Gradient-Descent Error Correction of POS Tagging

Prachya Boonkwan[†] **Thepchai Supnithi**[†] **Jaruwat Pailai**[‡] **Rachada Kongkachandra**[‡]
[†]Language and Semantic Technology Lab
National Electronics and Computer Technology Center (NECTEC), Thailand
[‡]Department of Computer Science, Faculty of Science and Technology
Thammasat University, Thailand
{prachya.boonkwan,thepchai.supnithi}@nectec.or.th
{pomkab,rdkpansy}@gmail.com

Abstract

POS tagging is a nontrivial task as it provides preliminary syntactic information to input sentences, proven to improve the accuracy of many NLP tasks that require syntactic analysis. The task of POS tagging for analytic languages (e.g. Thai, Chinese, etc.) is particularly challenging because syntactic ambiguity caused by sentence-like noun phrases and vice versa have to be statistically disambiguated in this level due to the annotation schema. In this paper, we report a comparative study of baseline accuracies and solving this problem by applying locality learning on nouns and verbs and experimenting on various models (SVMs and CRFs) and numbers of grams. BEST 2012 Corpus's news and entertainment sections, previously annotated with word boundaries, POS tags, and named entities, are used as our gold standard and test set. The experiment results show that CRFs outperform SVMs, where the best tagging accuracy is 96.46%, and local retagging can boost that to 97.82%.

Keyword: POS tagging, Thai language, natural language processing, support vector machine (SVM), conditional random fields (CRFs), language models, local retagging

1 Introduction

Part of Speech (POS) tagging is a task of assigning each word of the input text with a part-of-speech tag that carries lexical description. POS taggers are often used prior to syntactic parsing to reduce the ambiguity of parts of speech [1; 2; 3]. The POS tagging problem is local compared to parsing because it makes use of much more limited context, e.g. n previous POS tags or n-gram models. Although POS taggers can be trained in semi-supervised or unsupervised fashions, we are interested in an improvement of the supervised taggers such as [1], where certain syntactic ambiguities can be solved locally. We believe that doing so will have an impact on the accuracy.

In analytic languages such as Thai, Chinese, and Lao, POS tagging becomes a challenging problem because some ambiguous syntactic constructions cannot be distinguished by grammatical information that is normally determined morphologically. For example, in Thai, a simple sentence resembles a noun phrase, sharing the same patterns: Noun-Verb-Noun. In Chinese, pattern Noun-Verb-DE-Noun can be interpreted as either a sentence (DE is treated as an adverb) or a noun phrase (DE is treated as a relative pronoun). This issue of syntactic ambiguity significantly enlarges the search space of parsing.

One popular remedy to this issue is to distinguish these ambiguous tags before parsing. This has been widely applied in various linguistic resources such as Chinese Penn Treebank [4] and BEST Corpus [5]. However, following this scheme is likely to lower the accuracy of the taggers, because each tag is locally predicted from its context.

In this paper, we seek to correct frequent erroneous annotation by locally retagging them statistically. Ironically speaking, we will retag problematic tags produced by a baseline tagger (e.g. SVMs and CRFs) with another simple tagger particularly trained on them. We will show that we can achieve accuracy improvement by incorporating a simple local retagger into the baseline tagger.

The remainder of this paper is organized as follows. §2 formally describes our local retagging technique. §3 explains our baseline models which will be incorporated with our technique to show accuracy improvement. §4 shows experiment results. §5 discusses the experiment re-

sults. §6 elaborates some previous work relevant to the task of POS tagging and our technique. Finally, §7 concludes this paper.

2 Global and Local Taggers

We propose the use of globally and locally trained taggers to boost the accuracy of POS annotation by using the locally trained tagger to correct potentially ambiguous tags produced by the globally trained tagger. Although global and local taggers are completely separated, they can be trained in parallel.

Let us formalize our training strategy as follows. We have a set of training sentences $D = \{\mathbf{w}_i | 1 \leq i \leq M\}$ and each sentence $\mathbf{w}_i$ of length N_i is annotated with a corresponding sequence of POS tags $\mathbf{t}_i$. We would like to train a baseline tagger (called a *global tagger* henceforth) by estimating its parameters $\mathbf{a}_G$ from D with respect to a differentiable objective function F such as SVMs and CRFs. We then have a set of potentially ambiguous tags $\mathcal{Z}$ which will be reannotated by a simpler tagger (called a *local retagger* henceforth). We also estimate the the local retagger's parameters $\mathbf{a}_L$ from D with respect to the same objective function.

As illustrated in Algorithm 1, our training procedure is quite straightforward. We first initialize the parametes $\mathbf{a}_G$ and $\mathbf{a}_L$. Then we estimate $\mathbf{a}_G$ using the gradient descent with step size γ; i.e. we update $\mathbf{a}_G$ with the gradient $\nabla F(\cdot|\mathbf{a}_G)$ given the current parameters $\mathbf{a}_G$. Once we find any problematic tag t_{ij} described in $\mathcal{Z}$ along the way, we also estimate $\mathbf{a}_L$ using the gradient descent by updating $\mathbf{a}_L$ with the gradient $\nabla F(\cdot|\mathbf{a}_L)$ given the current parameters $\mathbf{a}_L$. The estimation process reiterates until the global tagger's parameters $\mathbf{a}_G$ converges.

We decided to train both taggers in the same time because of time saving reasons. There is a performance trade-off of the gradient descent when choosing step size γ. If the step size is too small or too large, it may take a lot of iterations to obtain the optimal parameters. Finding gradients $\nabla F(\cdot|\mathbf{a}_G)$ and $\nabla F(\cdot|\mathbf{a}_L)$ can be achieved by using linear optimization algorithms such as Limited-Memory BFGS Algorithm (L-BFGS) [6; 7]. Optionally, one can choose to train the global and local taggers separately but the performance difference is quite marginal when the number of iterations becomes large.

In tagging, we would like to annotate a set of testing sentences $D' = \{\mathbf{w}_i | 1 \leq i \leq M\}$ with sequences of POS tags; i.e. we would like to find for each sentence $\mathbf{w}_i$ the most likely sequence of POS tags T_i. Then if any tag T_{ij} is potentially ambiguous, we will replace current tag with a more optimal one t'_j, denoted by $t_i^{\downarrow t'_j}$.

Our tagging algorithm makes use of both $\mathbf{a}_G$ and $\mathbf{a}_L$ as illustrated in Algorithm 2. It can be seen as cascaded decoding. First, we use the global tagger to tag each sentence $\mathbf{w}_i$ with a sequence of tags T_i; i.e. we find the most likely tag sequence T_i with respect to the objective function $F(\cdot|\mathbf{a}_G)$ based on the global parameters. This step can be done by various statistical decoding algorithms such as Viterbi Algorithm. Then we scan for potentially ambiguous tags in T_i. If any tag T_{ij} is ambiguous as described in $\mathcal{Z}$, we then locally retag it with a new tag t'_j computed from the objective function $F(\cdot|\mathbf{a}_L)$.

Algorithm 1 `cascaded_learn`$(D, \mathcal{Z})$

▷ *Globally estimated model*
let $\mathbf{a}_G \leftarrow$ `init_params`(D)
▷ *Locally estimated model*
let $\mathbf{a}_L \leftarrow$ `init_params`(D)
▷ *Estimate* $\mathbf{a}_G$.
repeat
  for $i \leftarrow 1$ to M **do**
    for $j \leftarrow 1$ to N_i **do**
      let $\Phi_j \leftarrow \nabla F(j, \mathbf{w}_i, \mathbf{t}_i | \mathbf{a}_G)$
      set $\mathbf{a}_G \leftarrow \mathbf{a}_G - \gamma \Phi_j$
      if $t_{ij} \in \mathcal{Z}$ **then**
        let $\Psi_j \leftarrow \nabla F(j, \mathbf{w}_i, \mathbf{t}_i | \mathbf{a}_L)$
        set $\mathbf{a}_L \leftarrow \mathbf{a}_L - \gamma \Psi_j$
      end if
    end for
  end for
until $\mathbf{a}_G$ converges
return $(\mathbf{a}_G, \mathbf{a}_L)$

3 Baselines

In order to pinpoint the problems of POS annotation, we developed two baseline taggers by varying the sets of features and objective functions.

3.1 Features

We make use of n-gram models as the sets of features. There are two sets of features:

1. Minimalist trigram: This set of features includes all forward trigrams including two

Algorithm 2 `cascaded_tag`$(D', \mathcal{Z}, \mathbf{a}_G, \mathbf{a}_L)$

let $T \leftarrow \texttt{array}(1, M)$
▷ *Tag with the globally estimated model.*
for $i \leftarrow 1$ **to** M **do**
  set $T_i \leftarrow \arg\max_{\mathbf{t}} F(i, \mathbf{w}_i, \mathbf{t}|\mathbf{a}_G)$
  ▷ *Retag with the locally estimated model.*
  for $j \leftarrow 1$ **to** N_i **do**
    if $T_{ij} \in \mathcal{Z}$ **then**
      set $T_{ij} \leftarrow \arg\max_{t'_j} F(i, \mathbf{w}_i, \mathbf{t}_i^{\downarrow t'_j}|\mathbf{a}_L)$
    end if
  end for
end for
return T

next word forms and the current POS tag; i.e. each feature is in form:

$$\langle w_i, t_i, w_{i+1}, w_{i+2} \rangle \quad (1)$$

where each w_i is the current word form, t_i is its POS tag, and i is the current index.

2. Surrounding trigram: This set of features includes all surrounding trigrams including word forms and POS tags; i.e. each feature is in form:

$$\langle w_{i-2}, t_{i-2}, w_{i-1}, t_{i-1}, w_i, t_i, w_{i+1}, t_{i+1}, w_{i+2}, t_{i+2} \rangle \quad (2)$$

3.2 Objective Functions

We employ two discriminative models to develop the baseline taggers.

Support Vector Machines (SVMs): An SVM [8] is a non-probabilistic linear classifier that maps its inputs into a high-dimensional feature space. It performs classification in such space using a kernel function (such as linears, Gaussian radials, and hyperbolic tangents) whose parameters are optimized for clear class boundaries. In this paper, we use Gaussian radials as the kernel function, where the distance between any two observations ϕ_i and ϕ_j is measured by:

$$k(\phi_i, \phi_j) = \exp(-\gamma \|\phi_i - \phi_j\|^2) \quad (3)$$

for some $\gamma > 0$ which is normally set to $\frac{1}{2\sigma^2}$. The objective function of this model is the total distance between observations.

Conditional Random Fields (CRFs): A CRF [9] is an undirected probabilistic graphical model that captures relationships between observations with consistent interpretation encoded in terms of potential functions defined upon a feature space. In this paper, we use as a potential function a joint probability of a word w_i having POS tag t_i given its surrounding context ϕ_i; i.e.

$$\Psi(w_i, t_i, \phi_i) = P(t_i|w_i, \phi_i) \quad (4)$$

Parameters of CRFs are the co-efficients of each potential function. The objective function of this model is the probability of the entire observations computed from the potential functions and their co-efficients.

We utilize LIBSVM [10] and CRFsuite [11] for estimating the parameters for SVM and CRF, respectively.

3.3 Baseline Taggers and Retaggers

We developed two baseline taggers with the following scheme:

1. SVM: Our SVM tagger and retagger make use of Gaussian radials as the kernel function and the minimalist trigram as the feature space. We use the minilist features for SVM because of space limitation in parameter estimation using LIBSVM.

2. CRF: Our CRF tagger and retagger use a joint probability $P(t_i|w_i, \phi_i)$, where t_i is the current tag, w_i is the current word, and ϕ_i is its context, as a potential function and employ the surrounding trigrams as the feature space. We use the richer surrounding features for CRF because an implementation of L-BFGS Algorithm used in CRFsuite is more space-efficient in parameter estimation.

4 Experiments

4.1 Dataset and Experiment Settings

We evaluate our method against BEST 2012 News and Entertainment Corpus developed by NECTEC, Thailand [5]. It is a Thai language corpus that is manually word-segmented, and annotated with POS tags and named entities by graduate linguists. The tagset contains 34 syntactic labels. At the time of experiments, there are one million POS-annotated words in total. The peculiarity of its tagset is that, roughly

Table 1. Baseline F_1 accuracies evaluated on the entire corpus

Models	Avg F_1	S.D.
SVM	92.84	0.69
CRF	94.50	0.44

speaking, it classifies verbs into two broad types: VV (verb) and JJV (noun-modifying verb), to help shrink the search space of parsing.

From our observation, we found that most frequent tags annotated to the corpus are NN (noun; found 24.44%) and VV (verb; found 25.88%). They are major sources of errors because they are likely to be mistakenly tagged as other tags. We believe that correctly retagging these tags will have a strong impact on the accuracy. We therefore set our set of problematic tags $\mathcal{Z}$ as $\{\texttt{NN}, \texttt{VV}\}$; i.e. we will locally retag nouns and verbs when found.

In each experiment, we initialize the global and local parameters $\mathbf{a}_G$ and $\mathbf{a}_L$ by setting them to a non-zero uniform distribution. We then estimate them with a training set using Algorithm 1 and tag a test set using Algorithm 2. The accuracy is measured against the gold standard and reported in terms of F_1 scores.

4.2 Experiment 1: Baselines

First we evaluate our baseline models: SVM and CRF as mentioned in §3.3, without local retagging. We run 10-cross validation on both models against the entire one million words of the corpus. The accuracies are reported in Table 1. It is clear that CRF outperforms SVM with lower standard deviation despite its richer feature space.

We also evaluate each model as the data size varies. We first randomly divide the corpus into ten chunks of size 100,000 words. In evaluation, we increase the data size by incorporating more chunks into the training set, leaving the remainder as the gold standard. Using all 1,000,000 words means that we use the training set as the gold standard only to observe the accuracy threshold. The accuracy trends are shown in Figure 1. CRF still outperforms SVM from the beginning, which shows that CRFs are generally more stable to the change of training data sizes than SVMs do although the feature space of the former is richer than that of the latter.

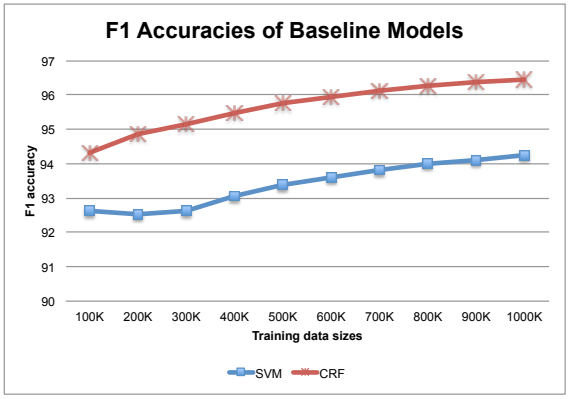

Figure 1. F_1 accuracy trends regarding the change of training data sizes

Table 2. F_1 accuracies before and after local retagging is incorporated

Models	F_1 Accuracy		Δ
	Baseline	Retagging	
SVM	94.12	96.06	+1.94
CRF	96.37	97.62	+1.25

4.3 Experiment 2: Local Retagging

First we introduce local retagging to our models: SVM and CRF and evaluate them against the gold standard. We run 10-cross validation on both models against the entire one million words of the corpus. The accuracies are reported in Table 2.

Although CRF outperforms SVM due to its model stability, it is intriguing to see that local retagging boosts more accuracy of SVM than that of CRF. We hypothesize that the baseline SVM is less tolerant to the curse of dimensionality than the baseline CRF, thus producing more errors. When we locally retag problematic tags i.e. NN and VV, we are concentrating on only a particular part of the feature space. This makes the data sparsity issue less influential and thus improves the accuracy.

We also evaluate our local retagging method with various training data sizes. As shown in Figure 2, it improves the accuracies of both SVM and CRF in all training data sizes. Likewise, the accuracies of both models incorporated with local retagging saturate earlier than their baseline counterparts. We hypothesize that we have reached the accuracy threshold where marginal errors caused by infrequent problematic tags are produced. Although it is not surprising that the

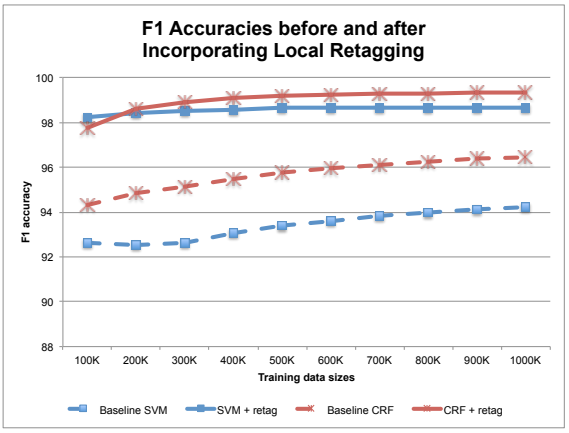

Figure 2. F_1 accuracy trends before and after incorporating local retagging

accuracy almost approaches 100% F1, we observe that CRF marginally outperforms SVM when we evaluate the system trained on the testing set (one million words).

5 Discussion

5.1 The Curse of Dimensionality

From all experiments we found that CRF is more tolerant to the issue of data sparsity than SVM although our CRF model has a richer feature space. This is because SVM's kernel function is harder to choose for consistent interpretation according to the observations and its parameters are relatively complicated to optimize. On the other hand, CRF's coefficients are much easier to optimize even on large feature spaces, given that we have already chosen potential functions. We therefore recommend the use of CRF in the tagging problem.

5.2 Retagging Frequent Problematic Tags

Local retagging on frequent problematic tags improves the accuracy of the baseline models. Our experiments show that errors are generated more frequently by frequent tags: `NN` and `VV`. Once we retag those problematic cases, the accuracies of both models start to approach each other. That means the curse of dimensionality becomes less influential to the accuracy when we concentrate only on a particular part of the feature space. We envisage that if we locally retag frequent problematic tags and repeat the process on less frequent ones, the accuracy is likely to be improved. We therefore suggest that hierarchical retagging is capable of solving the curse of dimensionality in complex models.

6 Related Work

POS tagging is a non-trivial NLP task. Initially, the task was achieved by several techniques including rule-based approach, stochastic prediction, and transformation-based learning. Rule-based taggers [12; 13; 14] assigned a tag to each word using a set of rules handcrafted by linguistic experts. This approach poses several problems in constructing the rules including rule conflicts and coverage of long-tail problematic cases. Machine learning techniques are nowadays widely used in the task, where POS tagging is considered as a sequential data classification problem. Those techniques include Hidden Markov Models (HMM) [15], SVM [8], and CRF [9].

As aforementioned, POS tagging for Thai is a challenging task because of sentence-like noun phrase issue and the like. Murata et al. [16; 17] compare eight machine learning techniques when applied to the task and conclude that SVM outperforms HMM. Pailai et al. [18] run their experiments on BEST 2012 Corpus and analyze frequent tagging errors caused by the data sparsity issue in terms of finite state machines.

This paper presents a local retagging method on these frequent problematic tags with a retagging model especially trained on them. We use a gradient descent algorithm to estimate the parameters of the global tagging model and the local retagging models. We show that this method can improve the accuracy of this task because we partially overcome the issue of data sparsity by concentrating on a particular part of the feature space.

7 Conclusion

We have presented an error correction technique for POS tagging by locally retagging frequent problematic tags generated from the baseline taggers. The experiments show that CRF is more tolerant to the issue of data sparsity than SVM. Local retagging on frequent problematic tags improves the accuracy of POS tagging because it concentrates on a particular part of the feature space, making the issue of data sparsity less influential.

Although we have shown that our technique

performs well, future work still remains on POS tagging. First, we will investigate the use of hierarchical local retagging based on the POS histogram. By doing so, we can gradually narrow the feature space on less frequent problematic cases and may obtain some more accuracy. Second, we will apply this technique to other tasks equivalent to a sequential data classification problem, such as named-entity recognition, and conduct error analysis. Third, we will apply this technique to other analytic languages and conduct linguistics-oriented error analysis. Fourth and last, we will investigate how to choose optimal kernel functions, potential functions, and features for the baseline models.

References

[1] K. W. Church, "A stochastic parts program and noun phrase parser for unrestricted text," in *Proceedings of the Second Conference on Applied Natural Language Processing*, 1988, pp. 136–143.

[2] E. Brill, "Transformation-based error driven parsing," in *Proceedings of the Third International Workshop on Parsing Technologies*, 1993, pp. 1–13.

[3] R. Weischedel, M. Meteer, R. Schwartz, L. Ramshaw, and J. Palmucci, "Coping with ambiguity and unknown words through probabilistic models," *Computational Linguistics*, vol. 19, no. 2, pp. 359–382, 1993.

[4] F. Xia, "The part-of-speech tagging guidelines for the Penn Chinese Treebank (3.0)," University of Pennsylvania, Tech. Rep., 2000.

[5] (2013, September) BEST Corpus. [Online]. Available: http://www.nectec.or.th/corpus/index.php/2011-06-18-07-44-24/2011-08-27-06-21-08/category/3-best-i-corpus

[6] R. Malouf, "A comparison of algorithms for maximum entropy parameter estimation," in *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL)*, 2002, pp. 49–55.

[7] G. Andrew and J. Gao, "Scalable training of l_1-regularized log-linear models," in *Proceedings of the 24th International Conference on Machine Learning*, 2007.

[8] C. Cortes and V. N. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, 1995.

[9] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 2001, pp. 282–289.

[10] (2013, September) Libsvm: A library for support vector machines website. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[11] (2013, September) Crfsuite: A fast implementation of conditional random fields (crfs). [Online]. Available: http://www.chokkan.org/software/crfsuite

[12] B. Greene and G. Rubin, "Automatic grammatical tagging of English," Department of Linguistics, Brown University, Brown University, Providence, Rhode Island, Tech. Rep., 1971.

[13] S. Klein and R. Simmons, "A computational approach to grammatical coding of English words," *Journal of the ACM*, vol. 10, pp. 334–347, July 1963.

[14] Z. Harris, *String Analysis of Language Structure*. The Hague, Netherlands: Mouton and Co., 1962.

[15] L. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.

[16] M. Murata, Q. Ma, and H. Isahara, "Part of speech tagging in Thai language using support vector machine," in *Proceedings of the Second Workshop on Natural Language Processing and Neural Networks (NLPNN)*, 2001, pp. 24–30.

[17] ——, "Comparison of three machine-learning methods for Thai part-of-speech tagging," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 1, no. 2, pp. 145–158, June 2002.

[18] J. Pailai, R. Kongkachandra, T. Supnithi, and P. Boonkwan, "A comparative study on different techniques for thai part-of-speech tagging," in *Proceedings of the 10th*

Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology (ECTI-CON), 2013, pp. 1–5.