A New Loss Function with "Markov Property" for Information Extraction

Yuta Tsuboi Hisashi Kashima*

Abstract

We propose a new loss function for the discriminative learning of Markov random fields, which is an intermediate loss function between sequential loss and pointwise loss. We show this loss function has "Markov property", that is, the importance of correct labeling for a particular position depends on the numbers of the correct labels around there. This property works to keep local consistencies and is useful for optimizing systems identifying structural segments, such as information extraction systems.

Keywords

Supervised Learning, Structured Output, Conditional Random Fields, Information Extraction, Named Entity Extraction

1 Introduction

Structure labeling problem is one of the most important problems in structured data analysis. This is a generalized supervised classification problem, where the labels for a set of target variables are to be predicted when the labels for a set of observed variables are given. Many real-world tasks are formalized as structure labeling problems, for example, part-of-speech tagging and information extraction tasks such as named entity extraction in natural language processing [23], and protein secondary structure prediction and gene discovery in bioinformatics [11].

In sequence labeling problems, hidden Markov models (HMMs) [23, 11] had been attaining some progress for years. Recently, conditional models such as maximum entropy Markov models (MEMMs) [24] and conditional random fields (CRFs) [19] have been attracting considerable attentions because of their capabilities to allow overlapping features, and their performances overwhelming those of HMMs. Especially, CRFs are considered as one of the state-of-art labelers, which are the discriminative learning of Markov random fields.

One of the reasons for the success of the conditional models is that their objective functions directly aim to raise the prediction accuracy for target variables. HMMs usually learn the joint probability distribution of target variables and observed variables, and utilize it for conditional prediction. This means that this approach tackles a more difficult problem than necessary, and needs more training examples. On the other hand, conditional models such as CRFs and MEMMs directly estimate the conditional probability distribution, and therefore more suitable for labeling tasks.

There are some researches on designing and comparing various loss functions (i.e. objective functions) for labeling problems [1, 15, 4]. Two important classes of the loss functions are sequential loss and pointwise loss [15]. The sequential loss is the original objective function that maximizes the sum of log-likelihoods, and the pointwise loss maximizes the sum of marginal log-likelihoods with target variables fixed at each position. This indicates that the sequential loss aims to correctly predict the whole target variables in a sequence. On the other hand, the pointwise loss aims to correctly predict each target variables as many as possible.

Information extraction is one of the typical applications of labeling problems, whose purpose is to identify semantic segments in sequence, tree, or graph representations of natural language, biological data, and so on. When applying the two loss functions to information extraction tasks, the sequential loss achieves this goal, but we have a possibility of resulting in a bad performance in difficult problems with relatively small data. The pointwise loss does its best, but is not enough to represent the aim to extract segments adequately.

In this paper, we propose *mixed loss*, which is an intermediate loss function between the sequential loss and the pointwise loss, and that has characteristics of both objective functions. It is defined as a linear combination of sequential loss and pointwise loss, and not seemingly be suitable for information extraction. However, in sequence labeling and tree labeling, we can show that the mixed loss has a "Markov property", that is, the importance of correct labeling for a particular position depends on the numbers of the correct labels around there. Therefore, our new loss function is expected to be useful to predict clusters of correct labels.

This paper is organized as follows. In Section 2, we give preliminaries of structure labeling problems, conditional random fields, and losses used for training them. In Section 3, we propose a new loss function

*IBM Research, Tokyo Research Laboratory.

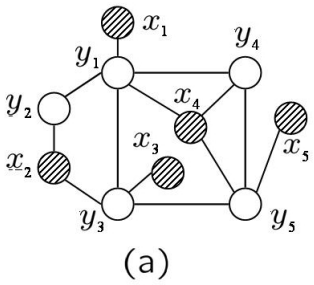


Figure 1: An example of graph structured data to be labeled. The black nodes are observed variables $\boldsymbol{x}$, and the white nodes are target variables $\boldsymbol{y}$.

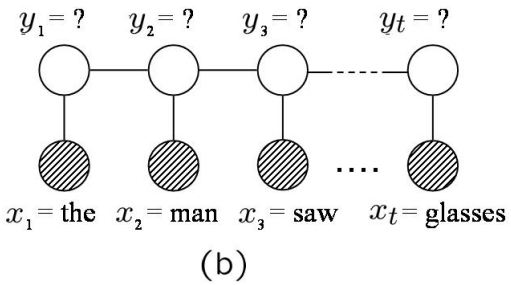


Figure 2: A graph representation of a sequence in part-of-speech tagging tasks. Given $\boldsymbol{x}$ as the sentence "the, man, saw ,$\cdots$, glasses.", $\boldsymbol{y}$ as the part-of-speech tags for the sequence, e.g."DT, NN, VBD, ..., NNS", are to be predicted.

called λ-mixed loss that is more suitable for information extraction tasks. In Section 4 shows the results for the preliminary experiment of a natural language task. In Section 5, we mention related work. Section 6 concludes this paper with discussion about the generalization of our approach.

2 Conditional Models for Labeling Structured Data

Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_T), x_t \in \Sigma_x$ be a set of *observed variables*, and $\boldsymbol{y} = (y_1, y_2, \ldots, y_T), y_t \in \Sigma_y$ be a set of *target variables*[1], where Σ_x and Σ_y are the sets of labels for observed and target variables, respectively. Also, suppose that there is a graph structure among the variables like Figure 1. Figure 2 is an example of such graphs in part-of-speech tagging tasks, x_t indicates the t-th word, and y_t indicates the part-of-speech tag of the t-th word.

In structure labeling problem, given the labels of each observed variables in $\boldsymbol{x}$, we want to assign correct labels for each target variable in $\boldsymbol{y}$. For this goal, we may exploit training data, $E = (e^{(1)}, e^{(2)}, \ldots, e^{(|E|)})$, whose i-th example is $e^{(i)} = (\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$, and $|\boldsymbol{x}^{(i)}| = |\boldsymbol{y}^{(i)}| = T^{(i)}$.

The model of conditional random fields (CRFs) is represented as a form of logistic regression extended to handle multi target variables and multi labels as follows,

$$f(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(\langle \Theta, \Phi(\boldsymbol{x}, \boldsymbol{y}) \rangle)}{\sum_{\tilde{\boldsymbol{y}}} \exp(\langle \Theta, \Phi(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \rangle)},$$

where Θ is the vector of the model parameters, and $\Phi(\boldsymbol{x}, \boldsymbol{y})$ is the feature vector for $(\boldsymbol{x}, \boldsymbol{y})$. Each element ϕ_i of $\Phi(\boldsymbol{x}, \boldsymbol{y})$ is the number of times the i-th feature appears in $(\boldsymbol{x}, \boldsymbol{y})$. Usually, each feature is defined to be a pair of consecutive two variables such as in Figure 3. Especially, a pair of target features (Figure 3 b) are called "transition features". Given the labels for $\boldsymbol{x}$, the labels for $\boldsymbol{y}$ are predicted by $\mathrm{argmax}_{\boldsymbol{y}} f(\boldsymbol{y}|\boldsymbol{x})$.

[1]For simplicity, we suppose that $|\boldsymbol{x}| = |\boldsymbol{y}| = T$, but we can easily extend the results in this paper in a case that $|\boldsymbol{x}| \neq |\boldsymbol{y}|$.

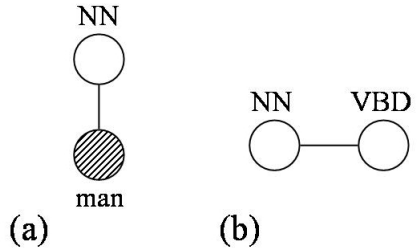


Figure 3: Each feature is defined as a pair of two consecutive variables such as (a) a pair of an observed variable and a target variable, or (b) a pair of two target variables.

The model is trained by finding the optimal parameters that minimize a loss function. In the original CRF model [19], the sum of negative log-likelihoods is used as the loss function.

Definition 1 (Sequential loss function [4]). *Sequential loss function L_1 is defined as*

$$L_1 = -\sum_i \log f(\boldsymbol{y}^{(i)}|\boldsymbol{x}^{(i)}).$$

The model is trained by iterative methods using the following gradient,

$$\frac{\partial L_1}{\partial \Theta} = -\sum_i \Big(\Phi(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}) - \sum_{\tilde{\boldsymbol{y}}} \Phi(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{y}}) f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \Big). \tag{2.1}$$

Note that both the loss function and its derivatives can be calculated efficiently by the dynamic programming techniques without enumerating all the possible target variable paths.

Let us consider the implication of the sequential loss function L_1. This loss function tries to learn the parameters that predict the labels for the whole target variables in a sequence simultaneously, since the likelihood of the set of target variable $\boldsymbol{y}^{(i)}$ for each example is maximized in this loss. However, there is a possibility of resulting in a bad performance in difficult problems with relatively small data since a large negative weight is given to the features whose transitions was never observed in a training set under sequential loss. In addition, there are some tasks, e.g. part-of-speech tagging, where it is enough to correctly predict target variables as many as possible.

Based on those ideas, Kakade et al. [15] proposed another loss function L_0, which is based on the marginal likelihood $\Pr(y_t = y_t^{(i)}|\boldsymbol{x}^{(i)})$ of the label $y_t^{(i)}$ at each position t.

Definition 2 (Pointwise loss function [15]). *Pointwise loss function L_0 is defined as*

$$L_0 = -\sum_i \sum_{t=1}^{T^{(i)}} \log \sum_{\tilde{\boldsymbol{y}}:\tilde{y}_t = y_t^{(i)}} f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}), \tag{2.2}$$

where $\sum_{\boldsymbol{y}:y_t=y_t^{(i)}}$ indicates summation over all possible labels for the target variables with the t-th target variable fixed as $y_t^{(i)} \in \Sigma_y$.

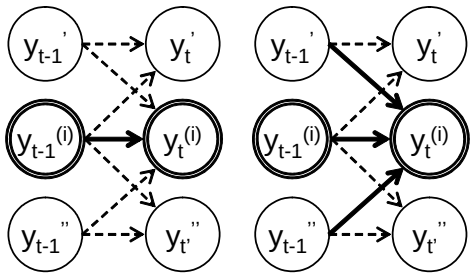
The pointwise loss function L_0 aims to correctly predict each of the target variables as many as possible. and do not care the consistencies among consecutive labels. The pointwise loss function is experimentally shown to be competitive to the sequential loss [1, 15].

The gradient of L_0 is

$$\frac{\partial L_0}{\partial \Theta} = -\sum_i \Big(\sum_{t=1}^{T^{(i)}} \sum_{\tilde{\boldsymbol{y}}:\tilde{y}_t = y_t^{(i)}} \Phi(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{y}}) f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}, y_t^{(i)}) - T^{(i)} \sum_{\tilde{\boldsymbol{y}}} \Phi(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{y}}) f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \Big). \tag{2.3}$$

This loss function and the derivatives can be also calculated efficiently by the dynamic programming techniques.

Under pointwise loss, a positive weight is given to all the features of which the second target label appears in a training data. Figure 4 describes the difference of the way to updating the weights of transition features between sequential loss and pointwise loss.



(a) sequential loss (b) pointwise loss

Figure 4: Weight updates for transition features under sequential loss and pointwise loss: The double circles show the labels which occurred in a training sample and the thin circles show the labels which did not occurred in the data. The bold edges describe rewarded transitions and dashed edges describe punished transitions by each loss function

3 New Loss Function for Information Extraction

Although each of the losses reviewed in the previous section makes sense in each context, we might imagine intermediate situations where it is desired to correctly predict clusters of variables. For example, in information extraction tasks such as named entity recognition and protein secondary structure prediction, we want to find local segments that indicate named entities, alpha helices or beta sheets regions, and they are represented as clusters of labels.

Therefore, a suitable loss function for information extraction is the one with the characteristics of both L_1 and L_0. In other words, we want a loss function with "Markov property", that is, the importance of correct labeling for a particular position depends on the numbers of the correct labels around there. We define the following new loss function L_λ for this purpose.

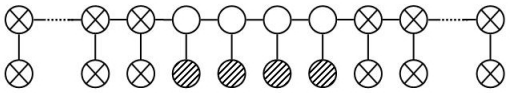
Definition 3 (λ-mixed loss function). *For a given constant $0 \leq \lambda \leq 1$, we call*

$$L_\lambda := \lambda L_1 + (1-\lambda) L_0 \tag{3.4}$$

$$= -\sum_i \Big(\lambda \log f(\boldsymbol{y}^{(i)}|\boldsymbol{x}^{(i)}) + (1-\lambda) \sum_{t=1}^{T^{(i)}} \log \sum_{\tilde{\boldsymbol{y}}:\tilde{y}_t = y_t^{(i)}} f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \Big) \tag{3.5}$$

λ-mixed loss function.

We can see this loss function goes between the sequential loss and the pointwise loss, since L_λ is



(a)

Figure 5: Sequence labeling. $\otimes$ indicates a dummy variable that always takes a special constant label.

identical to L_0 when $\lambda = 0$, and identical to L_1 when $\lambda = 1$.

On first sight, although the new objective function does not seem to enhance local consistencies of labels, we show that it really does in sequence labeling. Let us consider the CRF for sequence labeling, which is described as

$$f(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(\sum_{\tau=-k+1}^{T+k-1} \langle \Theta, \Phi(x_\tau, \boldsymbol{y}_\tau^{\tau+1}) \rangle)}{\sum_{\tilde{\boldsymbol{y}}} \exp(\sum_{\tau=-k+1}^{T+k-1} \langle \Theta, \Phi(x_\tau, \tilde{\boldsymbol{y}}_\tau^{\tau+1}) \rangle)}, \tag{3.6}$$

where $\boldsymbol{x} = (x_{-k+1}, \ldots, x_{T+k-1})$, $\boldsymbol{y} = (y_{-k+1}, \ldots, y_{T+k})$, and $\boldsymbol{y}_\tau^{\tau+1} = (y_\tau, y_{\tau+1})$. The variables in $t < 1$ or $t > T^{(i)}$ are dummy variables which always take a special label σ_0, i.e. $x_t = y_t = \sigma_0$. (See Figure 5.) These dummy variables are for convenience for the following analysis, and Eq. (3.6) is equivalent to the model without them.

Under these assumptions, we define another loss function that has Markov property.

Definition 4 (k-th order Markov loss function). *For a given integer $k > 0$, we call*

$$\begin{aligned} M_k &:= -\sum_i \sum_{t=-k+1}^{T^{(i)}} \log \sum_{\tilde{\boldsymbol{y}}: \tilde{\boldsymbol{y}}_t^{t+k} = \boldsymbol{y}^{(i)}{}_t^{t+k}} f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \\ &= -\sum_i \sum_{t=-k+1}^{T^{(i)}} \log \sum_{\tilde{\boldsymbol{y}}: \tilde{\boldsymbol{y}}_t^{t+k} = \boldsymbol{y}^{(i)}{}_t^{t+k}} \\ &\quad \cdot \frac{\exp(\sum_{\tau=-k+1}^{T^{(i)}+k-1} \langle \Theta, \Phi(x_\tau^{(i)}, \tilde{\boldsymbol{y}}_\tau^{\tau+1}) \rangle)}{\sum_{\tilde{\boldsymbol{y}}} \exp(\sum_{\tau=-k+1}^{T^{(i)}+k-1} \langle \Theta, \Phi(x_\tau^{(i)}, \tilde{\boldsymbol{y}}_\tau^{\tau+1}) \rangle)}, \end{aligned} \tag{3.7}$$

k-th order Markov loss function.

In words, the k-th order Markov loss function is the summation of the loss at position t which depends exclusively on the conditional distribution of next k positions. Note that, in contrast with the marginal likelihood (2.2) fixing only one target variable at a time, M_k fixes $k+1$ consecutive target variables at a time. Therefore, this loss function tries to correctly predict as many chunks of length $k+1$ as possible.

Now, we obtain the following theorem that claims equivalence of mixed loss function and Markov loss function.

Theorem 1. *For any integer $k \geq 0$, let*

$$\lambda = \frac{k}{k+1}. \tag{3.8}$$

Then,

$$\frac{1}{1-\lambda} L_\lambda = M_k.$$

Proof. *The proof is done by simple algebraic substitution. See Appendix for details.* □

This theorem indicates that, for any positive integer $k > 0$, minimization of $\frac{1}{1-\lambda} L_\lambda$ is equivalent to minimization of M_k by choosing λ that satisfies (3.8). Therefore, our new loss function works for correct prediction of labels while keeping local consistencies among them and we can see sequential loss and pointwise loss are the special cases of $k = \infty$ and $k = 0$ of Markov loss, respectively.

From (2.1) and (2.3), the gradients of the proposed loss function are derived as follows:

$$\begin{aligned} \frac{\partial L_\lambda}{\partial \Theta} &= -\sum_i \Big(\lambda \Phi(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}) \\ &\quad + (1-\lambda) \sum_{t=1}^{T^{(i)}} \sum_{\tilde{\boldsymbol{y}}: \tilde{y}_t = y_t^{(i)}} \Phi(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{y}}) f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \\ &\quad - (\lambda + (1-\lambda) T^{(i)}) \sum_{\breve{\boldsymbol{y}}} \Phi(\boldsymbol{x}^{(i)}, \breve{\boldsymbol{y}}) f(\breve{\boldsymbol{y}}|\boldsymbol{x}^{(i)}) \big) \Big). \end{aligned}$$

Figure 6 shows the way to updates weights of transition features under the second order Markov loss (or 2/3-mixed loss) function, and elaborates the equivalence of mixed loss and Markov loss the viewpoint of their derivatives. At each position, the weights of transition features in the chunk of length $k+1$ are updated under the manner of sequential loss and the weights of the transition before the chunk are updated under the manner of pointwise loss. Therefore, the summation of updated weights is mixed value of updated weights by the gradients of sequential loss and pointwise loss at the fixed rate λ.

In above case, since we inherently assumed that k is integer, corresponding λ can take only discrete values. Then, what if k is not integer, i.e. $\lfloor k \rfloor < k < \lceil k \rceil$? Intuitively, L_λ is just an intermediate loss between $M_{\lfloor k \rfloor}$ and $M_{\lceil k \rceil}$. The following two corollaries are easily

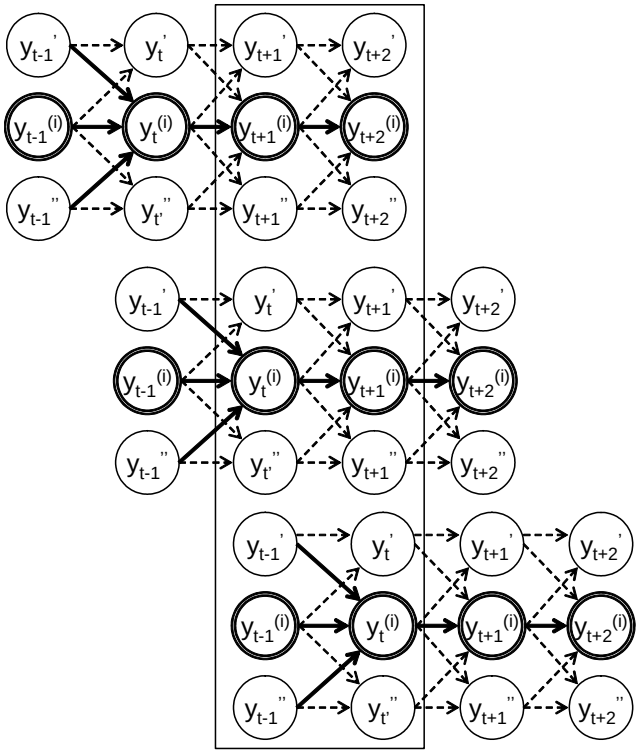


Figure 6: Weight updates of transition features under second order Markov loss function: The updated weight of the transition features in the box is equal to that of the mixed loss of $2/3L_1 + 1/3L_0$.

derived from Theorem 1. The first one just justifies this intuition, and the other gives another interpretation.

Corollary 1. *For any $k \geq 0$, let $\lambda = k/(k+1)$. Then,*

$$\frac{1}{1-\lambda} L_\lambda = (\lceil k \rceil - k) M_{\lfloor k \rfloor} + (k - \lfloor k \rfloor) M_{\lceil k \rceil}. \tag{3.9}$$

Proof. *It is easily verified from Theorem 1 and the definition of mixed loss (3.4) that*

$$\begin{aligned} M_{\lfloor k \rfloor} &= \lfloor k \rfloor L_1 + L_0, \\ M_{\lceil k \rceil} &= \lceil k \rceil L_1 + L_0. \end{aligned}$$

Noting that the following equations hold,

$$\begin{aligned} (\lceil k \rceil - k) + (k - \lfloor k \rfloor) &= 1, \\ (\lceil k \rceil - k)\lfloor k \rfloor + (k - \lfloor k \rfloor)\lceil k \rceil &= k, \end{aligned}$$

the right hand side of (3.9) becomes

$$k L_1 + L_0 = \frac{\lambda}{1-\lambda} L_1 + L_0 = \frac{1}{1-\lambda} L_\lambda.$$

□

Note that, since $0 \leq \lceil k \rceil - k, k - \lfloor k \rfloor \leq 1$ and $(\lceil k \rceil - k) + (k - \lfloor k \rfloor) = 1$, L_λ is just an internally dividing point between $M_{\lfloor k \rfloor}$ and $M_{\lceil k \rceil}$.

From another perspective, this can be understood as a weighted sum of Markov losses with exponentially decaying weights.

Corollary 2. *For any $0 < \lambda < 1$,*

$$\frac{1}{1-\lambda} L_\lambda = (1-\lambda) \sum_{\kappa=0}^{\infty} \lambda^\kappa M_\kappa. \tag{3.10}$$

Proof. *From Theorem 1, the summation in the right hand side becomes,*

$$\sum_{\kappa=0}^{\infty} \lambda^\kappa M_\kappa = \sum_{\kappa=0}^{\infty} \lambda^\kappa (\kappa+1) L_{\frac{\kappa}{\kappa+1}}.$$

Evaluating the infinite series,

$$\sum_{\kappa=0}^{\infty} \lambda^\kappa M_\kappa = \sum_{\kappa=0}^{\infty} \lambda^\kappa (\kappa L_1 + L_0) = \frac{\lambda}{(1-\lambda)^2} L_1 + \frac{1}{1-\lambda} L_0.$$

Substituting this into (3.10) completes the proof.

□

This weighted sum gives large weights to M_κ with small κ, and the weight decays exponentially as κ becomes larger. λ is the parameter controlling the speed of the decay, and small λ means fast decay. This corollary makes a weighted sum of Markov losses corresponds to L_λ with a particular $0 < \lambda < 1$, and we can interpret that the mixed loss is not intended only for a particular length of chunks, but for all length of chunks with weighting them depending on their lengths.

4 Experiment

To test our new loss function, we compared the performances of sequential, pointwise, and mixed loss functions for CRFs on a Named Entity Recognition task.

4.1 Data Set Named Entity Recognition (NER) is a subtask of information extraction which deals with identifying phrases that contain the names of persons, organizations, locations, times and quantities in sentences.

We used the Spanish corpus provided for the shared task of CoNLL2002 on NER [29]. The corpus is composed of a training set, a development set, and a test set, which contain 8322 (264680), 1914 (52849), and 1516 (51487) sentences (tokens), respectively. The shared task concentrates on four types of named entities: *persons*, *locations*, *organizations*, and *names of miscellaneous* entities. The tokens in the corpus are annotated

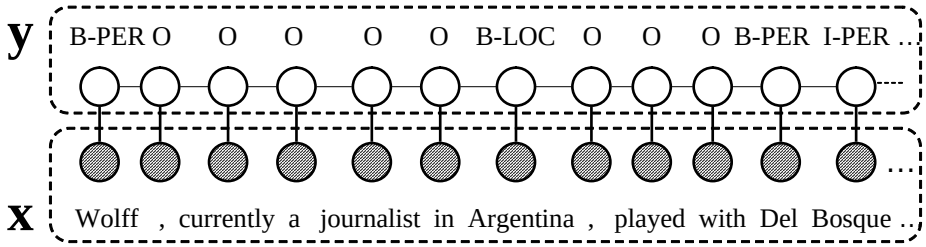


Figure 7: The named entity labels over an example sentence: Words labeled with O are non-named entities. The B–XXX label is used for the first word in a named entity of type XXX and I–XXX is used for all continued other words in named entities of type XXX. PER represents persons and LOC represents locations.

with nine target labels, i.e. $|\Sigma_y| = 9$: the beginning and continuation of the named entities and non-named entities. Figure 7 shows an example of named entity labels encoded in the data. The average phrase length of named entities is 1.74 in the corpus.

4.2 Experimental Setup We conducted two types of experiments: (1) evaluating according to the standard procedure of the shared task of CoNLL 2002, and (2) evaluating the performances varying training data sizes.

For the first experiment, the CRF with each loss function was trained with the training data. The development data was used for tuning the parameter of the CRFs. The parameter is the variance of Gaussian prior, which is a regularization term added to loss functions [6]. When the best parameters were found, CRFs were evaluated on the test data.

For the second experiment, we used the first 100, 200, 300, 600, and 1000 sentences of the train data in training phase. Both the development set and the test set were used in evaluation phase. In this experiment, we did not use regularization terms in objective functions to concentrate on the comparison of the performances of loss functions.

We used pairs of consecutive labels and observed word information as features (See Figure 3). Each word is represented as binary features which are composed of a word occurring and some spelling features. An example of the spelling features is "Is the word ends with period and the current label is Person?". Those spelling features can be overlapped each other. We use those features of not only the current word but also of the words within a fixed window of size 3 (i.e. the previous and next words). The more detailed description of features is described as the S3 features in [4]. For parameter estimation, we employed conjugate gradient descent method [25].

4.3 Results Table 1 shows the results according to the standard procedure of the shared task of CoNLL 2002. The column "point", "k=l", and "seq" represent the results of pointwise loss function, mixed (l-th Markov) loss function, and sequential loss function, respectively. Under the "Markov loss" interpretation of the proposed loss function, we investigated the performances varying the parameter k from 1 to 5. The performances were evaluated according to Precision, Recall, and F1 measure on the test set. Precision is the percentage of named entities found that are correct. Recall is the percentage of found named entities present in the corpus. F1 measure is the harmonic mean of the precision and recall of named entity.

The results indicate that the performances of the proposed loss function are competitive with the conventional loss function. Especially, the $k = 3$ Markov loss slightly performs better than others. Since the average length of named entities is 1.74, this results agree with our intuitions since the size of segment plus two represents proper local consistency to recognize the edges of segment.

Table 2 shows the results of the NER task varying training data sizes. We compared the performances of sequential loss, pointwise loss, and mixed losses varying the parameter k from 1 to 4. The performances were evaluated according to the average F1 measure of development set and test set.

In total, pointwise loss and mixed loss show higher performance than sequential loss, though the difference between pointwise loss and mixed loss was not stable when the size of training data was varied. The CRFs trained on mixed loss with $k = 3$ and 2 perform better than the others with the smaller training data sets of 100 and 200. With the larger training data sets, the performances of pointwise loss and mixed loss with $k = 1$ show higher performance than the others. This empirical result suggests that the proposed loss function

Table 1: Precision, Recall, and F1 measure according to the standard evaluation procedure of CoNLL-2002 NER shared task.

	Loss Function						
	point	k=1	k=2	k=3	k=4	k=5	seq
Precision	77.91	77.96	77.95	**78.10**	78.03	77.91	**78.10**
Recall	76.71	76.85	76.88	**76.96**	76.85	76.82	76.85
F1	77.30	77.40	77.41	**77.53**	77.43	77.36	77.47

Table 2: The average F1 measure of NER varying the size of training examples.

Training	**Loss Function**					
Set Size	point	k=1	k=2	k=3	k=4	seq
100	45.36	46.12	**46.96**	46.94	42.72	43.96
200	47.76	47.39	47.44	**47.77**	47.30	47.16
300	**53.37**	52.91	52.68	52.92	52.86	52.40
600	**59.32**	58.68	58.25	58.11	57.34	56.00
1000	61.26	**61.91**	61.38	61.33	61.34	61.05

works well in relatively small data set.

5 Related Work

After MEMMs [24] and CRFs [19] became conspicuous as standard labelers, several variations for enhancing their efficiency and accuracy have been proposed, for example, a perceptron-based efficient learning algorithm called hidden Markov (HM) perceptron [8], a support vector machine (SVM)-based algorithm [5, 27], and a boosting-based algorithm [2]. Most of them including MEMMs and CRFs allow dual representation so as to exploit kernels to handle rich features and their combinations [5, 30, 20, 3, 27]. Although represented in dual forms, they can not exploit structured convolution kernels [14, 12] that exploit arbitrarily large substructures such as string kernels [22, 21], tree kernels [9, 16], and graph kernels [13, 18], since they essentially come back to the primal space when finding the best labeling by using Viterbi decoding. For avoiding this, two stage learning approaches of candidate generation and classification are proposed in [10, 17].

Although our discussion in this paper is based on logarithmic loss functions, other types of loss functions such as exponential loss [2], and hinge loss [5, 30, 3, 27] have been proposed. Our point of view in this paper is orthogonal to this classification, so that our approach could also be applied to those types of objective functions.

In this paper, although we only discussed labeling problems of structured data, the discussion is not only limited to labeling, but more general problems of learning mapping from structured data to structured data [31, 7, 30, 26, 30, 28] are considered. One of the most related works to this paper is semi-Markov CRFs [26] that directly aim to segment sequence data. Their model makes consecutive observed variables correspond to a target variable that indicates a segment. Although the motivation of the semi-Markov CRFs is very similar to ours, their model is not readily deal with tree segmentation. Also, since the ideas of the semi-Markov CRF model and the mixed loss function are uncompetitive with each other, they can naturally complement to each other. While the semi-Markov CRFs cluster observed variables corresponding to segments, our loss function aims predicts clusters of target variables.

6 Conclusion and Discussion

In this paper, we proposed a new loss function called *mixed loss* for information extraction, which is an intermediate loss function between the two loss functions, sequential loss and pointwise loss, so that it has characteristics of both objective functions. Also, we showed that "Markov property" of the mixed loss in sequence labeling, that is, the importance of correct labeling for a particular position depends on the numbers of the correct labels around there.

It is possible to consider L_λ for data with more complex structures such as trees and graphs. In this

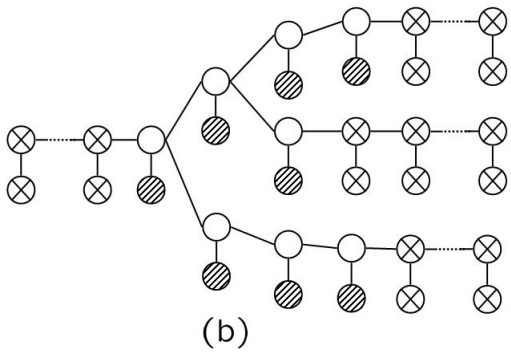


Figure 8: Tree labeling. $\bigotimes$ indicates a dummy variable that always takes a special constant label.

paper, we only showed Theorem 1 in the case of sequence labeling, but the theorem holds for data with rooted tree structures like Figure 8, e.g. information extraction from parse trees [17]. As in the case of sequence labeling, we assume that there are sufficiently many dummy variables above roots and below leaves, all of which take a special label σ_0. If segments in M_k are defined to be all tree-structured subgraphs with all their root-to-leaf paths have length of $k+1$, we can show the equivalence of L_λ and M_k by using a decomposition similar to Eq. (A.1).

$$
\begin{aligned}
(6.11)\quad & \sum_{\boldsymbol{y}:y_t=y_t^{(i)}} \exp\Big(\sum_{\tau\in V} \Theta\Phi(x_\tau^{(i)}, \boldsymbol{y}_\tau^{\pi(\tau)})\Big) \\
&= \Big(\sum_{\boldsymbol{y}_{IN(t)}\cup\{y_t\}:y_t=y_t^{(i)}} \exp\Big(\sum_{\tau\in IN(t)}^{t-1} \Theta\Phi(x_\tau^{(i)}, \boldsymbol{y}_\tau^{\pi(\tau)})\Big)\Big) \\
&\quad \cdot\Big(\sum_{\boldsymbol{y}_{OUT(t)}\cup\{y_t\}:y_t=y_t^{(i)}} \exp\Big(\sum_{\tau\in OUT(t)\cup\{t\}} \Theta\Phi(x_\tau^{(i)}, \boldsymbol{y}_\tau^{\pi(\tau)})\Big)\Big)
\end{aligned}
$$

where V is the set of the indices of tree nodes, and $\pi(t)$ is the index of the parent node of the t-th node. Also, $OUT(t)$ indicates the indices of the nodes "outside" of the t-th node, and similarly, $IN(t)$ is indicates the indices of the "inside" nodes.

For more general graph-structured data, we have no clear correspondence between L_λ and other objective functions so far. However, intuitively, this would also be an intermediate loss function which balances local accuracy and global consistency.

One possible generalization of our loss is to make L_λ position dependent. Although L_λ gives importance to clusters of labels, all positions are equally weighted. However, for example, it seems to be more important to correctly predict boundaries between named entities and other regions, and capability of tuning importance of each position is desirable.

One of the possibilities to make L_λ consider position dependent costs is to employ position dependent weights $w_t^{(i)} \geq 0$ as follows.

$$
L_\lambda := \lambda L_1 + (1-\lambda)\sum_i \sum_{t=1}^{T^{(i)}} w_t^{(i)} \log \sum_{\tilde{\boldsymbol{y}}:\tilde{y}_t=y_t^{(i)}} f(\tilde{\boldsymbol{y}}|\boldsymbol{x}^{(i)})
$$

The larger $w_t^{(i)}$ becomes, the more importance the label for y_t is given.

References

[1] Y. Altun and T. Hofmann. Large margin methods for label sequence learning. In *Proceedings of EuroSpeech*, 2003.

[2] Y. Altun, T. Hofmann, and M. Johnson. Discriminative learning for label sequences via boosting. In *Advances in Neural Information Processing Systems*, 2003.

[3] Y. Altun, T. Hofmann, and A. J. Smola. Gaussian process classification for segmenting and annotating sequences. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[4] Y. Altun, M. Johnson, and T. Hofmann. Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2003.

[5] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[6] S. F. Chen and R. Rosenfeld. A gaussian prior for smoothing maximum entropy models. Technical report, Carnegie Mellon University, 1999.

[7] M. Collins. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning*, pages 175–182, 2000.

[8] M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2002.

[9] M. Collins and N. Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems*, 2002.

[10] M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, 2002.

[11] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of*

Proteins and Nucleic Acids. Cambridge University Press, 1998.

[12] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):S268–S275, 2003.

[13] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the 16th Annual Conference on Computational Learning Theory*, 2003.

[14] D. Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California in Santa Cruz, 1999.

[15] S. Kakade, Y. W. Teh, and S. Roweis. An alternative objective function for Markovian fields. In *Proceedings of the 19th International Conference on Machine Learning*, 2002.

[16] H. Kashima and T. Koyanagi. Kernels for semi-structured date. In *Proceedings of the 19th International Conference on Machine Learning*, pages 291–298, 2002.

[17] H. Kashima and Y. Tsuboi. Kernel-based discriminative learning algorithms for labeling sequences, trees, and graphs. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[18] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[19] 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*, 2001.

[20] J. Lafferty, X. Zhu, and Y. Liu. Kernel conditional random fields: Representation and clique selection. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[21] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[22] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

[23] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA, 1999.

[24] A. McCallum, D.Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning*, 2000.

[25] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C*. Syndicate of the University of Cambridge, Cambridge, U.K., 2nd edition, 1992.

[26] S. Sarawagi and W. W. Cohen. Semi-Markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, 2005.

[27] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*, 2004.

[28] B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2003.

[29] E. F. Tjong and K. Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of Conference on Computational Natural Language Learning*, pages 155–158, 2002.

[30] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[31] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel depndency estimation. In *Advances in Neural Information Processing Systems*, 2003.

Appendix: The Proof of Theorem 1

The following decomposition plays an important role in the proof.

$$
\begin{aligned}
&\sum_{\tilde{\boldsymbol{y}}:\tilde{\boldsymbol{y}}_t^{t+k}=\boldsymbol{y}^{(i)}{}_t^{t+k}} \exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big) \\
&= \Bigg(\sum_{\tilde{\boldsymbol{y}}_{-k+1}^{t}:\tilde{y}_t=y_t^{(i)}} \exp\Big(\sum_{\tau=-k+1}^{t-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Bigg) \\
&\quad\cdot\exp\Big(\sum_{\tau=t}^{t+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\boldsymbol{y}^{(i)}{}_\tau^{\tau+1})\big\rangle\Big) \\
&\quad\cdot\Bigg(\sum_{\tilde{\boldsymbol{y}}_{t+k}^{T^{(i)}+k}:\tilde{y}_{t+k}=y_{t+k}^{(i)}} \exp\Big(\sum_{\tau=t}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Bigg) \\
&= F_t^{(i)}(y_t^{(i)})\cdot\exp\Big(\sum_{\tau=t}^{t+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\boldsymbol{y}^{(i)}{}_\tau^{\tau+1})\big\rangle\Big) \\
&\quad\cdot B_{t+k}^{(i)}(y_{t+k}^{(i)})
\end{aligned}
\tag{A.1}
$$

where we defined,

$$
\begin{aligned}
F_t^{(i)}(y_t^{(i)}) &:= \sum_{\tilde{\boldsymbol{y}}_{-k+1}^{t}:\tilde{y}_t=y_t^{(i)}} \exp\Big(\sum_{\tau=-k+1}^{t-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big), \\
B_t^{(i)}(y_t^{(i)}) &:= \sum_{\tilde{\boldsymbol{y}}_t^{T^{(i)}+k}:\tilde{y}_t=y_t^{(i)}} \exp\Big(\sum_{\tau=t}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big).
\end{aligned}
$$

From (3.5) and (3.6),

$$
\begin{aligned}
\frac{1}{1-\lambda}L_\lambda &= -\sum_i\Big(\frac{\lambda}{1-\lambda}\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\boldsymbol{y}^{(i)}{}_\tau^{\tau+1})\big\rangle \\
&+\sum_{t=-k+1}^{T^{(i)}}\log\sum_{\tilde{\boldsymbol{y}}:\tilde{y}_t=y_t^{(i)}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big) \\
&-(\frac{\lambda}{1-\lambda}+T^{(i)})\log\sum_{\tilde{\boldsymbol{y}}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Big).
\end{aligned}
$$

Substituting $k=\frac{\lambda}{1-\lambda}$ and decomposing the second term, we obtain

$$
\begin{aligned}
\frac{1}{1-\lambda}L_\lambda &= -\sum_i\Bigg(k\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\boldsymbol{y}^{(i)}{}_\tau^{\tau+1})\big\rangle \\
&+\sum_{t=-k+1}^{T^{(i)}}\log F_t^{(i)}(y_t^{(i)})+\sum_{t=-k+1}^{T^{(i)}}\log B_t^{(i)}(y_t^{(i)}) \\
&-(k+T^{(i)})\log\sum_{\tilde{\boldsymbol{y}}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Bigg)
\end{aligned}
$$

Combining the first three terms leads to

$$
\begin{aligned}
&\frac{1}{1-\lambda}L_\lambda \\
&= -\sum_i\Bigg(\sum_{t=-k+1}^{T^{(i)}}\log\Big(F_t^{(i)}(y_t^{(i)}) \\
&\qquad\cdot\exp\Big(\sum_{\tau=t}^{t+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\boldsymbol{y}^{(i)}{}_\tau^{\tau+1})\big\rangle\Big)\cdot B_{t+k}^{(i)}(y_{t+k}^{(i)})\Big) \\
&\quad-(k+T^{(i)})\log\sum_{\tilde{\boldsymbol{y}}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Big)\Bigg).
\end{aligned}
$$

Applying (A.1), we obtain

$$
\begin{aligned}
&\frac{1}{1-\lambda}L_\lambda \\
&= -\sum_i\Bigg(\sum_{t=-k+1}^{T^{(i)}}\log\Big(\\
&\qquad\sum_{\tilde{\boldsymbol{y}}:\tilde{\boldsymbol{y}}_t^{t+k}=\boldsymbol{y}^{(i)}{}_t^{t+k}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Big) \\
&\quad-(k+T^{(i)})\log\sum_{\tilde{\boldsymbol{y}}}\exp\Big(\sum_{\tau=-k+1}^{T^{(i)}+k-1}\big\langle\Theta,\Phi(x_\tau^{(i)},\tilde{\boldsymbol{y}}_\tau^{\tau+1})\big\rangle\Big)\Big)\Bigg) \\
&= M_k.
\end{aligned}
$$

□