Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Active Learning with Subsequence Sampling Strategy for Sequence Labeling Tasks

Dittaya Wanvarie[†], Hiroya Takamura[††] and Manabu Okumura[††]

We propose an active learning framework for sequence labeling tasks. In each iteration, a set of subsequences are selected and manually labeled, while the other parts of sequences are left unannotated. The learning will stop automatically when the training data between consecutive iterations does not significantly change. We evaluate the proposed framework on chunking and named entity recognition data provided by CoNLL. Experimental results show that we succeed in obtaining the supervised F_1 only with 6.98%, and 7.01% of tokens being annotated, respectively.

Key Words: *Active Learning, Sequence Labeling, Partial Annotation*

1 Introduction

Sequence labeling has many applications in natural language processing such as part-of-speech tagging, shallow parsing, semantic role labeling. Sequence labeling is a kind of the classification tasks. A classifier, or a tagger, will predict the output of each token in the given sequence. Sequence labeling is not trivial due to the dependency among the output labels in the sequence. The dependency also makes the annotation difficult, and is also the reason of high annotation cost.

We can divide a sequence into smaller subsequences consisting of tokens, and label each subsequence. In many cases, labeling a subsequence is easier than labeling the whole sequence. For example, we can exploit the keyword link in Wikipedia text for word boundaries. (Culotta and McCallum 2005) proposed an annotation style for information extraction (IE), which allows users to partially label the entity in the document. If we can train a tagger using partially labeled sequences instead of the fully labeled sequences, we can reduce the human annotation effort.

There are several approaches to train a tagger using partially labeled sequences. Suppose that we have a tagger, we can give labels to the unlabeled parts using the model prediction. However, the prediction is not as accurate as the manual annotation. Moreover, adding mislabeled tokens to the training set usually decreases the accuracy of the tagger. Instead, we propose to automatically estimate the label of unlabeled tokens without explicitly labeling them.

[†] Department of Computational Intelligence and Systems Science, Tokyo Institute of Technology
[††] Precision and Intelligence Laboratory, Tokyo Institute of Technology

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

The taggers in early iterations are usually not accurate, and may incorrectly predict the output labels. These implicitly incorrectly labeled tokens may be corrected when the taggers become more accurate in later iterations.

Active learning is another approach to reduce the annotation cost from the conventional passive learning by elaborately selecting the informative samples for training. The conventional active learning treats a sequence as a sample and search for the informative sequence. We argue that we are required to label some unnecessary tokens using this sequence sampling strategy. Since we propose to partially label a sequence, we propose a subsequence sampling strategy which is more efficient than the sequence sampling for our framework.

The organization of the rest of paper is as follows. We start with the discussion on related work to the proposed framework in section 2. Section 3 is devoted to the parameter estimation of CRFs. In section 4, we describe our framework in detail. Section 5 contains the experiments, discussion and analysis of the result. Finally, we conclude our contribution and discuss the future work in section 6.

2 Related work

Several algorithms are proposed to solve the sequence labeling task such as Hidden Markov Models (HMMs) (Rabiner 1989), Conditional Random Fields (CRFs) (Lafferty et al. 2001), Perceptron, Support Vector Machine (SVM) (Kudo and Matsumoto 2001), and ensemble approaches which combine the result of two or more approaches together (Nguyen and Guo 2007). Each of the mentioned algorithms is primarily proposed as a supervised learning algorithm, whose performance relies on a large amount of labeled data. The cost of data labeling is the main disadvantage of supervised learning, especially in the task such as sequence labeling whose data labeling is very expensive.

Semi-supervised learning is a learning paradigm which can benefit from both the highly-accurate-but-costly labeled data, and noisy-but-cheap unlabeled data. On one hand, we can boost the accuracy of the supervised system using additional large amount of unlabeled data (Ando and Zhang 2005; Suzuki and Isozaki 2008). On the other hand, we can reduce the annotation cost of a new corpus by labeling only a small part of the corpus, while maintaining the comparable accuracy to the supervised approaches which requires full annotation (Culotta and McCallum 2005).

There are several approaches in the semi-supervised learning paradigm. Self-training and transductive SVM (Joachims 1999) are examples of semi-supervised learning which assume that

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

the unlabeled data has the same structure as the labeled data. Hence, we can propagate the knowledge from the labeled set to the unlabeled set. Self-taught learning (Raina et al. 2007) starts learning the general concept from unlabeled data, and refine the concept to the specific task using labeled data. (Suzuki and Isozaki 2008) separated the model parameters for labeled and unlabeled data, and estimate both types of parameters simultaneously. (Dasgupta and Ng 2009) proposed to label only ambiguous samples and automatically label easy samples.

Using partially labeled data is another kind of semi-supervised learning approaches. Partially labeled data is obtained easier than fully labeled data in many situations. We can train a classifier on partially labeled sequences by predicting the labels of unlabeled tokens (Tomanek and Hahn 2009), propagating the label information from labeled to unlabeled tokens (Culotta and McCallum 2005), or implicitly estimating the labels in the training process (Bellare and McCallum 2007; Tsuboi et al. 2008). Another approach is to simplify the training into the point-wise learning, by neglecting the dependencies among output labels (Neubig and Mori 2010).

Apart from the semi-supervised learning, active learning paradigm is proposed to reduce the annotation cost from the passive learning by elaborately selecting highly informative samples for the training. (Settles and Craven 2008) have compared several active learning strategies for fully supervised sequence labeling tasks . Active learning is also used in combination with semi-supervised learning methods in order to reduce the annotation cost (Tomanek and Hahn 2009; Dasgupta and Ng 2009).

3 Conditional random fields (CRFs)

3.1 Conventional conditional random fields

The objective of the sequence labeling task is to find an output label sequence $\mathbf{y} \in \mathbf{Y} : (y_1, ..., y_T)$ of the given input sequence $\mathbf{x} \in \mathbf{X} : (x_1, ..., x_T)$. $\mathbf{X}$ and $\mathbf{Y}$ is the set of all possible input and the corresponding output sequences, respectively. T is the length of the sequence. We employ the conditional random fields (Lafferty et al. 2001) to model the probability of $\mathbf{y}$, given $\mathbf{x}$ by

$$P_\theta(\mathbf{y}|\mathbf{x}) = \frac{e^{\theta \cdot \mathbf{\Phi}(\mathbf{x},\mathbf{y})}}{Z_{\theta,\mathbf{x},\mathbf{Y}}} \ . \tag{1}$$

We have discussed that an output label, y, depends on both the input and the output tokens. Hence, our feature function $\Phi(\mathbf{x}, \mathbf{y}) : \mathbf{X} \times \mathbf{Y}$ is a mapping from both input and output to a real value. Suppose that we have d features, and $\mathbf{\Phi} = (\Phi_1, ..., \Phi_d)$ is our feature vector. θ is the

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

model parameters, or the weight vector, estimated in the training process. Finally, $Z_{\theta,\mathbf{x},\mathbf{Y}}$ is the normalizing factor defined by

$$Z_{\theta,\mathbf{x},\mathbf{Y}} = \sum_{\mathbf{y}\in\mathbf{Y}} e^{\theta\cdot\mathbf{\Phi}(\mathbf{x},\mathbf{y})} \ .$$

Let α_j be the probability of the prefix sequence to position j, called the forward probability:

$$\alpha_j(y') = \sum_{y''} (\alpha_{j-1}(y'')p_t(y'',y')p_s(y')) \ ,$$
$$\alpha_1(y') = p_s(y') \ .$$

Let β_j be the probability of the suffix sequence from position j, called the backward probability:

$$\beta_j(y') = \sum_{y''} (p_t(y',y'')p_s(y'')\beta_{j+1}(y'')) \ ,$$
$$\beta_T(y') = 1 \ .$$

$p_t(y',y'')$ is the transition probability from label y' to label y''. $p_s(y')$ is the output probability of label y'. After calculating all α_j, β_j, we can efficiently compute $Z_{\theta,\mathbf{x},\mathbf{Y}}$ by

$$Z_{\theta,\mathbf{x},\mathbf{Y}} = \sum_{y'\in Y_1} \alpha_1(y')\cdot\beta_1(y') \ ,$$

where Y_1 is all possible labels of y_1.

In order to estimate θ, we will maximize the following log likelihood function:

$$LL(\theta) = \sum_{n=1}^{N} ln(P_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})) \ . \tag{2}$$

N is the number of sequences in the training set. $\mathbf{x}^{(n)}$ and $\mathbf{y}^{(n)}$ are the n^{th} input sequence and the Viterbi output sequence of $\mathbf{x}^{(n)}$, respectively. We can apply standard optimization techniques such as L-BFGS (Liu and Nocedal 1989) or SGD (Vishwanathan et al. 2006) to the objective function in (2).

3.2 Conditional random fields for partially labeled sequences

The conventional method described in section 3.1 requires the whole sequence to be labeled in order to compute the sequence probability using (1). We need to re-define the sequence probability if the sequence is only partially labeled. Given a partially or ambiguously labeled sequence $\mathbf{L}$, let $\mathbf{Y_L}$ be the set of all possible output sequences consistent with $\mathbf{L}$. We follow

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

(Bellare and McCallum 2007; Tsuboi et al. 2008) to estimate the probability of $\mathbf{Y_L}$ given $\mathbf{x}$ by

$$P_\theta(\mathbf{Y_L}|\mathbf{x}) = \sum_{\mathbf{y}\in\mathbf{Y_L}} P_\theta(\mathbf{y}|\mathbf{x}) \ . \tag{3}$$

Using (3), the log likelihood in (2) is modified to

$$\begin{aligned} LL(\theta) &= \sum_{n=1}^{N} \ln P_\theta(\mathbf{Y_L}^{(n)}|\mathbf{x}^{(n)}) \\ &= \sum_{n=1}^{N} \left(\ln \frac{\sum_{\mathbf{y}\in\mathbf{Y_L}^{(n)}} e^{\theta\cdot\mathbf{\Phi}(\mathbf{x^{(n)}},\mathbf{y})}}{\sum_{\mathbf{y}\in\mathbf{Y}} e^{\theta\cdot\mathbf{\Phi}(\mathbf{x}^{(n)},\mathbf{y})}} \right) \\ &= \sum_{n=1}^{N} \left(\ln \sum_{\mathbf{y}\in\mathbf{Y_L}^{(n)}} (e^{\theta\cdot\mathbf{\Phi}(\mathbf{x^{(n)}},\mathbf{y})}) - \ln \sum_{\mathbf{y}\in\mathbf{Y}} (e^{\theta\cdot\mathbf{\Phi}(\mathbf{x^{(n)}},\mathbf{y})}) \right) \\ &= \sum_{n=1}^{N} (\ln Z_{\theta,\mathbf{x}^{(n)},\mathbf{Y_L}^{(n)}} - \ln Z_{\theta,\mathbf{x}^{(n)},\mathbf{Y}}) \ . \end{aligned} \tag{4}$$

$\mathbf{Y_L}^{(n)}$ is the set of all possible output sequences of $\mathbf{x}^{(n)}$. $Z_{\theta,\mathbf{x}^{(n)},\mathbf{Y_L}^{(n)}}$ can be computed by the same algorithm used for $Z_{\theta,\mathbf{x},\mathbf{Y}}$. We then apply the standard optimization techniques to (4) as done to (2).

4 Proposed framework

Our active learning framework is shown in Fig. 1. The framework consists of 2 main parts,

$S_t : \{(\mathbf{x}, \mathbf{y_L})\}$ is a set of all training sequences with current annotation at iteration t
S_{sel} is a set of informative tokens
x is an input token
repeat
  $curmodel \leftarrow train(S_t)$ {Training}
  $S_{sel} \leftarrow Q_{\text{tok}}(curmodel, S_t)$ {Selecting q tokens (at most)}
  for $x \in S_{sel}$ **do**
    $S_{t+1} \leftarrow update(S_t, x, label(x))$ {Annotation}
  end for
until $(|S_{sel}| < q)$ and $(\kappa(S_t, S_{t+1}) > 0.9999)$ {Stopping criteria}

Fig. 1 Active learning framework

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

the query strategy and the stopping criteria. We denote our framework as partially labeling with token sampling (*[Proposed]*).

4.1 Query strategy

The key of active learning is the query strategy which should select good samples for training. Firstly, we will define the sample and its informativeness. We have discussed in the introduction that we can partially label a sequence. Since a token is the smallest subsequence unit, we will treat each token as our sample. Then, we will search for informative tokens, in contrast to the conventional active learning which treats the whole sequence as a sample and search for informative sequences.

In each iteration, a tagger can predict the output of each token with different confidence level. We define the prediction confidence of each token by its marginal probability:

$$P_\theta(y_j = y'|\mathbf{x}) = \frac{\alpha_j(y'|\mathbf{x}) \cdot \beta_j(y'|\mathbf{x})}{Z_\theta(\mathbf{x}, \mathbf{Y})} \ , \tag{5}$$

where α and β are the forward and backward probabilities defined in section 3. Since a sequence can have several possible output sequences, the marginal probability of a token depends on the choice of the output sequence. In order to compare the prediction confidence of tokens between sequences, we choose the prediction confidence based on the most likely output sequences defined by the Viterbi sequence:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) \ . \tag{6}$$

When the prediction confidence of a token is less than the predefined threshold, it indicates that the token may contain crucial information which is not previously known to the tagger. Hence, we define such a token to be *informative*, and ask a human annotator to label the token. In each iteration, we search for q tokens which are the most informative, and update their labels with human annotation. Although we search for tokens, we provide the informative tokens with their context, i.e. the whole sequences, to a human annotator since context is necessary for the labeling. A sequence may be selected several times. Such a sequence will have several labeled tokens.

We train the tagger in each iteration using CRFs defined in section 3. The training set contains both fully labeled, partially labeled, and unlabeled sequences. However, our CRFs parameter estimation cannot benefit from the unlabeled sequences. On the contrary, adding unlabeled sequences will slow the optimization process. Hence, we just train the tagger using the fully and partially labeled sequences.

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

4.2 Stopping criteria

Since we search for informative tokens, we can stop the learning if we cannot find more informative tokens in the training set. However, adding few new labeled tokens to the training set does not help improving the F_1 but just wasting the training time of the tagger.

In the query process, we use the tagger to predict the output of all tokens in the whole training set. Let the predicted output of the whole training set at iteration t be S_t. If there are few differences between the predicted labels of the consecutive iterations (i.e. S_t and S_{t+1}), we can stop the learning. We measure the difference of predicted sets using Kappa statistics (Bloodgood and Vijay-Shanker 2009). From our experiment, standard κ threshold at 0.99 makes the learning stop so early that the system cannot achieve high F_1. In this paper, we regard the datasets as similar when $\kappa(S_t, S_{t+1}) > 0.9999$. However, the difference of the predicted training sets may be small if the query size is small. We also add the condition that the number of informative tokens must be less than the query size in order to stop the learning.

5 Experiments and results

5.1 Data

We conducted experiments on chunking task using CoNLL-2000 dataset (Tjong Kim Sang and Buchholz 2000), and named entity recognition task using CoNLL-2003 dataset (Tjong Kim Sang and De Meulder 2003). We summarize the statistics of each dataset in Table 1.

Both tasks are formulated as a sequence labeling task. A token in a sequence refers to a word, while the sequence itself refers to a sentence. The label of each token is either *B*egin-Chunk, *I*nside-Chunk or *O*utside chunk, e.g. *B*-NP is the beginning token of an NP chunk. For the named

Table 1 Data statistics.

Dataset	**#Sequences**	**#Tokens**	**#Chunk types**	**#Chunks**
CoNLL2000				
training	8936	211727	22	106978
test	2012	47377	19	23852
CoNLL2000 NP-Chunking				
training	8936	211727	3	55081
test	2012	47377	3	12422
CoNLL2003				
training	14987	204567	8	23499
test	3684	46666	8	5648

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

entity recognition task, a chunk is a named entity chunk, e.g. PERSON, ORGANIZATION.

We analyzed the result using CoNLL-2000 chunking dataset but simplified the task to noun phrase (NP) chunking. For the NP chunking task, we have 3 types of label, *B*-NP, *I*-NP, and *O*. Fig. 2 shows an example of NP chunking formulated as a sequence labeling task.

The feature templates for each data set are shown in Table 2. Each template is shown in a bracket. w_i and lw_i are the word and its lowercase at position i in a sentence. p_i is the part-of-speech (POS) of w_i, which is provided in the dataset. c_i is a chunk type, e.g. NP chunk. wtp is the word type described in Table 3. $pw[c]_i$ and $sw[c]_i$ are c-character prefix and suffix of word w_i, e.g. 3-character prefix of the word *American* is *Ame*. y_i is an output label of w_i.

$$\mathbf{y} = (y_1, ..., y_T)$$

y:	B-NP	I-NP	O	O	B-NP	I-NP	I-NP	O	B-NP	O
x:	Mr.	Meador	had	been	executive	vice	president	of	Balcor	.

$$\mathbf{x} = (x_1, ..., x_T)$$

Fig. 2 Sequence labeling model

Table 2 Feature set.

Feature types	Templates	
	CoNLL2000	CoNLL2003
Word	$[w_{i-2}], [w_{i-1}], [w_i], [w_{i+1}], [w_{i+2}]$,	$[w_{i-2}], [w_{i-1}], [w_i], [w_{i+1}], [w_{i+2}]$,
	$[w_{i-1}, w_i], [w_i, w_{i+1}]$	-
PrevWord	-	$[w_{i-4}, w_{i-3}, w_{i-2}, w_{i-1}]$
NextWord	-	$[w_{i+1}, w_{i+2}, w_{i+3}, w_{i+4}]$
Lowercase word	-	$[lw_{i-2}], [lw_{i-1}], [lw_i], [lw_{i+1}], [lw_{i+2}]$,
	-	$[lw_{i-2}, lw_{i-1}], [lw_{i-1}, lw_i], [lw_i, lw_{i+1}], [lw_{i+1}, lw_{i+2}]$,
Word type	-	$[wtp_{i-2}], [wtp_{i-1}], [wtp_i], [wtp_{i+1}], [wtp_{i+2}]$
POS	$[p_{i-2}], [p_{i-1}], [p_i], [p_{i+1}], [p_{i+2}]$,	$[p_{i-2}], [p_{i-1}], [p_i], [p_{i+1}], [p_{i+2}]$,
	$[p_{i-2}, p_{i-1}], [p_{i-1}, p_i], [p_i, p_{i+1}], [p_{i+1}, p_{i+2}]$	$[p_{i-2}, p_{i-1}], [p_{i-1}, p_i], [p_i, p_{i+1}], [p_{i+1}, p_{i+2}]$
	$[p_{i-2}, p_{i-1}, p_i], [p_{i-1}, p_i, p_{i+1}], [p_i, p_{i+1}, p_{i+2}]$	$[p_{i-1}, p_i, p_{i+1}]$
Chunk type	-	$[c_{i-2}], [c_{i-1}], [c_i], [c_{i+1}], [c_{i+2}]$,
	-	$[c_{i-2}, c_{i-1}], [c_{i-1}, c_i], [c_i, c_{i+1}], [c_{i+1}, c_{i+2}]$
	-	$[c_{i-1}, c_i, c_{i+1}]$
Prefix	-	$[pw2_{i-1}], [pw2_i], [pw2_{i+1}], [pw3_{i-1}], [pw3_i], [pw3_{i+1}]$
Suffix	-	$[sw2_{i-1}], [sw2_i], [sw2_{i+1}], [sw3_{i-1}], [sw3_i], [sw3_{i+1}]$
Transition	$[y_{i-1}]$	$[y_{i-1}]$

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Table 3 Word types and examples.

Description	Examples
starts with capital letter	Confidence, September, But
all capital letters	PLC, GNP, A
contains both uppercase and lowercase letters	anti-American, Chancellor, Lawson
single digit	1, 2, 3
contains only digits	16, 1988, 190
contains at least two periods	U.K., A.P., F.S.B
ends with a period	Ala., p.m., vs.
contains a dash	year-ago, 1-800-453-9000, 10-fold
single character	A, a, b
contains punctuation	US$, #, ’,’
contains quotation	’s, I’m, n’t

5.2 Evalution

Our goal is to achieve the supervised F_1 at the smallest annotation cost. F_1 is calculated using CoNLL evaluation (Tjong Kim Sang and Buchholz 2000). We claim that we achieve the supervised F_1 if the predicted output is not significantly different from the supervised prediction in McNemar test (Gillick and Cox 1989). We measure the annotation cost in 2 aspects; the number of manually labeled tokens, and the time required in the whole training (excluding the human annotation)[1].

5.3 Initial set

The conventional active learning usually validates the result among several random initial sets since the training is sensitive to the initial set. We compared the initial set of short sequences, long sequences and random sequences in Fig. 3. Each initial set consists of 50 sequences. The confidence threshold is set to 0.99. All tokens with the prediction confidence less than the threshold are informative. The top 500 informative tokens are selected and manually labeled in each iteration of active learning. The result in Fig. 3 shows that the initial set with long sequences clearly converges with fewer labeled tokens than the other sets, while requires similar training time. We discuss that long sequences contain more information than short sequences. The model trained on these long sequences is more accurate and more reliable than the model trained on short sequences. Although a long and complex sequence may require long annotation

[1] We conducted all experiments on 3.00 GHz Xeon E5450 server using 4 cpus. Our CRFs tagger was implemented in C.

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

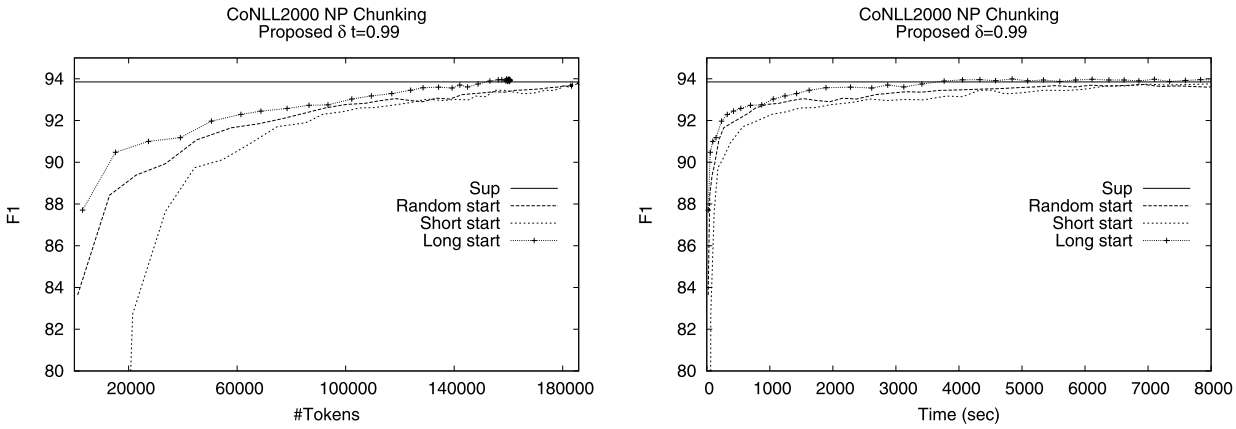

Fig. 3 F_1 of each initial set and the annotation cost

time, the heuristic is applied only in the initial step. The sampling in the other steps is according to the query strategy of active learning, and the whole number of labeled tokens of using long initial set is less than when using the other initial sets. Hence, we will use 50 longest sequences as our initial set in all of the following experiments. Note that there may be more sophisticated models which are more efficient than the heuristic tested in this work. We leave the task to find the best initial set to future work.

5.4 Baseline systems

In this section, we will compare several iterative and non-iterative baselines.

[Sup]: The first baseline is a simple supervised system trained on the whole training set. The F_1 of *[Sup]* is the upper-bound of all systems.

[Pointwise]: (Neubig and Mori 2010) proposed a point-wise training framework in the domain adaptation task, which neglects the dependencies among the output labels. Since we will compare each method on a single domain, we train a point-wise system in supervised manner. Specifically, we use all features in Table 2 except the transition feature. The supervised point-wise system is the upper bound accuracy of a point-wise active learning system.

[Partial]: The last non-iterative baseline is the system trained on partially labeled data. We calculate the confidence score of each unlabeled tokens using the model trained on the initial set. All tokens with confidence less than δ are manually labeled. The other tokens are left unlabeled.

The comparison between non-iterative baselines is shown in the first half of Table 4. Neither *[Pointwise]* nor *[Partial]* achieves the supervised F_1. Although the training of *[Pointwise]* is fast, the F_1 of the system is competitive but still significantly worse than the supervised upper bound. The low F_1 achievement of *[Pointwise]* indicates that the dependencies between output labels are

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Table 4 Comparison between baseline systems on NP chunking task. Boldface F_1 indicates that the prediction is not statistically significantly different from the supervised prediction.

Systems	F_1	%Labeled tokens	Training time (sec)
[Sup]	**93.86**	100.00	521
[Pointwise]	92.52	100.00	57
[Partial] $\delta = 0.90$	92.57	7.86	247
[Partial] $\delta = 0.99$	93.43	15.66	287
[FuSAL]	**93.85**	100.00	4019
[Bootstrap]	89.05	1.43	1302
[Tomanek09] $\delta = 0.99$	92.91	5.48	3432
[Wanvarie10] $\delta = 0.99$	**93.79**	7.73	2340

important. Since the upper bound F_1 of the supervised point-wise system is not high, we will not apply the iterative training to point-wise features. *[Partial]* achieves higher F_1 than *[Pointwise]* even though it requires less number of labeled tokens, but its best F_1 is still significantly lower than the supervised upper bound.

The following experiment is the comparison among the iterative baselines. All iterative systems sample 500 sequences with low Viterbi score (defined in (6)) in each iteration. These iterative baselines are sequence-based sampling in contrast to the token-based sampling of the proposed framework. The token in each sampled sequence is informative if its score (defined in (5)) is less than the confidence threshold, δ. We set δ to 0.99.

[FuSAL]: The first iterative baseline is a fully supervised active learning system, whose all training tokens are manually labeled. *[FuSAL]* becomes almost exactly the same as *[Sup]* after we label the whole training set. The only difference between *[FuSAL]* and *[Sup]* is the sentence ordering in the training set.

[Bootstrap]: Bootstrapping is a semi-supervised learning approach. Only tokens in the initial set are manually labeled. The other tokens are either labeled by the model prediction if their prediction confidence is high, or left unlabeled otherwise.

[Tomanek09]: The second semi-supervised active learning baseline is the system proposed in (Tomanek and Hahn 2009). In each sampled sentence, the low confidence tokens are manually labeled, while the high confidence tokens are automatically labeled by the model prediction.

[Wanvarie10]: The last semi-supervised active learning baseline is proposed in (Wanvarie et al. 2010). The system is similar to *[Tomanek09]* except that the high confidence tokens are left unlabeled instead of being labeled by the model prediction. They implicitly estimate those labels in the training set using the method described in (Tsuboi et al. 2008).

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Figure 4 shows the learning curve of each training strategy. Note that the F_1 of *[Sup]* does not vary since we train the tagger once with the whole training set. We just draw a straight F_1 line of *[Sup]* for reference. Similarly, the number of manually labeled tokens of *[Bootstrap]* is also fixed to the number of tokens in the initial set. However, the F_1 of *[Bootstrap]* varies since we gradually add tokens to the training set. We draw a straight line of final F_1 of *[Bootstrap]* vs. number of labeled tokens.

The bottom line of the bootstrapping system (*[Bootstrap]*) indicates that we can learn little new information from unlabeled tokens. Hence, the F_1 of *[Bootstrap]* is far worse than the other baselines.

Among the three learning curves in the middle of the graph on the left of Fig. 4, the bottom curve of *[FuSAL]* has the slowest learning rate. The F_1 of *[FuSAL]* gradually increased and reached the supervised level when approximately 150 k of tokens are labeled. Due to the limited space, we cut the learning curve to the first 15 k labeled tokens. The other two curves represent learning rate of *[Tomanek09]* and *[Wanvarie10]*. Although *[Tomanek09]* achieves much higher F_1 than *[Bootstrap]* with 4% additional labeled tokens, its best F_1 is still far worse than the supervised F_1. On the other hand, *[Wanvarie10]* achieves slightly higher F_1 than *[Tomanek09]* when similar number of labeled tokens are available, and finally achieves the supervised F_1 after more tokens are labeled. Since *[Wanvarie10]* did not explicitly label high confidence tokens as done in *[Tomanek09]*, the prediction errors in early iterations are recovered when the tagger becomes more precise in later iterations. As a result, *[Wanvarie10]* achieves higher F_1 than *[Tomanek09]*. The error recovery is also the reason that the F_1 of active sampling is superior to the F_1 of the batch sampling, since the batch sampling relies on the accuracy of the initial model which is usually poor.

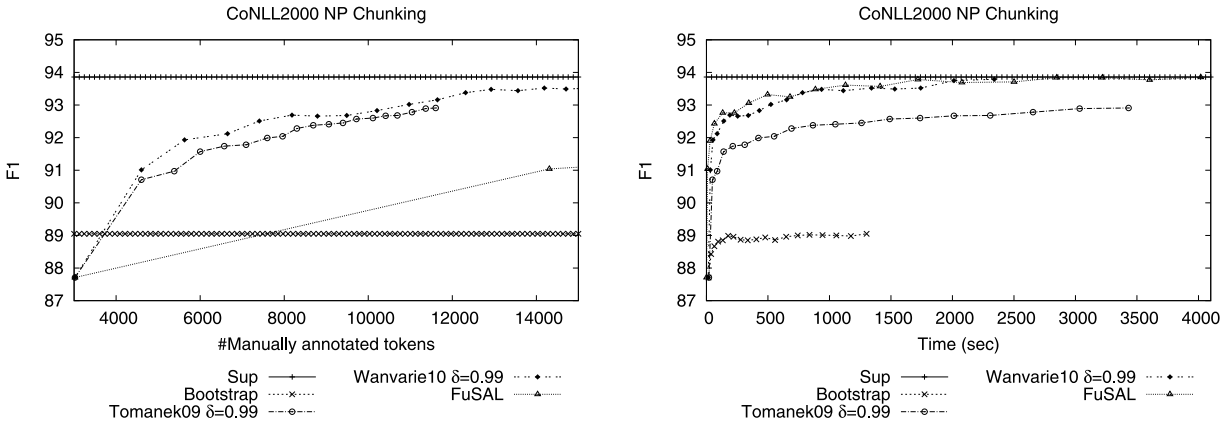

Fig. 4 Comparison among baseline systems

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

The graph on the right of Fig. 4 shows the training time of each baseline. The training with few labeled tokens tends to require less training time. We will discuss on the training time later in section 5.7.

Finally, we compare the achievement of active and batch sampling baselines in Table 4. In the terms of manually labeled tokens, active sampling methods clearly require few labeled tokens and achieve higher F_1. Since the model becomes more certain in late iterations, we need to label few new tokens but still achieve high F_1. However, the iterative methods have a drawback in the training time since we have to re-estimate the model parameters in every iteration.

Due to the limited space, we select only *[Sup]* and *[Wanvarie10]* to compare with the proposed method (*[Proposed]*) in the rest of the experiments.

5.5 Effect of the query size

We can save the annotation cost in terms of labeled tokens by selecting only few highly informative tokens in each iteration. However, a certain number of tokens are necessary to achieve the certain level of F_1. If the query size is small, we have to do several iterations to obtain sufficient number of labeled tokens, and we have to spend more time on the training of the taggers. Since the CRFs training is the most time consuming process in the framework, it is not practical to re-train the taggers several times. The query size setting controls the trade-off between the number of manually labeled tokens and the time required in the training. We compare the query size settings varied from 50, 500, to 1000 sequences per iteration. The confidence threshold is set to 0.99. The result is shown in Fig. 5.

The large query size setting clearly reduces the whole training time, but slightly requires more labeled tokens. Note that we did not count the actual time spent by the annotator. Large

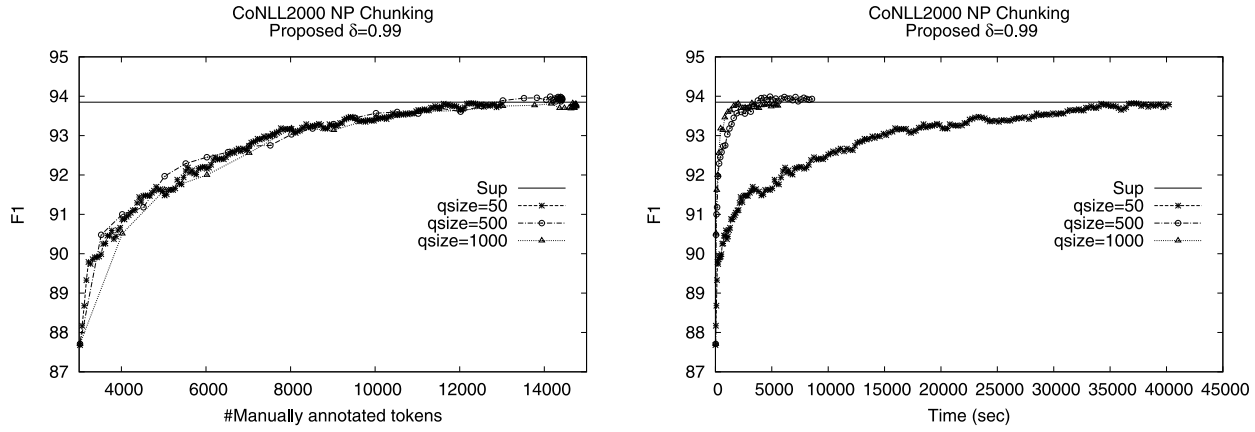

Fig. 5 Query size and the annotation cost

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

query setting usually requires more time for an annotator to label the queried set. Nevertheless, we believe that labeling a single token will not take long time. We set the query size to 500 tokens per iteration in all of the following experiments. In other words, at most 500 sequences are presented to the annotator at each iteration.

5.6 Effect of the confidence threshold

With high confidence threshold settings, the model becomes more reliable with the trade-off of the human annotation cost. From the result in Fig. 6, the system with low threshold, $\delta = 0.60$, requires few manually labeled tokens but stops early before achieving the supervised F_1. We argue that the number of labeled tokens is insufficient, and the model trained on this training set produces many errors. In contrast, with the high threshold, $\delta = 0.90$ and 0.99, the systems require more labeled tokens than the system with low threshold setting, but achieve the supervised F_1 level.

5.7 F_1 and the annotation cost

We compare the annotation cost and F_1 of the proposed framework and the baselines in Fig. 7 and Table 5. With $\delta = 0.99$, *[Proposed]* requires slightly fewer numbers of manually labeled tokens than *[Wanvarie10]*, but achieves the supervised F_1. With smaller threshold, $\delta = 0.90$, *[Proposed]* requires much less number of manually tokens, and still achieves the supervised F_1. Although the achieved F_1 of the three systems are not the same, the difference is not statistically significant. *[Proposed]* with $\delta = 0.90$ requires comparable computational time to *[FuSAL]*, but requires much less labeled tokens in order to achieve the supervised F_1.

The training time of a single iteration depends on two factors; the data size and the complexity

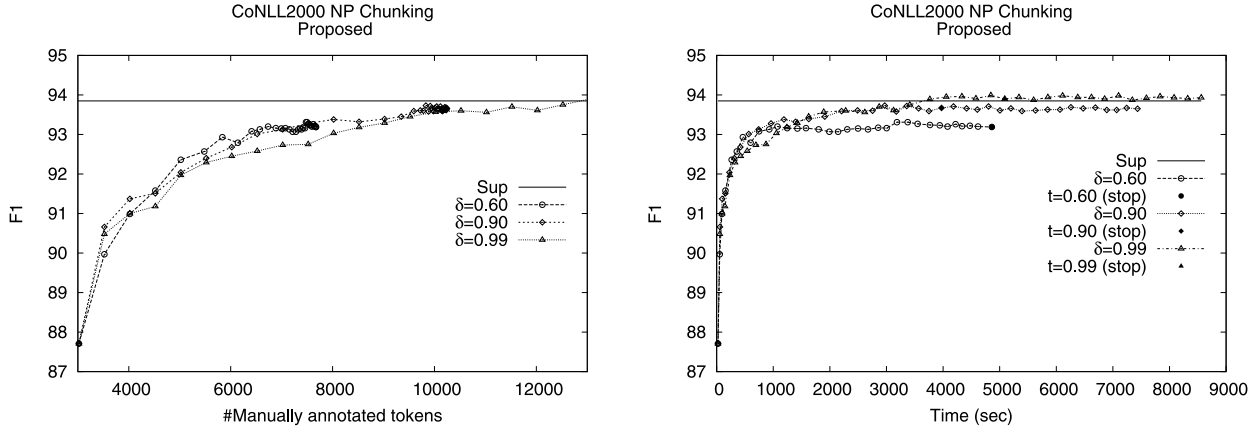

Fig. 6 Effect of threshold setting

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

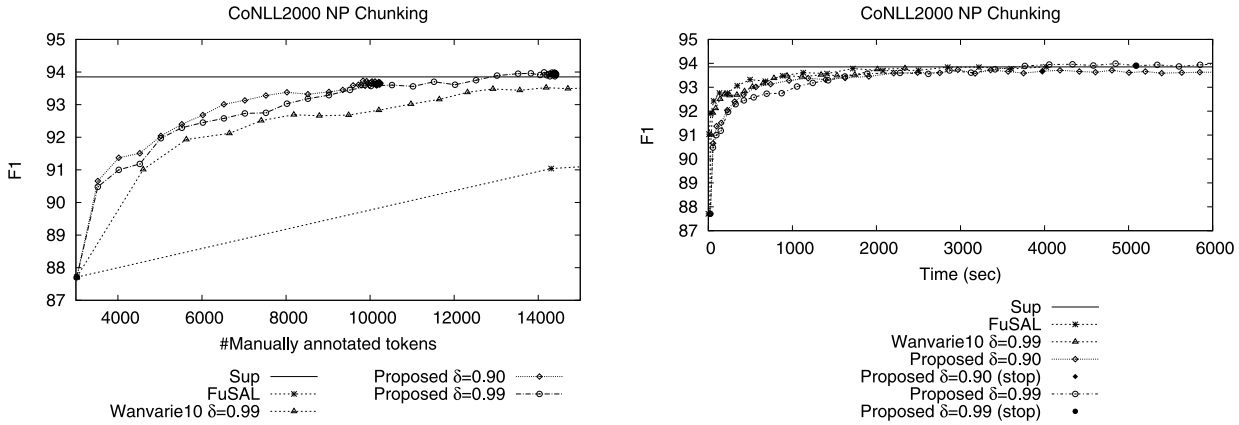

Fig. 7 Comparison between *Proposed* and baseline systems

Table 5 Comparison between *[Proposed]* and baseline systems on NP chunking task. Boldface F_1 indicates that the prediction is not statistically significantly different from the supervised prediction.

Systems	F_1	%Labeled tokens	%Sequences	Training time (sec)
[Sup]	**93.86**	100.00	100.00	521
[FuSAL]	**93.85**	100.00	100.00	4019
[Wanvarie10] $\delta = 0.99$	**93.79**	7.73	100.00	2340
[Proposed] $\delta = 0.90$	**93.67**	4.73	51.75	3973
[Proposed] $\delta = 0.99$	**93.90**	6.71	68.46	5090

of label distribution. The data size is the number of tokens in the training set, both labeled and unlabeled tokens. The data size does matter since the optimization of CRFs will verify the current parameter settings on each sample in the dataset. Hence, a large training set requires more training time than a small training set. With the same data size, the training set with complex label distribution tends to require more time than the simple training set. The complexity of the distribution depends on the number of manually labeled tokens. Therefore, the training set with fully labeled sequences requires more training time than the partially labeled training set.

In *[Proposed]*, we have to label some sequences more than once, i.e. labeling more than a token in the sequence. With $\delta = 0.90$, 51.75% of the training sequences are informative. Among the informative sequences, 61.57% of sequences have only a single labeled token. The other sequences are uninformative, and never be selected. We conclude that our strategy can efficiently select informative tokens to reduce the annotation cost, while maintaining the supervised F_1.

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

5.8 Result on chunking and named entity recognition

Finally, we conducted experiments on the chunking and the named entity recognition tasks. Fig. 8 and Fig. 9 show that the proposed algorithm consistently outperforms all baselines in both datasets. The F_1 of the proposed framework also reaches the supervised F_1 with less manually labeled tokens. Although the final F_1 of *[Proposed]* with threshold at 0.99 is slightly higher than the F_1 of the same method with threshold at 0.90, the prediction results from both systems are not statistically significantly different from the supervised F_1.

Table 6 summarizes the annotation cost and F_1 achievement of the proposed framework and the baseline systems. We can reduce the annotation cost in terms of labeled tokens to only 6.98% in CoNLL chunking, and 7.01% in CoNLL named entity recognition task. Similar to the result on NP chunking task, not all of the sequences are informative, and half of these informative sequences have only a single labeled token. The result confirms our intuition that we do not need

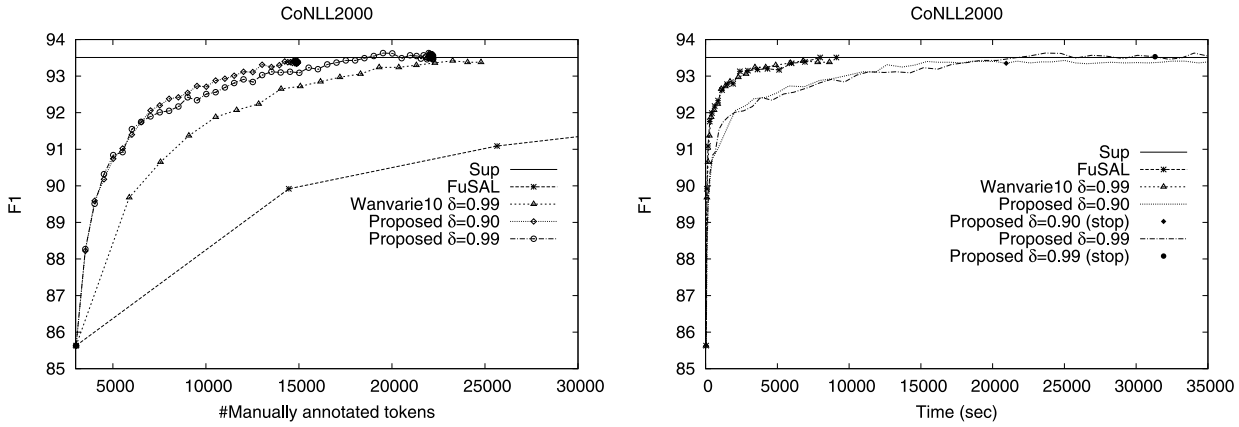

Fig. 8 Result on CoNLL chunking task

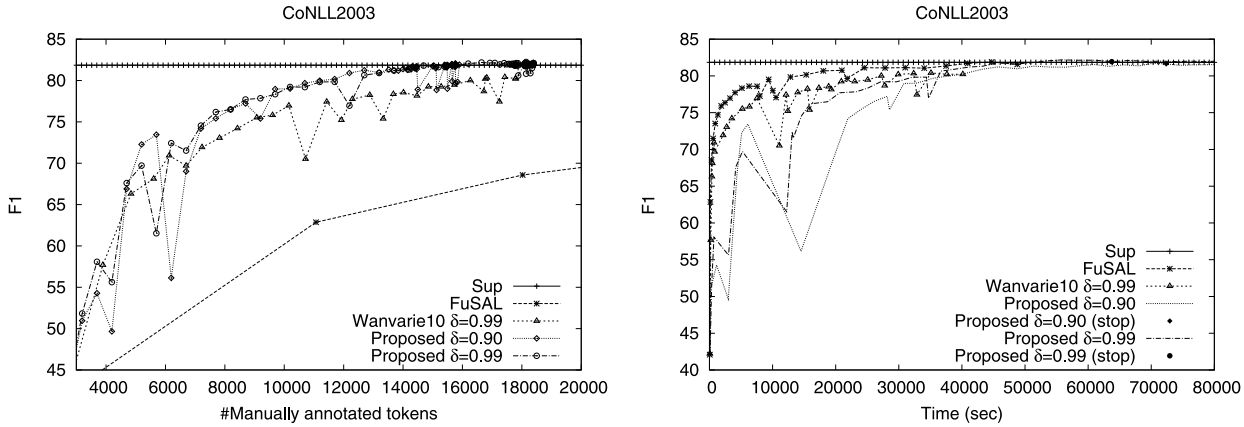

Fig. 9 Result on CoNLL Named Entity Recognition task

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Table 6 Comparison between *[Proposed]* and baseline systems on CoNLL2000 and CoNLL2003 datasets. Boldface F_1 indicates that the prediction is not statistically significantly different from the supervised prediction.

Systems	F_1	%Labeled tokens	%Sequences	Training time (sec)
CoNLL2000				
[Sup]	**93.51**	100.00	100.00	1323
[FuSAL]	**93.51**	100.00	100.00	9111
[Wanvarie10] $\delta = 0.99$	**93.38**	11.71	100.00	8629
[Proposed] $\delta = 0.90$	**93.35**	6.98	69.64	20945
[Proposed] $\delta = 0.99$	**93.53**	10.31	69.76	31310
CoNLL2003				
[Sup]	**81.85**	100.00	100.00	4547
[FuSAL]	**81.85**	100.00	100.00	52698
[Wanvarie10] $\delta = 0.99$	80.23	8.70	100.00	37630
[Proposed] $\delta = 0.90$	**81.64**	7.01	48.98	72378
[Proposed] $\delta = 0.99$	**81.95**	8.44	56.22	63738

to label the whole sequences. However, the proposed framework needs twice more training time than *[FuSAL]* systems. In contrast to the result in NP chunking task, chunking and named entity recognition tasks have much more types of labels, which makes the label distribution complex. Thus, both tasks require more computational time than NP chunking.

6 Conclusion and future work

We have proposed a semi-supervised active learning framework which requires less manually labeled tokens to achieve the supervised F_1. The key of our framework is the token sampling and labeling which can both reduce the annotation cost and recover the prediction errors from previous iterations.

Since the actual annotation cost of all tokens may not be equal, we can directly incorporate a more realistic cost model such as (Tomanek et al. 2010), which can represent the actual human annotation time, to our query strategy. Our framework is not limited to the sequence labeling task. We are planning to extend our work to a general structured prediction task. Moreover, we can also employ other learning algorithms apart from the CRFs tagger. However, we need to re-define the prediction confidence.

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Acknowledgement

We thank participants in the NLP special interest group of IPSJ for valuable comments. The first author is supported by the Higher Educational Strategic Scholarships for Frontier Research Network, Thailand.

Reference

Ando, R. K. and Zhang, T. (2005). "A High-Performance Semi-Supervised Learning Method for Text Chunking." In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 1–9, Morristown, NJ, USA. Association for Computational Linguistics.

Bellare, K. and McCallum, A. (2007). "Learning Extractors from Unlabeled Text Using Relevant Databases." In *Sixth International Workshop on Information Integration on the Web*.

Bloodgood, M. and Vijay-Shanker, K. (2009). "A Method for Stopping Active Learning Based on Stabilizing Predictions and the Need for User-Adjustable Stopping." In *CoNLL '09: Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 39–47, Morristown, NJ, USA. Association for Computational Linguistics.

Culotta, A. and McCallum, A. (2005). "Reducing Labeling Effort for Structured Prediction Tasks." In *AAAI'05: Proceedings of the 20th national conference on Artificial intelligence*, pp. 746–751. AAAI Press.

Dasgupta, S. and Ng, V. (2009). "Mine the Easy, Classify the Hard: a Semi-Supervised Approach to Automatic Sentiment Classification." In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pp. 701–709, Morristown, NJ, USA. Association for Computational Linguistics.

Gillick, L. and Cox, S. (1989). "Some Statistical Issues in the Comparison of Speech Recognition Algorithms." In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 1989*, pp. 532–535, vol. 1.

Joachims, T. (1999). "Transductive Inference for Text Classification using Support Vector Machines." In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 200–209, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kudo, T. and Matsumoto, Y. (2001). "Chunking with Support Vector Machines." In *NAACL '01: Second meeting of the North American Chapter of the Association for Computational*

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Linguistics on Language technologies 2001, pp. 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Lafferty, J. D. et al. (2001). "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data." In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Liu, D. C. and Nocedal, J. (1989). "On the limited memory BFGS method for large scale optimization." *Math. Program.*, **45** (3), pp. 503–528.

Neubig, G. and Mori, S. (2010). "Word-based Partial Annotation for Efficient Corpus Construction." In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D. (Eds.), *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Nguyen, N. and Guo, Y. (2007). "Comparisons of Sequence Labeling Algorithms and Extensions." In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pp. 681–688, New York, NY, USA. ACM.

Rabiner, L. R. (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." In *Proceedings of the IEEE*, pp. 257–286.

Raina, R. et al. (2007). "Self-Taught Learning: Transfer Learning from Unlabeled Data." In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pp. 759–766, New York, NY, USA. ACM.

Settles, B. and Craven, M. (2008). "An Analysis of Active Learning Strategies for Sequence Labeling Tasks." In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1070–1079, Morristown, NJ, USA. Association for Computational Linguistics.

Suzuki, J. and Isozaki, H. (2008). "Semi-Supervised Sequential Labeling and Segmentation Using Giga-Word Scale Unlabeled Data." In *Proceedings of ACL-08: HLT*, pp. 665–673, Columbus, Ohio. Association for Computational Linguistics.

Tjong Kim Sang, E. F. and Buchholz, S. (2000). "Introduction to the CoNLL-2000 Shared Task: Chunking." In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning*, pp. 127–132, Morristown, NJ, USA. Association for Computational Linguistics.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). "Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition." In Daelemans, W. and Osborne,

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

M. (Eds.), *Proceedings of CoNLL-2003*, pp. 142–147. Edmonton, Canada.

Tomanek, K. et al. (2010). "A Cognitive Cost Model of Annotations Based on Eye-Tracking Data." In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 1158–1167, Uppsala, Sweden. Association for Computational Linguistics.

Tomanek, K. and Hahn, U. (2009). "Semi-Supervised Active Learning for Sequence Labeling." In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 1039–1047, Suntec, Singapore. Association for Computational Linguistics.

Tsuboi, Y. et al. (2008). "Training Conditional Random Fields using Incomplete Annotations." In *Proceedings of the 22nd International Conference on Computational Linguistics—Volume 1*, COLING '08, pp. 897–904, Morristown, NJ, USA. Association for Computational Linguistics.

Vishwanathan, S. V. N. et al. (2006). "Accelerated Training of Conditional Random Fields with Stochastic Gradient Methods." In *ICML '06: Proceedings of the 23rd International Conference on Machine Learning*, pp. 969–976, New York, NY, USA. ACM.

Wanvarie, D. et al. (2010). "Active Learning with Partially Annotated Sequence." Tech. rep., Special Interest Group of Natural Language Processing, Information Processing of Japan.

Dittaya Wanvarie: received the B. Eng from Chulalongkorn University, Thailand in 2005. Currently, she is a Ph.D. candidate at Tokyo Institute of Technology. Her research interests include morphological analysis, syntactic analysis, and machine learning.

Hiroya Takamura: was born in 1974. He received B.E. and M.E. from the University of Tokyo in 1997 and 2000 respectively (in 1999 he was a research student at Technische Universitaet von Wien). He received Dr. Eng. from Nara Institute of Science and Technology in 2003. He was an assistant professor at Tokyo Institute of Technology from 2003 to 2010. He is currently an associate professor at Tokyo Institute of Technology. His current research interest is computational linguistics. He is a member of the Information Processing Society of Japan and the Association for Computational Linguistics.

Manabu Okumura: was born in 1962. He received B. E., M. E. and Dr. Eng. from Tokyo Institute of Technology in 1984, 1986 and 1989 respectively. He was an assistant at the Department of Computer Science, Tokyo Institute of

Information and Media Technologies 6(3) : 680–700 (2011)
reprinted from: Journal of Natural Language Processing 18(2) : 153–173 (2011)

Technology from 1989 to 1992, and an associate professor at the School of Information Science, Japan Advanced Institute of Science and Technology from 1992 to 2000. He was also a visiting associate professor at the Department of Computer Science, University of Toronto from 1997 to 1998. He is currently a professor at Precision and Intelligence Laboratory, Tokyo Institute of Technology. His current research interests include natural language processing, especially automatic text summarization, computer assisted language learning, and text data mining. He is a member of the Information Processing Society of Japan, the Japanese Society for Artificial Intelligence, the American Association for Artificial Intelligence, the Association for Computational Linguistics, and the Japanese Cognitive Science Society.

(Received October 7, 2010)
(Revised January 11, 2011)
(Rerevised January 11, 2011)
(Accepted February 23, 2011)