The neuro tagger can also learn this kind of rules because it can tag the word using only ipt_t (the input of the target word), in the case of reducing the (l, r) to (0,0), as described in Sec. 3.1. The rules with single input described here, however, are a more general case, in which the input can be ipt_x $(x = t - l, \cdots, t + r)$.

[2]To see whether this set is suitable, a number of additional experiments were conducted using various sets of templates. The details are described in Sec. 4.

Table 1: Set of templates for transformation rules

Change tag τ^a to tag τ^b when:
(single input)
(input consists of a POS)
1. left (right) word is tagged τ.
2. second left (right) word is tagged τ.
3. third left (right) word is tagged τ.
(input consists of a word)
4. target word is w.
5. left (right) word is w.
6. second left (right) word is w.
(AND logical input of words)
7. target word is w_1 and left (right) word is w_2.
8. left (right) word is w_1 and second left (right) word is w_2.
9. left word is w_1 and right word is w_2.
(AND logical input of POS and words)
10. target word is w_1 and left (right) word is tagged τ.
11. left (right) word is w_1 and left (right) word is tagged τ.
12. target word is w_1, left (right) word is w_2, and left (right) word is tagged τ.

Table 2: Procedure for learning transformation rules

1. Apply neuro tagger to training corpus, which is then updated.
2. Compare tagged results with desired ones and find errors.
3. Match templates for all errors and obtain set of transformation rules.
4. Select rule in corpus with the maximum value of ($cnt\_good - h \cdot cnt\_bad$), where
   cnt_good: number that transforms incorrect tags to correct ones,
   cnt_bad: number that transforms correct tags to incorrect ones,
   h: weight to control the strictness of generating the rule.
5. Apply selected rule to training corpus, which is then updated.
6. Append selected rule to ordered list of transformation rules.
7. Repeat steps 2 through 6 until no such rule can be selected, i.e., $cnt\_good - h \cdot cnt\_bad \leq 0$.

(l, r) was inversely reduced step by step as (3,3) $\rightarrow$ (3,2) $\rightarrow$ (2,2) $\rightarrow$ (2,1) $\rightarrow$ (1,1) $\rightarrow$ (1,0) $\rightarrow$ (0,0) as needed, provided that the number of units in the middle layer was kept at the maximum value.

Rule-based corrector: The parameter h in the evaluation function ($cnt\_good - h \cdot cnt\_bad$) used in the learning procedure (Table 2) is a weight to control the strictness of generating a rule. If h is large, the weight of *cnt_bad* is large and the possibility of generating incorrect rules is reduced. By regarding the neuro tagger as already having high accuracy and using the rule-based corrector as a fine tuner, weight h was set to a large value, 100. Applying the templates to the training corpus, which had already been tagged by the neuro tagger, we obtained an ordered list of 520 transformation rules. Table 3 shows the first 15 transformation rules.

Results: Table 4 shows the results of POS tagging for the testing data. In addition to the accuracy of the neuro tagger and hybrid system, the table also shows the accuracy of a baseline model, the HMM, and a rule-based model for comparison. The baseline model is one that performs tagging without using the contextual information; instead, it performs tagging using only frequency information: the probability of POS that each word can be. The rule-based model, to be exact, is also a hybrid system con-

Table 3: First 15 transformation rules

No.	From	To	Condition
1	PREL	RPRE	left word is punctuation and right word is ระดับ
2	PREL	RPRE	left word is อยู่
3	Unknown	ADVN	left word is tagged XVAE
4	XVMM	XVBM	left word is หรือ
5	VATT	ADVN	left word is กัน
6	Unknown	VATT	left word is tagged PREL
7	NCMN	RPRE	left word is ผล
8	VATT	VSTA	left word is สามัญ
9	PREL	RPRE	right word is ระดับ and second right word is ความ
10	VSTA	ADVN	target word is ต่อเนื่อง
11	VATT	ADVN	target word is สูงสุด
12	NCMN	RPRE	target word is ทาง and left word is ออกแบบ
13	NCMN	RPRE	left word is ใน and left word is tagged NCMN
14	Unknown	ADVN	third left word is tagged VACT
15	NCMN	CNIT	target word is ภาพ

where PREL: Relative Pronoun, RPRE: Preposition, ADVN: Adverb with normal form, ...

Table 4: Results of POS tagging for testing data*

model	baseline	HMM	rule-based	neuro	hybrid
accuracy	0.836	0.891	0.935	0.944	0.955

$*$*Accuracy* was determined only for ambiguous words.

sisting of an initial-state annotator and a set of transformation rules. As the initial-state annotator, however, the baseline model is used instead of the neuro tagger. And, its rule set has 1,177 transformation rules acquired from a more general template set, which is described at the end of this section. The reason for using a general template set is that the set of transformation rules in the rule-based model should be the main annotator, not a fine post-processing tuner. For the same reason, the parameter to control the strictness of generating a rule, h, was set to a small value, 1, so that a larger number of rules were generated.

As shown in the table, the accuracy of the neuro tagger was far higher than that of the HMM and higher than that of the rule-based model. The accuracy of the rule-based model, on the other hand, was also far higher than that of the HMM, although it was inferior to that of the neuro tagger. The accuracy of the hybrid system was 1.1% higher than that of the neuro tagger. Actually, the rule-based corrector corrected 88.4% and 19.7% of the errors made by the neuro tagger for the training and testing data, respectively.

Because the template set shown in Table 1 was designed only to make up for the shortcomings of the neuro tagger, the set is small compared to that used by Brill (1994). To see whether this set is large enough for our system, we performed two additional experiments in which (1) a set constructed by adding the templates with OR logical input of words to the original set and (2) a set constructed by further adding the templates with AND and OR logical inputs of POSs to the set of case (1) were used. The set used in case (2) included the set used by Brill (1994) and all the sets used in our experiments. It was also used for acquiring the transformation rules in the rule-based model. The experimental results show that compared to the original case, the accuracy in case (1) was improved very little and the accuracy in case (2) was also improved only 0.03%. These results show that the original set is nearly large enough for our system.

To see whether the set is suitable for our system, we performed an additional experiment using the original set in which the templates with OR logical inputs were used instead of the templates with AND logical inputs. The accuracy dropped by 0.1%. Therefore, the templates with AND logical inputs are more suitable than those with OR logical inputs.

We also performed an experiment using a template set without lexical information. In this case, the accuracy dropped by 0.9%, indicating that lexical information is important in tagging.

To determine the effect of using a large h for generating rules, we performed an experiment with $h = 1$. In this case, the accuracy dropped by only 0.045%, an insignificant difference compared to the case of $h = 100$.

By examining the acquired rules that were obtained by applying the most complete template set, i.e., the set used in case (2) described above, we found that 99.9% of them were those that can be obtained by applying the original set of templates. That is, the acquired rules were almost those that are difficult for the neuro tagger to acquire. This reinforced our expectation that the rule-based approach is a well-suited method to cope with the shortcoming of the neuro tagger.

Finally, it should be noted that in the literatures, the tagging accuracy is usually defined by counting all the words regardless of whether they are ambiguous or not. If we used this definition, the accuracy of our hybrid system would be 99.1%.

5 Conclusion

To construct a practical tagger that needs as little training data as possible, neuro taggers, which have high generalizing ability and therefore are good at dealing with the problems of data sparseness, have been proposed so far. Neuro taggers, however, have crucial shortcomings: they cannot utilize lexical information; they have trouble learning rules with single inputs; and they cannot learn training data to an accuracy of 100%. To make up for these shortcomings, we introduced a rule-based corrector, which is constructed by a set of transformation rules obtained by error-driven learning, for post processing and constructed a hybrid tagging system. By examining the transformation rules acquired in the computer experiments, we found that the 99.9% of them were those that the neuro tagger can hardly acquire, even when using a template set including those for generating the rules that the neuro tagger can easily acquire. This reinforced our expectation that the rule-based approach is a well-suited method to cope with the shortcoming of the neuro tagger. Computer experiments showed that 19.7% of the errors made by the neuro tagger were corrected by the transformation rules, so the hybrid system reached an accuracy of 95.5% counting only the ambiguous words and 99.1% counting all the words in the testing data, when a small corpus with only 22,311 ambiguous words was used for training. This indicates that our tagging system can nearly reach a practical level in terms of tagging accuracy even when a small Thai corpus is used for training. This kind of tagging system can be used to constructs multilingual corpora that include languages in which large corpora have not yet been constructed.

References

Brill, E.: Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging, *Computational Linguistics*, Vol. 21, No. 4, pp. 543-565, 1994.

Charoenporn, T., Sornlertlamvanich, V., and Isahara, H.: Building a large Thai text corpus - part of speech tagged corpus: ORCHID, *Proc. Natural Language Processing Pacific Rim Symposium 1997*, Phuket, Thailand, pp. 509-512, 1997.

Daelemans, W., Zavrel, J., Berck, P., and Gillis, S.: MBT: A memory-based part of speech tagger-generator, *Proc. 4th Workshop on Very Large Corpora*, Copenhagen, Denmark, pp. 1-14, 1996.

Haykin, S.: *Neural Networks*, Macmillan College Publishing Company, Inc., 1994.

Ma, Q. and Isahara, H.: A multi-neuro tagger using variable lengths of contexts, *Proc. COLING-ACL'98*, Montreal, pp. 802-806, 1998.

Ma, Q., Uchimoto, K., Murata, M., and Isahara H.: Elastic neural networks for part of speech tagging, *Proc. IJCNN'99*, Washington, DC., pp. 2991-2996, 1999.

Merialdo, B.: Tagging English text with a probabilistic model, *Computational Linguistics*, Vol. 20, No. 2, pp. 155-171, 1994.

Quinlan, J.: *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann, 1993.

Schmid, H.: Part-of-speech tagging with neural networks, *Proc. COLING'94*, Kyoto, Japan, pp. 172-176, 1994.