Journal of ICT, 14, 2015, pp: 1–20

COST-SENSITIVE STRUCTURED PERCEPTRON INCORPORATING CATEGORY HIERARCHY FOR NAMED ENTITY RECOGNITION

[1]Shohei Higashiyama, [2]Blondel Mathieu, [3]Kazuhiro Seki & [4]Kuniaki Uehara

[1] *Graduate School of System Informatics*
Kobe University, Japan
[2&4] *NTT Communication Science Laboratories, Kobe University, Japan*
[3] *Faculty of Intelligence and Informatics*, *Konan University, Japan*

higashiyama@ai.cs.kobe-u.ac.jp; mathieu@blondel.org;
seki@konan-u.ac.jp;uehara@kobe-u.ac.jp

ABSTRACT

Named Entity Recognition (NER) is a fundamental natural language processing task for the identification and classification of expressions into predefined categories, such as *person* and *organization*. Existing NER systems usually target about 10 categories and do not incorporate analysis of category relations. However, categories often belong naturally to some predefined hierarchy. In such cases, the distance between categories in the hierarchy becomes a rich source of information that can be exploited. This is intuitively useful particularly when the categories are numerous. On that account, this paper proposes an NER approach that can leverage category hierarchy information by introducing, in the structured perceptron framework, a cost function more strongly penalizing category predictions that are more distant from the correct category in the hierarchy. Experimental results on the GENIA biomedical text corpus indicate the effectiveness of the proposed approach as compared with the case where no cost function is utilized. In addition, the proposed approach demonstrates the superior performance over a representative work using multi-class support vector machines on the same corpus. A possible direction to further improve the proposed approach is to investigate more elaborate cost functions than a simple additive cost adopted in this work.

Keywords: Named entity recognition, category hierarchy, cost-sensitive learning, biomedical text mining.

INTRODUCTION

Named Entity Recognition (NER) is a fundamental natural language processing (NLP) task that requires the identification and classification of expressions into predefined categories (e.g., *person*, *location*, *organization*, etc). Existing NER systems typically target about 10 categories and do not assess category relations. However, it is often the case that categories belong to some predefined hierarchy. For example, *organization* can be sub-categorized into *company* and *university*. Named entities that are categorized into fine-grained types could be useful for practical NLP tasks, such as information extraction, question answering, and text data mining. For example, stock price prediction using news articles requires determination of whether each word recognized as an organization name is a company name or not. In such case, it is beneficial that there be a *company* subcategory of *organization*. In general, varieties of categories are required so that NER systems can extract diverse information and knowledge. For these reasons, some researchers have investigated categorization of named entities into more specific categories. For example, Fleischman (2001) and Fleischman and Hovy (2002) have proposed a method for automatic sub-categorization of entities. Sekine, (2002; 2004) Sudo and Sekine and Nobata have developed a named entity hierarchy consisting of around 200 categories. However, the focus of these works was to extend the category set and NER was performed on standard approaches which do not incorporate the relations among category hierarchy in the learning procedure. In this paper, we propose a structured prediction-based approach which leverages the named entity hierarchy. To motivate our approach, we give an example of named entity hierarchy in Figure 1. The hierarchy contains high-level categories, such as *person* and *location*, but also more specific categories, such as *politician*, *doctor*, *country*, and *city*. Although it is incorrect to assign the categories *doctor* or *city* to the entity "Japan", *city* is clearly closer to the correct category *country*. Thus, the degree of misclassification of *city* can be regarded as lower than that of *doctor*. Our approach can leverage this intuition by incorporating misclassification costs in the learning algorithm. It is naturally important to assign entities to the correct categories to achieve more accurate recognition of entities. Additionally, it is necessary to reduce the degree of misclassification because it directly affects NLP tasks for which extracted entities are important clues (e.g., information extraction and question answering).

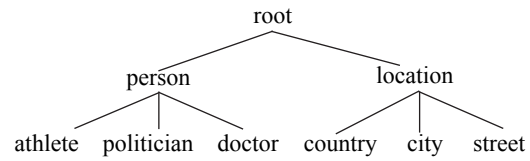

Figure 1. Example of hierarchy of named entities.

For the case in which NE categories belong to some predefined hierarchy, the distance between categories in the hierarchy is a rich source of information that can be exploited. It is intuitively useful particularly for numerous categories. We define a hierarchical cost function tailored for NER which captures the intuition that a greater distance between categories in the hierarchy induces a larger misclassification cost. Using this function, we develop an NER system based on the structured perceptron framework (Collins, 2002). We perform experiments on the GENIA biomedical text corpus and the results demonstrate the effectiveness of our approach at both reducing the degree of misclassifications and increasing the accuracy of exact category predictions.

The remainder of this paper is structured as follows: First, we describe the background of the present work, focusing on NER and the structured perceptron. Second, we summarize the existing works on NER in general and cost-sensitive learning. Then, we detail our proposed approach exploiting category hierarchy in the cost-sensitive learning framework. After that, we report the methodology and results of our evaluative experiments. Lastly, we conclude this paper with a brief summary of the key findings and future work.

INTRODUCTION

Named Entity Recognition

Named entity types and hierarchies

The sixth Message Understanding conference (MUC-6) (Grishman and Sundheim, 1996) was the first to dedicate a task to extracting entities. This task targeted entities of seven types, which include proper names of three types (*person*, *location* and *organization*) and numeral expressions of four types (*date*, *time*, *money* and *percent*). In addition to these types, the IREX project (Sekine and Isahara, 2000) added the type *artifact* (product name or book title) and the ACE program (Doddington, Mitchell, Przybocki, Ramshaw, Strassel and Weischedel, 2004) added *geographical and political entity (GPE)* (location that has a government). The shared task at the CoNLL-2003 workshop (Tjong Kim Sang and De Meulder, 2003) aimed at language-independent NER and its data set has been widely used in a number of recent works. The workshop only targeted entities of four types (*person*, *location*, *organization* and *miscellaneous*). For these reasons, existing NER studies have historically targeted small category sets (usually less than 10 categories) and have not considered relations between these categories (Chieu & Ng, 2002; Isozaki & Kazawa, 2002; McCallum & Li, 2003; Ritter, Clark, Mausam & Etzioni, 2011; Zamin, Oxley & Bakar, 2013; Zhou & Su, 2002).

However, some studies have examined fine-grained entities, which have been proven useful for NLP tasks such as information extraction, question answering, and automated ontology construction. Fleischman (2001) divided the type *location* into subtypes, such as *country*, *city*, and *street* using machine learning algorithms, such as decision tree. Similarly, Fleischman and Hovy (2002) divided the type *person* into subtypes such as *athlete*, *politician* and *doctor*. Sekine and Nobata (2004) defined a named entity hierarchy that includes many fine-grained subcategories such as *museum* and *river*, and added a wide range of categories such as *product* and *event*. Moreover, Ohta, Tateisi and Kim (2002) constructed the GENIA corpus for use in the biomedical domain and introduced a hierarchy consisting of entities, such as gene and protein names, appearing in the corpus. A drawback of these works is that their focus was mainly on extending the category set and NER was performed by standard approaches. In other words, they do not take advantage of the rich information contained in the hierarchy.

Identifying named entity scope

NER can be formulated as a sequence labeling task that requires assignment of a label sequence to an input sequence. However, it is also important for an NER system to identify the entity *scope* (its beginning and its end). To do so, individual word tokens are typically assigned with labels created by combining a semantic class (e.g., *person* and *location*) and an chunk IOB tag, where I, O and B denote the inside, outside, and beginning of an entity, respectively. For example, continuous tokens “in”, “New”, “York” and “City” are denoted with labels O, B-LOCATION, I-LOCATION, I-LOCATION, respectively, meaning that “New York City” is recognized as an entity with semantic class *location*. Note that noun phrase detection (Rahman and Omar, 2013) may precede NER to first identify named entity scopes.

The Structured Perceptron

The perceptron is a classic machine learning algorithm originally proposed by Rosenblatt (1958). It was later extended to voted and averaged variants by Freund and Schapire (1999). They showed that, despite the simplicity of those algorithms, the perceptron performs comparably to support vector machines (SVMs) on various classification problems. The structured perceptron (Collins, 2002) is an extension of the averaged perceptron to structured prediction problems. Collins applied the structured perceptron to sequence labeling tasks, such as part-of-speech tagging, and empirically demonstrated its higher performance than that of maximum entropy. It is well-known that

the structured perceptron can be seen as a special case of structured SVMs (Tsochantaridis, Hofmann, Joachims and Altun, 2004) which does not employ regularization. It sets the step size to one at every iteration. Shalev-Shwartz, Singer and Srebro (2007) proposed an algorithm to analytically set the step size. We use the structured perceptron as our baseline because this method is easy to implement and can be expected to achieve high performance. We show in our experiment section that it performs comparably to the more complicated structured Passive–Aggressive algorithm (Crammer, Dekel, Keshet, Shalev-Shwartz and Singer , 2006).

Learning algorithm

We now describe the structured perceptron learning algorithm. Let χ be the set of instances and let Yx be the set of possible label sequences for an instance $x \in X$, where x denotes a token sequence (e.g., a sentence) in the training or test set. Likewise, $y \in \Upsilon_x$ denotes a possible label sequence of x. Υ_x is equivalent to the direct product L, where *n* is the length of x and L is the set of labels (e.g., B-PERSON and O). Let *f*: $\chi \rightarrow \Upsilon$ be the discriminative function defined as

$$f(x,y)=\langle w,\Phi(x,y)\rangle , \quad (1)$$

where $\langle \cdot,\cdot \rangle$ denotes the inner product and $\Phi xy \in R$ is the feature vector of x and y. The goal of learning is to find a weight vector $w \in R^d$ so that the discriminative function f predicts the correct label sequences of training instances. The prediction $\hat{y}$ for x is given by

$$\hat{y}=\text{argmax}_{y \in \Upsilon} f(x,y) \quad (2)$$

On each round *t* of training, the structured perceptron receives a training instance-label pair $x_t y_t$ and outputs its prediction $\hat{y}t$ by Eq. (2). If $\hat{y}_t \neq y_t$, w is then updated as follows:

$$w^{t+1} \leftarrow w^t + \Phi(x_t,y_t) - \Phi(x_t,\hat{y}_t) \quad (3)$$

where w^t is the weight vector on round *t*. Learning is iterated through all the (permuted) training instances *T* times. Label sequences of test instances can be predicted by Eq. (2) in the same manner as training instances.

Parameter averaging

Collins (2002) showed that the accuracy of the structured perceptron model can be improved by averaging learned parameters $w^{j,i}$ in the end of learning. In

precise, let $w^{j,i}$ denotes the weight vector after updating it by the i-th instance in the j-th iteration. Then, the averaged weight vector $\overline{w}$ is defined as follows:

$$\overline{\mathbf{w}} = \frac{1}{nm} \sum_{j=1}^{m} \sum_{i=1}^{n} \mathbf{w}^{j,i} \tag{4}$$

where *m* is the number of training iterations and *n* is the number of instances in the training data. After the entire training phase, one could use $\overline{\mathbf{w}}$ as the weight vector instead of $\mathbf{w}^{m,n}$. This technique is called "parameter averaging", which will be also employed in the present work.

LITERATURE REVIEW

Named Entity Recognition

Recent NER methods mostly rely on supervised machine learning. In the CoNLL-2003 shared task, the most popular method among the participants was maximum entropy. In particular, Florian (2003) achieved the best performance by a system that combines maximum entropy with other methods such as hidden Markov models. Chieu and Ng (2003) achieved the second best performance by also using maximum entropy. Conditional random fields (CRFs) (Lafferty, McCallum & Pereira., 2001) extend maximum entropy to structured prediction. For decoding, Finkel et al. (2005) used Gibbs sampling instead of the more widely used the Viterbi and Forward–Backward algorithms. They applied the method to the English dataset in the CoNLL-2003 shared task and reported comparable performance to that of the participants. Tsochantaridis et al. (2004) proposed an extension of SVMs to structured prediction. They applied the structured SVMs to NER and reported lower error rates of the methods than those of CRFs.

As for studies of entities from categories belonging to some predefined hierarchy, some have used the GENIA corpus (Ohta, Tateisi & Kim., 2002). In the GENIA corpus, the named entity hierarchy is represented by a tree and includes 48 categories of biomedical terms. In fact, 36 of those categories corresponding to leaf nodes are used for annotation. However, many studies using the GENIA corpus target only a subset of the categories. For example, the JNLPBA shared task (Kim, Ohta, Isuruoka and Tsujii, 2004) was limited to 5 out of the 36 categories for simplifying the task. However, Lee et al. (2004) targeted all categories in the GENIA corpus. They formulated hierarchical NER as a two-phase classification problem: first identifying whether a word is a named entity or not, then classifying identified entities using multi-class

SVMs. They reported higher performance than a one-phase classification or a rule-base method.

Cost-sensitive Learning and Category Hierarchies

Cost-sensitive learning is an machine learning framework that incorporates misclassification costs. Although standard non-cost-sensitive methods maximize the accuracy of classification, cost-sensitive methods lower the expected misclassification costs defined by a cost function that returns a different cost for each different error.

An existing cost-sensitive approach, Passive–Aggressive (PA) (which is a kind of margin perceptron) was proposed by Crammer, Dekel, Keshet, Shalev Shwart & Singer (2006). PA is very similar to the cost-sensitive structured perceptron approach that we develop. Johansson and Nugues (2008) applied cost-sensitive PA to semantic role labeling (SRL). Although they reported that their system particularly addressing dependency syntax has performance approaching that of more general constituent-based systems for SRL, they did not report a comparison with non-cost-sensitive models. Therefore, contributions by cost functions were not clear. Moreover, Johansson and Nugues used a somewhat crude cost function for an input sentence and a parse tree, which returns only three values: zero for a correct tree, 0.5 for a partial error, and one for an outright error. Hierarchical classification was previously formulated as cost-sensitive online learning problem by Dekel, Keshet & Singer (2004), but their method only supports regular classification. Song, Son, Noh, Park and Lee (2012) applied cost-sensitive multi-class SVMs to part-of-speech (POS) tagging using cost functions that consider a hierarchy of POS tags. They reported that their methods reduced serious errors compared to other non-cost-sensitive POS taggers and achieved higher performance for a subsequent NLP task of text chunking. However, they formulated POS tagging task as not a sequence labeling problem but a multi-class classification problem. Therefore, the cost functions that they used only support multi-class classification, similarly to the work by Dekel et al. (2004).

We target a somewhat more difficult problem, since NER is a sequence labeling problem. Moreover, we propose a cost function that is tailored for NER.

EXPLOITING CATEGORY HIERARCHY VIA COST-SENSITIVE LEARNING

As we emphasized, category hierarchy is a rich source of information which encodes prior knowledge about the category relationships. By considering this

information, we can expect to achieve lower degree of misclassification errors. In this section, we formulate NER with a category hierarchy as a cost-sensitive structured prediction problem. To achieve lower misclassification costs, we use a cost-sensitive extension of the structured perceptron and propose a novel hierarchical cost function tailored for NER. In *Evaluation* section later, we compare our hierarchical cost function with other cost functions traditionally used for structured prediction.

Cost-sensitive Structured Perceptron

In this section, we briefly describe how to extend the standard structured perceptron to incorporate a cost function. We define a cost function over labels c:$L\times L\rightarrow$R+, i.e., a function which from a pair of labels returns a non-negative real value. We assume that the cost function returns zero if the two labels are the same. As an example, the 0/1 cost function can be defined as a function which returns 0 when the two labels are the same and always returns 1 when the two labels differ (i.e., all misclassifications are assumed to have an equal cost). From the cost function over labels, we then define a cost function over label sequences as the sum of individual costs:

$$C(\mathbf{y},\mathbf{y}')=\sum_{k=1}^{|\mathbf{y}|} c(y_k,y'_k)\,, \tag{5}$$

where y and y'$\in Y$ are sequences of equal length, |y| denotes the length of y and yk (or $y'k$) denotes the k th element in the sequence y_k (or y'_k). The basic idea of the cost-sensitive extension of the structured perceptron is to use the following evaluation function f_C instead of f as in Eq. (1):

$$fC(x\_^{i},y)=\langle w\Phi x,y\rangle+\alpha C(y\__{t},y)=>w,\Phi(x,y)>+\alpha C(y\__{t},y) \tag{6}$$

where intuitively $\alpha>0$ controls the effect of the cost function C on learning. In the training phase where correct label sequences are available, the evaluation function f_C is utilized and the score of $\mathbf{y}$ is computed. Then, the label sequence with the highest score is output as in Eq. (7).

$$\hat{\mathbf{y}}_t=\text{argmax}_{\mathbf{y}\in Y} f(\mathbf{x}_t,\mathbf{y})+\alpha C(\mathbf{y}_t,\mathbf{y}) \tag{7}$$

If the output $\hat{y}t$ is different from the correct label sequence, the weight vector w is updated by Eq. (3) as with the standard structured perceptron where the cost of misclassification is not considered. In the test phase, on the other hand, the correct label sequence is not known and the prediction is made based on Eq. (2).

Effects of Introducing Cost Function in Learning

The previous section described our proposed approach incorporating misclassification cost and how it can be trained and can be used for prediction. This section discusses the expected effect by introducing the cost function in learning.

The evaluation function f_C defined in Eq. (6) is the sum of the original (cost-insensitive) function f in Eq. (1) and the cost function C weighted by α. As will be defined in the next section, the cost indicates how much a label sequence **y** and the correct sequence **y***t* for input x*t* differ. This means that more different label sequences tend to have larger scores for the function f_C and thus are often chosen as the prediction. This may sound counterintuitive but works as follows: if the prediction $\hat{\mathbf{y}}$ is incorrect, it will be used to update the weight vector w by Eq. (3) such that the inner product of the feature vector $\Phi(x\_{t},y)$ and w becomes smaller and, at the same time, that of $\Phi(x_t y_t)$ and w becomes greater. In short, because of the boost by the cost function, label sequences more different from the correct label sequence may be initially chosen, but through a number of updates, those label sequences will eventually have smaller scores. Here, it should be mentioned that, despite the cost function, even if a label sequence similar to the correct one had a high score and were chosen, then the right side of Eq. (3) will be mostly canceled out due to the similarity between the correct sequence and the predicted one and thus it would not have a large effect on the update of the weight vector w.

In addition, mutually similar label sequences would have a similar set of features. Therefore, adjusting weights w for the correct label sequence y_t would also adjust them, to some extent, in favor of label sequences similar to y_t. The same applies to an incorrect prediction ŷ and its similar sequences as well. As a result, after many iterations, label sequences similar to the correct sequence would have high scores for the evaluation function according to their similarities and, conversely, those which are far from the correct one would have low scores.

In the test phase using Eq. (1), on the other hand, there is no longer the boost by the cost function and the difference between the scores of the correct label sequence and incorrect ones will be made greater. Therefore, it is expected that more correct sequences will be chosen for prediction.

In summary, the advantages of using the evaluation function f_C as defined in Eq. (6) are

- To actively use label sequences far different from the correct one, which enables us to effectively update the weight vector w and

- To further differentiate the scores of the evaluation function for the correct and incorrect label sequences by way of removing the term representing the cost function in the test phase.

Especially, when we use a cost function considering the hierarchical nature of the categories of named entities, even incorrect tags in predicted label sequences will contribute in learning since they are distinguished by their distance or closeness to the correct ones in the hierarchy.

A Hierarchical Cost Function for NER

We now describe our hierarchical cost function tailored for NER. When a token that is a component of an entity is misclassified into an incorrect category, its misclassification cost can be defined as the distance between the correct and the predicted category in the hierarchy. However, as we previously emphasized, we do not assign directly entity categories (e.g., *location*) to individual word tokens but rather IOB-augmented labels (e.g., B-LOCATION and I-LOCATION). This means that we cannot use the category hierarchy (such as the one illustrated in Figure 1) directly, since it is defined over entity categories, not IOB-augmented labels. Thus, we combine a distance function over entity categories and a mapping function for retrieving entity categories from IOB-augmented labels. We first assume that the hierarchical structure of categories forms a tree and define the distance d between two nodes as the number of edges in the shortest path connecting them in the tree.

Then, to extend the notion of distance between entity categories to IOB-augmented labels, we introduce a simple mapping function. Formally, let $E=\{e_1,....,e_N\}$ be the set of entity categories, where N is the number of categories. Additionally, let $L_e=\{l_{1B}l_{1I}...,l_{NB},l_{NI}\}$ be the set of augmented labels. We used l_{nB} to denote the label of the category e_n with tag B. Similarly, l_{nI} denotes the label of the category e_n with tag I. We use l_O to denote the label of non-entity tag O. The entire set over which the structured perceptron makes prediction for each individual word token is thus $L=L_e\cup\{l_O\}$. We define a mapping function $\phi:L_e\rightarrow E$, which from an augmented label outputs the corresponding entity category as

$$\begin{aligned}\phi(l_{nB}) &= e_n \\ \phi(l_{nI}) &= e_n\end{aligned} \tag{8}$$

Using the distance between categories $d:E\times E\rightarrow R+$ and the mapping function ϕ, we now define the hierarchical cost function $c_{hie}:L\times L\rightarrow R+$ in Eq. (9).

$$c_{hie}(l,l')=\begin{cases} d(\phi(l);\phi(l')) & \text{if } l;l' \in \mathcal{L}_e \\ 0 & \text{if } l=l'=l_O \\ M & \text{otherwise} \end{cases} \quad (9)$$

Therein, M denotes the maximum value of function c_{hie}. Function c_{hie} returns the maximum value M when either l or l' is equal to l_O. In our model, we determine M by

$$M=\max_{e,e' \in Le} d(e,e')\}+1 \quad (10)$$

The first term on the right-hand side of the equation is the maximum value of the distance d which depends on the hierarchical structure. To summarize, we have defined a hierarchical cost function c_{hie} which describes the distance separating labels, including entity and non-entity labels. We use the *chie* function in the structured perceptron framework for lowering misclassification costs based on the category hierarchy.

EVALUATION

Dataset

We evaluated our approach for NER with category hierarchy with the GENIA corpus ver. 3.02. The corpus contains 2,000 biomedical literature abstracts annotated with named entity types and boundaries of terms in molecular biology. The corpus provides a hierarchy of biomedical entity categories, such as protein and gene names. The number of unique entity categories occurring in the corpus is 36. The maximum depth of the hierarchy is six[1]. The GENIA corpus sometimes includes nested entities. For example, in the sentence "IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase", "IL-2 gene expression" is annotated with *other_name* and "IL-2 gene" is also annotated with *DNA_domain_or_region*. When entities are nested in that way, we targeted the most shallowly nested entities. That is, in the previous example, we assumed that "IL-2 gene expression" was annotated only with *other_name*.

[1] The GENIA corpus ver. 3.0 hierarchy is described in the paper by Kim et al. (2003). The same hierarchy is used for ver. 3.02 except that some parts of names of categories differ from those for the ver. 3.0.

Features

For features, we used a combination of general-purpose NER features and biomedical domain features. Specifically, our features were based on the followings:

- Tokens in a window of size two around the current token,
- The prefix and the suffix of the current token, and
- The character types of the current token.

We restricted prefix and suffix features to those occurring at least 30 times in the corpus and used character type features of 17 types as proposed by Zhou and Su (2004). This included biomedical domain features, such as Roman numerals, Greek letters and base sequence features (A, T, C, and G).

These features were used along with the k-th label, y_k, in label sequence **y** and a pair of $k-1$-th and k-th labels, (y_{t-1}, y_t) (i.e., label bi-gram). Let O denotes the vocabulary set appearing in the training data. Then, for example, features for a term $o_h \in O$ and labels $l_i l_j \in L$ are defined below, where x_k is the term we currently focus on.

$$\varphi^{0}_{h,i}(x_k, y_k) = \begin{cases} 1 & \text{if } x_k = o_k \wedge y_k = l_i \\ 0 & \text{otherwise} \end{cases}$$

$$\varphi_{h,i,j}(x_k, y_k, y_{k-1}) = \begin{cases} 1 & \text{if } x_k = o_k \wedge y_k = l_i \wedge y_{k-1} = l_j \\ 0 & \text{otherwise} \end{cases}$$

Evaluation metrics

We evaluated our approach on the GENIA corpus using a 10-fold cross-validation, i.e., all results in this section are averaged over 10 randomized experiments. For an evaluation metric, we used the average of the c_{hie} costs over all tokens in the test set. To be specific, the average cost was computed as follows:

$$\overline{c}_{hie} = \frac{1}{N(\mathcal{T})} \sum_{\mathbf{x} \in \mathcal{T}} \sum_{x_t \in \mathbf{x}} c_{\text{hie}}(\hat{y}_k, y_k) \qquad (11)$$

where $\mathcal{T}$ denotes the test set, $N(\mathcal{T})$ denotes the total number of terms appearing in $\mathcal{T}$, and y_k and $\hat{y}_k$ denote the correct and predicted labels for k-th term x_k of $x \in \mathcal{T}$, respectively.

The performance is considered better when the average cost is lower. The value range of the function c_{hie} depends on category hierarchies. For the GENIA corpus, c_{hie} takes 0 as the minimum value and 11 as the maximum value (M in Eq. (10)). Later, we also use the F-measure for the purpose of comparing our approach with an existing method. Note however that the F-measure can not distinguish different degrees of misclassification costs and is not suitable for evaluating our approach. On the other hand, the average cost defined above becomes 0 for the perfect predictions and becomes greater as the predictions become farther from the correct label sequences in the category hierarchy. In this sense, the average cost is more suitable for assessing the performance of the models considering the hierarchical structure of NE categories, such as our approach, than the F-measure.

Effect of the parameter α

For using the cost function in Eq. (6), parameter α needs to be properly set. For this purpose, we determined the optimal value for α as follows:

1. For each fold of cross-validation, we divided the fold into two subsets; one was for learning and the other was for validation. The former was randomly chosen 75% of the fold and the latter was the remaining 25%.
2. We set α to a certain value and learned a model using the learning data. Then, we computed the average cost (see Eq. (11)). This procedure was repeated for all the folds and the mean average cost over all the folds was associated with the particular value of α chosen in the beginning.
3. We change the value of α and repeat the above procedure. The value of α leading to the least cost was considered as the optimum.

To examine the effect of our hierarchical cost function, we computed the average cost for different values of α. Specifically, α was increased from 0 to 10 by a step size of 0.2. The number of iterations in training was set to five. Note that α=0 is equivalent to the model with no cost function. The results are plotted in Figure 2. When the value of α is relatively small, the cost function has a small effect on the resulting model and consequently the average cost is large. When α becomes larger, the average cost gradually decreases, which indicates that the cost function is working properly in updating the weight vector **w**. However, the average cost more or less saturated after around α=6. There, we decreased the step size to 0.1 and explored the optimal α in the range between 6 and 10. As a result, we found that α=9.0 yielded the least average cost and determined it as the optimum.

In the following experiments, we used the optimal α=9.0 and performed cross-validation. For each fold, we re-trained the model using the entire fold

data (the original fold data before splitting into training and validation data mentioned above) and used the test data for assessing the performance.

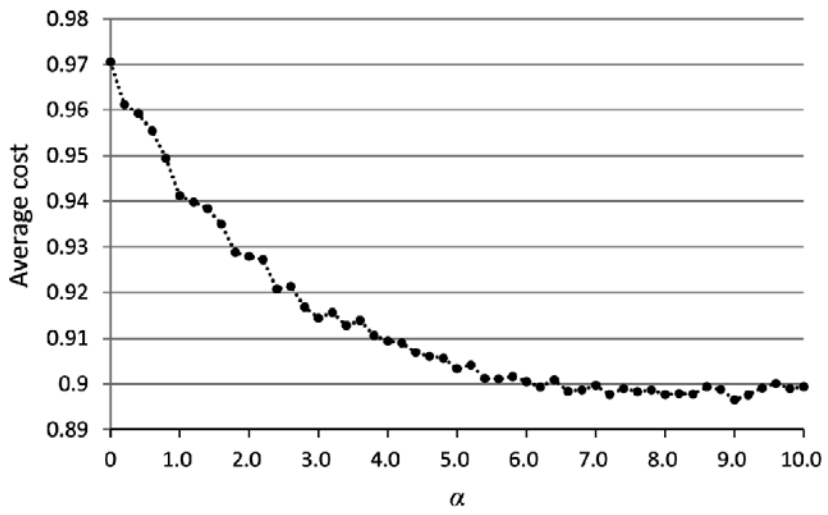

Figure 2. Average cost for increasing values of α when using the cost function c_{hie}.

Comparison with Other Cost Functions

In the present work, we defined the hierarchical cost function *chie* considering the category hierarchy as in . However, other cost functions, especially those that do not use category hierarchy information, are important for validating the effectiveness of our proposed cost function. To this end, we define a 0/1 cost function *c*0/1, which returns zero when two labels belong to the same category and one when they do not as shown in Eq. (12).

$$c_{0/1}(l,l') = \begin{cases} 0 & \text{if } \phi(l)=\phi(l') \\ 1 & \text{otherwise} \end{cases} \quad (12)$$

Note that in this case, C in Eq. (12) corresponds to the Hamming loss. In addition to using *c*0/1, we also compared our approach to the standard structured perceptron (i.e., without cost function). The optimal value of α for the 0/1 cost function was determined in a similar way to that described in the previous section. The average costs obtained from a 10-fold cross-validation are given in Table 1.

It is observed that the models using cost functions outperformed the standard structured perceptron. In particular, the c_{hie} function performed the best. Based on paired *t*-tests, we found that the results were significant at the 0.01 confidence level. These results suggest that cost functions are effective for decreasing the degree of misclassifications. In addition, that cost functions considering the category hierarchy are more effective. However, we found that the cost-sensitive structured perceptron needs slightly more iterations to obtain optimal performance.

Table 1

Comparison of cost functions. Results were significant at the 0.01 confidence level

	Ave. cost	α	*T*
With no cost function	0.94	0	6
*c*0/1 function	0.90	47	9
chie function	0.88	9	10

Comparison with structured Passive–Aggressive

To evaluate the effectiveness of using the structured perceptron for learning, we compared it to the structured extension of the Passive–Aggressive (PA) algorithm (Crammer et al., 2006). PA is a kind of margin perceptron that updates the parameter vector at each iteration by solving a small quadratic program. PA differs from the structured perceptron in that the step size of PA depends on the cost function shown below:

$$\hat{\mathbf{y}}_\mathbf{t} = \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}\cdot\Phi(\mathbf{x}_t,\mathbf{y}) + C(\mathbf{y},\mathbf{y}_t)$$

$$\tau_t = \min\left(C, \frac{\mathbf{w}\cdot(\Phi(\mathbf{x}_t,\hat{\mathbf{y}}_\mathbf{t}) - \Phi(\mathbf{x}_t,\mathbf{y}_t)) + C(\hat{\mathbf{y}}_\mathbf{t},\mathbf{y}_t)}{|\Phi(\mathbf{x}_t,\mathbf{y}_t) - \Phi(\mathbf{x}_t,\hat{\mathbf{y}}_\mathbf{t})|^2}\right)$$

$$\mathbf{w} \leftarrow \mathbf{w} + \tau_t(\Phi(\mathbf{x}_t,\mathbf{y}_t) - \Phi(\mathbf{x}_t,\hat{\mathbf{y}}_\mathbf{t})) ,$$

where $C>0$ is a regularization parameter. Our preliminary experiments using the validation sets showed that a smaller value of C led to a larger number of iterations but achieved a lower average cost when converged. A trade-off was found between training time and performance. The value of C which achieved the highest performance was 0.03.

The results on the test sets (after re-training the model on the entire training set once α and *T* have been selected) are shown in Table 2. The performance and the number of iterations for the structured perceptron (SP) are listed again for reference. The average cost for PA was slightly lower than that for the structured perceptron, but no significant difference was found by a paired *t*-test even at the 0.05 significance level. In the meantime, the number of iterations for training PA was almost three times as large as that of the structured perceptron due to the fact that PA needed a larger number of iterations to achieve the highest performance. Therefore, we can say that the structured perceptron has close performance to PA in spite of the much shorter training time.

Table 2

Comparison with structured Passive–Aggressive

	Ave. cost	Regularization	*T*
SP + *chie* function	0.88	α=9	10
PA + *chie* function	0.87	*C*=0.03	29

Comparison with an Existing Method

We compared our approach with the method of Lee, Hwang, Yim & Rim, (2004). We chose their method because they targeted all categories in the GENIA corpus unlike many studies, including those from the JNLPBA shared task (Kim et al., 2004). Their method performed NER using a two-step approach. In the first step, a word token was classified into entity or non-entity. In the second step, if a word token was an entity, it was further classified in entity categories using multi-class SVMs. They used a 10-fold cross-validation and reported a 66.7% *F*-measure for all categories. We evaluated the structured perceptron with no cost function and with our proposed hierarchical cost function. Unlike the previous experiments, we used the *F*-measure as an evaluation metric in this experiment. The value of α and the number of iterations *T* are the same as those found and reported previously.

Our method outperformed the method reported in earlier study (Lee et al., 2004) by around four points. The structured perceptron (with no cost function) even outperformed it by around three points. This is presumably due to that Lee et al. predicted labels in a sequence independently. That is, preceding terms' labels had no effect on the succeeding terms' labels. However, the structured perceptron does consider the preceding terms and labels. Our results suggest that approaches considering the overall sequence of tokens when predicting the label sequence are more effective for NER tasks.

From this experiment, we see that our hierarchical cost function improved the *F*-measure. The *F*-measure for the structured perceptron with and without the cost function was significantly different in a paired *t*-test at a 0.01 significance level. We can thus conclude that our hierarchical cost function not only reduces the degree of misclassifications but also increases the accuracy of exact category predictions, as indicated by the *F*-measure.

Table 3

Comparison with an existing method

	F-measure
Lee et al. (2004)	66.7
SP (with no cost function)	69.8
SP (*chie* function)	70.7

CONCLUSION

This paper described our named entity recognition method considering category relations of named entities in a hierarchy. The proposed method leverages category hierarchy information by introducing a hierarchical cost function tailored for NER in the cost-sensitive framework. We evaluated the effectiveness of our proposed method on the GENIA biomedical text corpus. Experimental results strongly suggest that our approach not only decreases the degree of misclassifications but also significantly increases the accuracy of exact category predictions. For simplicity, the cost of an entire sequence in Eq. (5) was defined as the sum of individual costs. In future work, we plan to investigate more elaborate cost functions on the entire sequence.

ACKNOWLEDGEMENTS

This work is partially supported by JSPS Kakenhi grant #25330363 and MEXT, Japan.

REFERENCES

Chieu, H., & Ng, H. (2003). Named entity recognition with a maximum entropy approach. In *Proceedings of the 7th Conference on Natural Language Learning (CoNLL-2003)*, pages 160–163.

Chieu, H. L., & Ng, H. T. (2002). Named entity recognition: A maximum entropy approach using global information. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7.

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1-8.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *The Journal of Machine Learning Research (JMLR)*, *7*, 551-585.

Dekel, O., Keshet, J., & Singer, Y. (2004). Large margin hierarchical classification. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 27-42.

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., & Weischedel, R. (2004). The automatic content extraction (ACE) program–tasks, data, and evaluation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, 837-840.

Finkel, J., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*, 363–370.

Fleischman, M. (2001). Automated subcategorization of named entities. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, 25-30.

Fleischman, M., & Hovy, E. (2002). Fine grained classification of named entities. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, 1-7.

Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). Named entity recognition through classifier combination. In *Proceedings of the 7th Conference On Natural Language Learning (CoNLL-2003)*, 168-171.

Freund, Y., & Schapire, R. (1999). Large margin classification using the perceptron algorithm. *Machine Learning*, *37*(3),277–296.

Grishman, R., & Sundheim, B. (1996). Message understanding conference-6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, 466-471.

Isozaki, H., & Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th International Conference on Computational Linguistics*, 1-7.

Johansson, R., & Nugues, P. (2008). Dependency-based semantic role labeling of propbank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 69–78.

Kim, J., Ohta, T., Tateisi, Y., & Tsujii, J. (2003). Genia corpus–a semantically annotated corpus for bio-textmining. *Bioinformatics*, *19* (Suppl.1),180-182.

Kim, J., Ohta, T., Tsuruoka, Y., Tateisi, Y., & Collier, N. (2004). Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA)*, pages 70-75.

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

Lee, K., Hwang, Y., Kim, S., & Rim, H. (2004). Biomedical named entity recognition using two-phase model based on SVMs. *Journal of Biomedical Informatics*, *37*(6),436-447.

McCallum, A. and Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 188-191.

Ohta, T., Tateisi, Y., & Kim, J. (2002). The genia corpus: An annotated research abstract corpus in molecular biology domain. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT 2002)*, pages 82-86.

Rahman, S. A., & Omar, N. (2013). Transforming noun phrase structure form into rules to detect compound nouns in Malay sentences. *Journal of Information and Communication Technology*, *12*,161-173.

Ritter, A., Clark, Mausam, S., & O, Etzioni. (2011). Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1524-1534.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6),386–408.

Sekine, S., & Isahara, H. (2000). IREX: IR and IE evaluation project in Japanese. In *Proceedings of the 2nd of the Language Resources and Evaluation Conference (LREC)*, 1475–1480.

Sekine, S. and Nobata, C. (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, 1977–1980.

Sekine, S., Sudo, K., & Nobata, C. (2002). Extended named entity hierarchy. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, 1818–1824.

Shalev-Shwartz, S., Singer, Y., & Srebro, N. (2007). Pegasos: Primal estimated sub-gradient solver for SVM. In *Proceedings of the 24th International Conference on Machine Learning*, 807–814.

Song, H., Son, J., Noh, T., Park, S., and Lee, S. (2012). A cost sensitive part-of-speech tagging: differentiating serious errors from minor errors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, 1025–1034.

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning (CoNLL-2003)*, 142–147.

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

Zamin, N., Oxley, A., & Bakar, Z. A. (2013). Projecting named entity tags from a resource rich language to a resource poor language. *Journal of Information and Communication Technology*, *12*:, 121–146.

Zhou, G., & Su, J. (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 473–480.

Zhou, G., & Su, J. (2004). Exploring deep knowledge resources in biomedical name recognition. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA)*, 96–99.