Information Processing and Management 49 (2013) 954–965

Contents lists available at SciVerse ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Named entity recognition with multiple segment representations

Han-Cheol Cho [a,*], Naoaki Okazaki [b], Makoto Miwa [c], Jun'ichi Tsujii [d]

[a] Suda Lab., Dept. of Computer Science, The University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113-8656, Japan
[b] Inui & Okazaki Lab., Dept. of System Information Sciences, Tohoku University, 6-3-09 Aramakiaza-Aoba, Aoba-ku, Sendai 980-8579, Japan
[c] National Centre for Text Mining, Manchester Interdisciplinary Biocentre, 131 Princess Street, Manchester M1 7DN, UK
[d] Microsoft Research Asia, New West Campus, 3rd Floor, Tower 2, No. 5, Dan Ling Street, Haidian District, Beijing 1000080, People's Republic of China

ARTICLE INFO

Article history:
Received 9 April 2012
Received in revised form 3 March 2013
Accepted 5 March 2013
Available online 9 April 2013

Keywords:
Named entity recognition
Machine learning
Conditional random fields
Feature engineering

ABSTRACT

Named entity recognition (NER) is mostly formalized as a sequence labeling problem in which segments of named entities are represented by label sequences. Although a considerable effort has been made to investigate sophisticated features that encode textual characteristics of named entities (e.g. PEOPLE, LOCATION, etc.), little attention has been paid to segment representations (SRs) for multi-token named entities (e.g. the *IOB2* notation). In this paper, we investigate the effects of different SRs on NER tasks, and propose a feature generation method using multiple SRs. The proposed method allows a model to exploit not only highly discriminative features of complex SRs but also robust features of simple SRs against the data sparseness problem. Since it incorporates different SRs as feature functions of Conditional Random Fields (CRFs), we can use the well-established procedure for training. In addition, the tagging speed of a model integrating multiple SRs can be accelerated equivalent to that of a model using only the most complex SR of the integrated model. Experimental results demonstrate that incorporating multiple SRs into a single model improves the performance and the stability of NER. We also provide the detailed analysis of the results.

1. Introduction

Named Entity Recognition (NER) aims to identify meaningful segments in input text and categorize them into pre-defined semantic classes such as the names of people, locations and organizations. This is an important task because its performance directly affects the quality of many succeeding natural language processing (NLP) applications such as information extraction, machine translation and question answering. NER has been mostly formalized as a sequence labeling problem that performs the recognition of segments and the classification of their semantic classes simultaneously by assigning a label to each token of an input text.

While many researchers have focused on developing features that capture textual cues of named entities, there are only a few studies (Leaman & Gonzalez, 2008; Ratinov & Roth, 2009) that examined the effects of different segment representations (SRs) such as the *IOB2* and the *IOBES* notations. This issue has been extensively discussed for a different NLP task, word segmentation (WS). In this task, complex SRs consisting of four to six segment labels have been proposed based on linguistic

* Corresponding author. Tel.: +81 90 8561 5481; fax: +81 22 795 4285.
E-mail addresses: hccho@is.s.u-tokyo.ac.jp (H.-C. Cho), okazaki@ecei.tohoku.ac.jp (N. Okazaki), makoto.miwa@manchester.ac.uk (M. Miwa), jtsujii@microsoft.com (J. Tsujii).

0306-4573/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ipm.2013.03.002

intuitions (Xue, 2003) and statistical evidence from corpora (Zhao, Huang, Li, & Lu, 2006) and shown to be more effective than the simple *BI* SR.[1] However, complex SRs are not always beneficial, especially when the size of training data is small, because they can result in undesirably sparse feature space. In NER, the data-sparseness problem is an important issue because only a small portion of training data is named entities. Therefore, the use of a complex SR, which may better explain the characteristics of target segments than a simple SR, may not be much effective or even can bring performance degradation.

In this paper, we present a feature generation method that creates an expanded feature space with multiple SRs. The expanded feature space allows a model to exploit highly discriminative features of complex SRs while alleviating the data-sparseness problem by incorporating features of simple SRs. Furthermore, our method incorporates different SRs as feature functions of Conditional Random Fields (CRFs), so we can use the well-established procedure for training. We also show that the tagging speed of a proposed model using multiple SRs can be boosted up as fast as that of the model using only the most complex SR of the proposed model. The proposed method is evaluated on the two NER tasks: the BioCreative 2 gene mention recognition task (Smith, 2008) and the CoNLL 2003 NER shared task (Tjong Kim Sang & De Meulder, 2003). The experimental results demonstrate that the proposed method contributes to the improvement of NER performance.

The next section investigates several SRs developed for various NLP tasks, and explains a hierarchical relation among them that is the key concept to our proposed method. In Section 3, we shows the effect of different SRs on NER and analyze the results in two ways. This analysis motivates the necessity of using multiple SRs for NER. Section 4 describes the proposed feature generation method that creates an expanded feature space with multiple SRs. We also show how to speed up the tagging speed of a model using the proposed method. In Section 5, we present the experimental results and the detailed analysis. Finally, Section 6 summarizes the contribution of our research and future work.

2. Segment representations

SRs are necessary for sequence labeling tasks that involve segmentation as a sub-task. This section introduces SRs used in various NLP tasks and presents a hierarchical relation among these SRs that will become the basis of our proposed method.

2.1. Segment representations in various NLP tasks

Several SRs have been developed for and adopted to various NLP tasks such as NER (Ratinov & Roth, 2009), WS (Xue, 2003; Zhao et al., 2006) and shallow parsing (SP) (Kudo & Matsumoto, 2001; Tjong Kim Sang & Veenstra, 1999). Table 1 presents the definition of some of these SRs. Each SR in the *SR type* column consists of segment labels in the *Segment Labels* column. The *Examples* column presents a few example label sequences of named entities, chunks and words with respect to the target tasks. We would like to note that the *O* label of the SRs in the NER and the SP tasks denotes a token that does not belong to any target segments. In WS, however, the *O* label is not necessary because every character of an input sentence is a part of a word.

In NER, the *IOB*2 and the *IOBES* SRs have been used most frequently. The *IOB*2 SR distinguishes tokens at the **B**eginning, the **I**nside and the **O**utside of named entities. On the other hand, the *IOBES* SR identifies tokens at the **B**eginning, the **I**nside and the **E**nd of multi-token named entities, tokens of **S**ingle token named entities and tokens of the **O**utside of named entities. In SP, the *IOB*2 and the *IOBES* SRs work in the same manner as in NER. The *IOE*2 SR uses the *E* label to differentiate the end tokens of chunks instead of the *B* label of the *IOB*2 SR. The *IOB*1 and the *IOE*1 SRs are basically equivalent to the *IO* SR that uses the *I* label to denote tokens of chunks and the *O* label to indicate tokens outside chunks. However, the *IO* SR cannot distinguish the boundary of two consecutive chunks of a same type. To overcome this problem, the *IOB*1 SR assigns B^* label to the token at the beginning of the second chunk, whereas the *IOE*1 SR gives the E^* label to the token at the end of the first chunk. Lastly, in WS, the *BI* SR identifies the beginning and the inside of words, the *BIS* SR deals with single character words separately by assigning the *S* label to these words and the *BIES* SR uses the *E* label for the end characters of words. In addition, the BB_2IES assigns the B_2 label to the second characters of words consisting of more than two characters, whereas the BB_2B_3IES gives the B_2 and the B_3 labels to the second and third characters of words comprised of more than three characters.

Table 2 shows a sample text annotated with the seven SRs which will be used in this work. In addition to the *IOB*2 and the *IOBES* SRs that have been commonly used in NER, we also use the *IOE*2 SR to investigate whether it is better to distinguish the beginning or the end of named entities. The *IO* SR is adopted as the simplest SR that actually does not perform any segmentation. Because two named entities are not likely to appear consecutively, we can recognize named entities as a sequence of tokens that have a same label. The *BI*, the *IE* and the *BIES* SRs, to the best of our knowledge, were proposed for WS and have not been used for NER. We apply these SR to NER by regarding the *O* label as a semantic class and augmenting it with the remaining segment labels. This application is based on the observation that tokens appearing around named entities are not random words. In this example, for instance, the left round bracket appears between the full name of a gene and its abbreviation and the right round bracket occurs after the abbreviated gene name. Therefore, it is worth differentiating these tokens from the others by assigning separate labels.

[1] The *BI* SR identifies characters at the **B**eginning and **I**snide of words.

Table 1
Definition of SRs for NER, WS and SP.

Task	SR type	Segment labels	Examples
NER	*IOB2*	*B*, *I*, *O*	B, BI, BII, . . . , O
	IOBES	*S*, *B*, *I*, *E*, *O*	S, BE, BIE, BIIE, . . . , O
SP	*IOB2*	*B*, *I*, *O*	B, BI, BII, . . . , O
	IOE2	*I*, *E*, *O*	E, IE, IIE, . . . , O
	IOB1	B^*, *I*, *O*	I, II, . . . , B^*, B^*I, B^*II, . . . , O
	IOE1	*I*, E^*, *O*	I, II, . . . , E^*, IE^*, IIE^*, . . . , O
	IOBES	*S*, *B*, *I*, *E*, *O*	S, BE, BIE, BIIE, . . . , O
WS	*BI*	*B*, *I*	B, BI, BII, . . .
	BIS	*S*, *B*, *I*	S, BI, BII, . . .
	BIES	*S*, *B*, *I*, *E*	S, BE, BIE, BIIE, . . .
	BB_2IES	*S*, *B*, B_2, *I*, *E*	S, BE, BB_2E, BB_2IE, . . .
	BB_2B_3IES	*S*, *B*, B_2, B_3, *I*,*E*	S, BE, BB_2E, BB_2B_3E, BB_2B_3IE, . . .

Table 2
A sample text annotated with various SRs. (NEs are in italic face font.)

Text	*IO*	*IOB2*	*IOE2*	*IOBES*	*BI*	*IE*	*BIES*
Gamma	I-gene	B-gene	I-gene	B-gene	B-gene	I-gene	B-gene
glutamyl	I-gene	I-gene	I-gene	I-gene	I-gene	I-gene	I-gene
transpeptidase	I-gene	I-gene	E-gene	E-gene	I-gene	E-gene	E-gene
(	O	O	O	O	B-O	E-O	S-O
GGTP	I-gene	B-gene	E-gene	S-gene	B-gene	E-gene	S-gene
)	O	O	O	O	B-O	I-O	B-O
activity	O	O	O	O	I-O	I-O	I-O
in	O	O	O	O	I-O	I-O	I-O
the	O	O	O	O	I-O	I-O	I-O
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .

2.2. Relation among segment representations

Conceptually, only two segment labels are necessary (e.g. *B-gene* and *I-gene* for gene names) to distinguish segment boundaries unambiguously. However, many words tend to appear at specific positions not at random places. For example, the names of location often end with the words such as "Street", "Road" and "Avenue" and the names of companies are frequently followed by the phrases such as "Corporation" and "Co., Ltd." Therefore, complex SRs that can capture these characteristics of target segments are able to create a more informative feature space than simple SRs. Xue (2003) articulated that choosing a suitable SR is a task-specific problem that depends on the characteristics of segments and the size of available training data.

Segment labels of a complex SR often denote more specific positions than those of a simple SR. While every pair of any SRs can be inter-convertible if enough context information (segment labels of neighboring tokens) is provided, some of them are *deterministically* mappable by looking at only current labels. For example, to convert the *IOBES* SR to the *IOB2* SR, we can simply map the *B* and the *S* labels of the *IOBES* SR to the *B* label of the *IOB2* SR, the *I* and the *E* labels to the *I* label. Fig. 1 shows the hierarchical relation among the seven SRs used in the previous example in Table 2. In this figure, a complex SR can be deterministically mapped to a simple SR if they are connected by directed arrow(s). Table 3 shows how to map the segment labels of the *BIES* SR to those of simpler six SRs.

The existing sequence labeling framework using the Viterbi algorithm assumes the Markov property for computational tractability. Therefore, it is impossible to use arbitrary context information for mapping segment labels of one SR to those of another SR. However, we can avoid this problem by considering only a subset of SRs that can be deterministically mapped

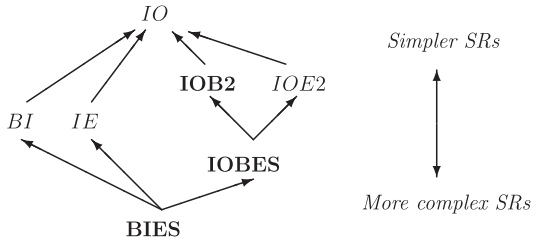

Fig. 1. The hierarchical relation among the seven SRs.

Table 3
Mapping segment labels of the *BIES* SR to those of the simpler six SRs. *Non-segment* is a sequence of tokens tagged with the *O* label.

	Segment				Non-segment			
BIES	S	B	I	E	S	B	I	E
⇓								
BI	B	B	I	I	B	B	I	I
IE	E	I	I	E	E	I	I	E
IOBES	S	B	I	E	O	O	O	O
IOB2	B	B	I	I	O	O	O	O
IOE2	E	I	I	E	O	O	O	O
IO	I	I	I	I	O	O	O	O

from one SR to another SR as shown in Fig. 1. For example, when we use the *IOBES* SR, we can utilize the features created from not only this SR but also the other SRs which can be deterministically mapped from it (e.g. *IOB2*, *IOE2* and *IO*).

3. The effects of different segment representations on NER

To investigate the effects of different SRs on NER, we performed a preliminary experiment on the BioCreative 2 gene mention recognition (BC2GMR) task (Smith, 2008). For the experiment, we trained seven models with seven different SRs (*IO*, *IOB2*, *IOE2*, *BI*, *IE*, *IOBES* and *BIES*), but with the same textual cues.[2] Among these SRs, the *BI*, the *IE* and the *BIES* SRs were originally designed for the WS task and do not use the *O* label. We assumed a sequence of continuous *O* labeled tokens as a kind of special named entities, namely *O*-class named entity, and gave them separate *O* labels to apply these SRs to the NER tasks. For example, the *BI* SR uses the *B-O* and *I-O* labels instead of the *O* label.

For machine learning, we implemented a linear-chain CRFs with the L-BFGS algorithm.[3] Lafferty, McCallum, and Pereira (2001). defines a linear chain CRFs as a distribution:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x})$$

where $\mathbf{x} = \langle x_1, x_2, \ldots, x_T \rangle$ is an input token sequence, $\mathbf{y} = \langle y_1, y_2, \ldots, y_T \rangle$ is an output label sequence for $\mathbf{x}$, $Z(\mathbf{x})$ is a normalization factor over all label sequences, T is the length of the input and output sequences, K is the number of features, f_k is a feature and λ_k is a feature weight for the f_k.

In a linear-chain CRFs, f_k is either a transition feature or a state feature. For example, a transition feature[4] f_i, which represents the transition from the *B-gene* label to the *E-gene* label of the *IOBES* SR, can be defined as

$$f_i(y_{t-1}, y_t, \mathbf{x}) = \begin{cases} 1 & ((y_{t-1} = \mathbf{B} - \mathbf{gene}) \wedge (y_t = \mathbf{E} - \mathbf{gene})) \\ 0 & (otherwise) \end{cases}$$

and a state feature[5] f_j, which indicates that the current state is *E-gene* and its corresponding input token is "protein", can be defined as

$$f_j(y_{t-1}, y_t, \mathbf{x}) = \begin{cases} 1 & ((y_t = \mathbf{E} - \mathbf{gene}) \wedge x_t = (\mathbf{protein})) \\ 0 & (otherwise). \end{cases}$$

Training a linear chain CRFs model is equivalent to find a set of feature weights which maximize a model log-likelihood for a given training data. However, it is often necessary to use *regularization* to avoid overfitting. We use the following model log-likelihood formula (Sutton & McCallum, 2007). The last term is for regularization.

$$l(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k\left(y_{t-1}^{(i)}, y_t^{(i)}, \mathbf{x}^{(i)}\right) - \sum_{i=1}^{N} logZ(\mathbf{x}^{(i)}) - C \sum_{k=1}^{K} \lambda_k^2$$

The parameter C determines the strength of regularization and it can be chosen by using development data. A smaller C value will result in a model that fits training data better than a bigger C value, while it is more likely to be overfitting. In the preliminary experiment, we reserved the last 10% of the original training data as the development data for tuning the C value. We examined ten C values[6] for each model and used the best performing C value for evaluation on the test data.

We used features generated from input tokens, lemmas, POS-tags, chunk-tags and gazetteer matching results. The detailed explanation of the feature set is in Section 5.

---

[3] http://www.chokkan.org/software/liblbfgs/.

[4] A transition feature is a combination of previous and current labels. An input token sequence is not used for transition features in the current implementation.

[5] A state feature is a combination of a current label and a textual cue created from a sequence of input tokens within a context window.

[6] These C values are 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, and 2^{4}.

Table 4
The performance of the seven models on the BC2GMR task.

Model	#Labels	Precision	Recall	F1-score
IO	2	88.13	81.39	84.63
IOB2	3	88.73	83.07	85.81
IOE2	3	88.79	83.48	86.05
BI	4	89.64	83.10	86.25
IE	4	89.12	82.15	85.49
IOBES	5	89.83	83.53	86.56
BIES	8	90.58	83.26	86.77

Table 5
The comparison of tagging results between the *IO* and *BIES* models.

From IO	# Of instances	To BIES	# Of instances
TP	5153	TP	4899
		FN	254
FN	1178	TP	372
		FN	806
TN	–	TN	–
		FP	235
FP	694	TN	381
		FP	313

3.1. Evaluation based on standard performance measures

The seven models are evaluated in standard performance measures: precision, recall and F1-score. As shown in Table 4, precision tends to improve as the number of labels increases. On the other hand, recall does not exhibit such a clear tendency where the *IOE*2 and *IOBES* models achieve the higher recall than other models. If we follow the conventional approach, the *BIES* SR, which has not been used for NER, will be most suitable for this corpus.

3.2. Evaluation based on the difference of tagging results

Although the evaluation in standard performance measures demonstrated that the *BIES* SR is most suitable for this corpus, we found that the tagging results of these seven models are quite varied. Table 5 shows how the tagging results change when the SR alters from the simplest one (*IO*) to the most complex one (*BIES*) in terms of true positive (TP), false negative (FN), true negative (TN) and false positive (FP). Since the *BIES* model clearly outperforms the *IO* model, we anticipate that the *BIES* model will produce more correct tagging results. The *BIES* model actually corrects 372 false negatives and 381 false positives of the *IO* model. However, surprisingly, it introduces new 254 false negatives and 235 false positives which are non-negligible amount of errors.

This analysis suggests that different SRs can produce feature spaces which are complementary to each other; and using multiple SRs is highly likely to improve NER performance. In the following section, we explain how to integrate multiple SRs into a CRF-based NER model.

4. The proposed method

This section presents a feature generation method which incorporates multiple SRs into a single CRF-based NER model. An expanded feature space created with the proposed method allows a model to exploit both high discriminative power of complex SRs and robustness of simple SRs against the data sparseness problem.

In Section 4.1, we explain the mapping relation of the SRs, and design four groups of SRs for the proposed method. Section 4.2 describes a modified linear chain CRFs model which can automatically generate and evaluate features of multiple SRs. In Section 4.3, we show that a simple model computation after training makes the tagging speed of a proposed model using multiple SRs as fast as the conventional model using the most complex SR of the proposed model.

4.1. The mapping relation of segment representations

In Section 2.2, we presented a hierarchical relation among seven SRs that can be deterministically mappable and explained how to exploit multiple SRs without violating the Markov property. We call the most complex SR among all SRs used

Table 6
Main and additional SRs used for four groups.

Group	Main SR	Additional SR
IOB2+	*IOB2*	*IO*
IOBES+	*IOBES*	*IOB2*, *IOE2*, *IO*
BIES+	*BIES*	*BI*, *EI*, *IOBES*, *IOB2*, *IOE2*, *IO*
BIES&IO	*BIES*	*IO*

for a model as a *main SR*, and the other SRs as *additional SRs*. A conventional NER model can be interpreted as a model using only a main SR. For the experiment, we selected two most popular SRs, *IOB2* and *IOBES*, and the most complex one, *BIES*, as the main SRs. As additional SRs, we basically use all deterministically mappable SRs to show the maximum effect of the proposed method. Three groups of SRs are shown in Table 6 and their names are marked with '+' symbol. In addition, we trained a model using only the *BIES* and the *IO* SRs, which are the most complex and the simplest SRs. This will minimize the increase of the total number of features, while allowing the model exploit complementary feature information of SRs in very different granularities.

4.2. A modified linear chain CRFs model for multiple segment representations

In Section 3, we briefly introduced a linear chain CRFs. To enable a model to use features generated from multiple SRs, we define a set of feature sets, $\Gamma = \{F_l\}$, where F_l is a set of features generated from the l SR. Then, we re-define a model as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp \sum_{t=1}^{T} \sum_{F_l \in \Gamma} \sum_{f \in F_l} \lambda_f f(y_{t-1}, y_t, \mathbf{x})$$

where f is a feature of a feature set F_l of the SR l, and λ_f is a feature weight for the feature f. This modified CRFs model can use features generated from multiple SRs.

However, we need to remind that a label sequence $\mathbf{y}$ belongs to the main SR. Therefore, it cannot directly evaluate the features of additional SRs. For example, a model, which uses the *IOBES* as its main SR and the *IOB2* as its additional SR, may have a transition feature $f'_i \in F_{IOB2}$ as below. (To avoid confusions, we explicitly use the name of the SR as superscript to which a label belongs.)

$$f'_i(y^{IOBES}_{t-1}, y^{IOBES}_t, \mathbf{x}) = \begin{cases} 1 & \left(\left(y^{IOBES}_{t-1} = \mathbf{B} - \mathbf{gene}^{IOB2}\right) \right. \\ & \left. \wedge \left(y^{IOBES}_t = \mathbf{I} - \mathbf{gene}^{IOB2}\right)\right) \\ 0 & (otherwise) \end{cases}$$

This feature cannot be directly evaluated because the input argument labels (y_{t-1} and y_t) are of the main SR (*IOBES*) while the feature is of an additional SR (*IOB2*).

To solve this problem, we define a label conversion function, $g^l(y)$ which converts a label y of the main SR into a label y' of the SR l. Then the transition feature above can be re-defined as

$$f'_i(y^{IOBES}_{t-1}, y^{IOBES}_t, \mathbf{x}) = \begin{cases} 1 & \left(\left(g^{IOB2}(y^{IOBES}_{t-1}) = \mathbf{B} - \mathbf{gene}^{IOB2}\right) \right. \\ & \left. \wedge \left(g^{IOB2}(y^{IOBES}_t) = \mathbf{I} - \mathbf{gene}^{IOB2}\right)\right) \\ 0 & (otherwise). \end{cases}$$

The same modification applies to state features. For example, a state feature $f'_j \in F_{IOB2}$ can be re-defined as

$$f'_j(y^{IOBES}_{t-1}, y^{IOBES}_t, \mathbf{x}) = \begin{cases} 1 & \left(x_t = (\mathbf{protein}) \right. \\ & \left. \wedge \left(g^{IOB2}(y^{IOBES}_t) = \mathbf{I} - \mathbf{gene}^{IOB2}\right)\right) \\ 0 & (otherwise). \end{cases}$$

For $g^l(y)$, we use a *deterministic* conversion function that works as explained in Section 4.1. This mapping function allows us to use well-established algorithms for training a model.

4.3. Boosting up tagging speed

A models using the proposed method generates more features and it inevitably slows down training speed. However, we can speed up the tagging speed of this model as fast as the model using only the main SR. The proposed method uses a deterministic label mapping function. It means that we know what kinds of features of additional SRs will be triggered for every feature of the main SR. By calculating the sum of feature weights that always appear together in advance and using it as the new weights for the main SR, the model can work as if it uses only the main SR. The model size and tagging speed will be identical to the model actually trained with the main SR only.

Table 7
Features for the biomedical NER.

Class	Description
Token	$\{w_{t-2}, \ldots, w_{t+2}\} \wedge y_t$, $\{w_{t-2,t-1}, \ldots, w_{t+1,t+2}\} \wedge y_t$, $\{\bar{w}_{t-2}, \ldots, \bar{w}_{t+2}\} \wedge y_t$ $\{\bar{w}_{t-2,t-1}, \ldots, \bar{w}_{t+1,t+2}\} \wedge y_t$,
Lemma	$\{l_{t-2}, \ldots, l_{t+2}\} \wedge y_t$, $\{l_{t-2,t-1}, \ldots, l_{t+1,t+2}\} \wedge y_t$, $\{\bar{l}_{t-2}, \ldots, \bar{l}_{t+2}\} \wedge y_t$, $\{\bar{l}_{t-2,t-1}, \ldots, \bar{l}_{t+1,t+2}\} \wedge y_t$
POS	$\{p_{t-2}, \ldots, p_{t+2}\} \wedge y_t$, $\{p_{t-2,t-1}, \ldots, p_{t+1,t+2}\} \wedge y_t$,
Lemma & POS	$\{l_{t-2}p_{t-2}, \ldots, l_{t+2}p_{t+2}\} \wedge y_t$, $\{l_{t-2,t-1}p_{t-2,t-1}, \ldots, l_{t+1,t+2}p_{t+1,t+2}\} \wedge y_t$
Chunk	$\{c_t, w_{t\_last}, \bar{w}_{t\_last}, the_{lhs}\} \wedge y_t$
Character	Character 2,3,4-grams of w_t
Orthography	All capitalized, all numbers, contain Greek letters, ... (Detailed explanation of the orthographical features can be found in the related work (Lee et al., 2004))
Gazetteer	$\{g_{t-2}, \ldots, g_{t+2}\} \wedge y_t$, $\{g_{t-2,t-1}, \ldots, g_{t+1,t+2}\} \wedge y_t$, $\{g_{t-2}l_{t-2}, \ldots, g_{t+2}l_{t+2}\} \wedge y_t$, $\{g_{t-2,t-1}l_{t-2,t-1}, \ldots, g_{t+1,t+2}l_{t+1,t+2}\} \wedge y_t$

Table 8
Explanation of symbols used for features (see Table 7).

Symbol	Description
w_t	A tth word
$\bar{w}_t$	A normalized tth word. If w_t contains numbers, continuous numeric parts are conflated into a single zero (e.g. "p53" to "p0"). If w_t is a non-alphanumeric character, it becomes an under-bar symbol (e.g. "-" to "_").
l_t	A tth lemma
$\bar{l}_t$	A normalized tth lemma
p_t	A tth POS-tag
c_t	the chunk type of w_t
w_{t_last}	The last word of a current chunk
$\bar{w}_{t\_last}$	The normalized last word of a current chunk
the_{lhs}	If 'the' exists from the beginning of a current chunk to w_{t-1}
g_t	Gazetteer label for the tth word

5. Experiments

The proposed method is evaluated on two NER tasks in different domains: the BioCreative 2 gene mention recognition (BC2GMR) task (Smith, 2008) and the CoNLL 2003 NER shared task (Tjong Kim Sang & De Meulder, 2003).

We added a necessary functionality[7] into our implementation of a linear-chan CRFs so that it produces features with a given set of SRs as shown in Table 6. For machine learning, the L-BFGS algorithm is chosen. The training process terminates if the variance of the model likelihood of the latest twenty models is smaller than 0.0001 or if it reaches the maximum number of iterations, 2000.

5.1. NER in the biomedical domain

To prepare the experiment, we performed the following pre-processing. First, the corpus is tokenized based on the same tokenization method in the previous work (Leaman & Gonzalez, 2008). Although this tokenization method produces more tokens than the Penn Treebank tokenization,[8] the output is very consistent: that is, no named entities begin or end in the middle of a token. Second, the tokenized texts are fed into the GENIA tagger (Tsuruoka & Tsujii, 2005) to obtain lemmatization, POS-tagging and shallow parsing information. Lastly, we applied two gazetteers compiled from the EntrezGene (Maglott, Ostell, Pruitt, & Tatusova, 2005) and the Meta-thesaurus of the Unified Medical Language Systems (UMLS) (Bodenreider, 2004).

Features are extracted from tokens, lemmas, POS-tags, chunk-tags and gazetteer matching results. The feature set for our biomedical NER system is listed in Table 7 and the symbols used for the features are explained in Table 8. Most of these features are common for biomedical NER tasks (Leaman & Gonzalez, 2008; Lee, Hwang, Kim, & Rim, 2004; Nadeau & Sekine, 2007), while chunk features and several orthographic features are newly added. The L2-regularization parameter (C) is optimized by using the first 90% of the original training data as the training data and the rest 10% as the development data. Ten C values[9] are tested on the development data and the best-performing one is chosen for each model.

[7] While this functionality is not difficult to implement, we found that incorporating it into a publicly available CRF toolkit, CRFSuite (Okazaki, 2007), is not a simple task because of its optimized code for speed.

[8] http://www.cis.upenn.edu/treebank/tokenization.html.

[9] These C values are 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, and 2^{4}.

Table 9
The performance on the BC2GMR task. AFI stands for the average number of feature instances per feature in the training data. #Feat means the number of unique features (million).

Model	Precision	Recall	F1-score	AFI	#feat
IO	77.67 (88.13)	70.10 (81.39)	73.69 (84.63)	17.00	4.2
IOB2 (BM)	78.60 (88.73)	72.12 (83.07)	75.22 (85.81)	16.38	6.4
IOE2	78.64 (88.79)	72.56 (83.48)	75.48 (86.05)	16.29	6.4
BI	79.31 (89.64)	72.04 (83.10)	75.50 (86.25)	15.06	8.5
IE	79.15 (89.12)	71.54 (82.15)	75.15 (85.49)	15.02	8.5
IOBES	79.59 (89.83)	72.58 (83.53)	75.93 (86.56)	15.68	10.6
BIES (best BM)	**80.70** (90.58)	72.58 (83.26)	76.42 (86.77)	13.44	16.9
IOB2+	78.56 (88.51)	72.39 (83.21)	75.35 (85.78)	16.69	10.9
IOBES+	79.93 (89.88)	72.86 (83.65)	76.24 (86.66)	16.33	27.5
BIES+ (best PM)	80.61 (90.18)	**73.80** (84.17)	**77.05** (87.08)	15.60	61.4
BIES& *IO*	80.40 (90.00)	73.54 (84.00)	76.82 (86.90)	15.01	21.2

Highest scores are shown in bold font.

The BC2GMR task provides two types of annotations: the main and the alternative annotations. A gene name in the main annotation may have alternative names that are semantically equivalent but have different textual spans. Therefore, one can say that the official evaluation using both of them is based on a relaxed-match criterion. Table 9 summarizes the experimental results of seven models using a single SR (the conventional models) and four models using multiple SRs (the proposed models) based on the strict-match and the relaxed-match (in a pair of parentheses). We use the strict-match results for comparing the models because the detection of correct entity boundaries is also an important sub-task of NER and the relaxed-match results can underestimate it.

Conventional models tend to improve precision as they use more complex SRs than the baseline models[10] (BM). The best baseline model (best BM) records the highest precision that is notably higher than that of the BM. However, recall does not exhibit such an obvious tendency. For example, the recall of the best BM is almost identical to that of the *IOE*2 and the *IOBES* models.

Proposed models improve both precision and recall when they use complex SRs. In addition, every proposed model outperforms the conventional models that employ one of the SRs used by the proposed model. The best proposed model (best PM) achieves higher recall (1.22%) and comparable precision (−0.09%) to the best BM. The improvement of recall is an important merit of the proposed method because NER models frequently suffer from low recall due to an asymmetric label distribution where the *O* labels dominate the other labels (Kambhatla, 2006) in training data. Considering that the only difference of the proposed models from the conventional ones is a set of SRs for feature generation, we can conclude that the proposed method effectively remedies the data sparseness problem of using complex SR while takes advantage of its high discriminative power. This conclusion is also supported by the relation between the average number of feature instances per feature (AFI) and the number of features (#feat). For example, the best PM has about 20% higher AFI (15.60) than the best BM (13.44), whereas it has almost four times more features than the best BM.

To verify whether these improvements are meaningful, we performed the statistical significance test using the bootstrap re-sampling method (Smith, 2008), which is commonly used for NER. Table 10 presents the estimated p values for the proposed models (the top row) against the conventional models (the leftmost column). In most cases, the proposed models have the p values lower than 0.05. Comparing a proposed model and its counterpart model, which uses the main SR of the proposed model, the p value decreases as the proposed model integrates more SRs of different granularity. As a result, the *BIES*+ model has the p value lower than 0.05 whereas the *IOB*2+ and the *IOBES*+ do not. Interestingly, the *BIES*&*IO* model also rejects the null hypothesis against the best BM given the threshold p value 0.05. Considering that both the *BIES*&*IO* and the *IOB*2+ models use only two SRs, integrating SRs of very different granularities is more effective than that of similar granularity.

We also show how the tagging results change when the proposed method is applied. For the sake of analysis, we use two conventional models, *BIES* and *IO*, and the proposed model, *BIES*&*IO*, that utilizes the SRs of the *IO* and *BIES* models. In Table 11, the tagging results of the two conventional models are divided into two groups depending on whether they make the same predictions or not. Then, we investigated what kinds of predictions the *BIES*&*IO* model makes. The upper table titled with "Agreed" shows the tagging results of the *BIES*&*IO* model when the *IO* and *BIES* models make the same predictions. In most cases, the *BIES*&*IO* model makes the same predictions with the conventional models ($\geqslant$96%). In the lower table titled with "Disagreed", the two conventional models make different predictions and only one of them is correct. We can see that the tagging results of the *BIES*&*IO* model tend to follow the results of the *BIES* model (from about 78% to 91%). However, the *BIES*&*IO* model makes less predictions same to the *BIES* model when it makes wrong predictions (from about 90% to 80%), even though the *BIES* model clearly outperforms the *IO* model by 2.73 points in F1-score.

We present several gene names that are correctly recognized obviously by the help of the proposed method. For example, *BIES*&*IO* model correctly recognized a gene name *mouse and human HPRT genes*, whereas the *BIES* model recognized only a part of it, *human HPRT genes*. Both words, *mouse* and *human*, mostly appear at the beginning of a gene name (94 vs. 25 times in the training data), whereas rarely in the middle of a gene name (7 vs. 3 times). The *BIES* model is likely to give the *B* label to

[10] The baseline model uses the most popular SR, *IOB*2.

Table 10
The estimated p values between the proposed models and the conventional models. p values lower than 0.05 are in boldface.

	IOB2+	*IOBES+*	*BIES+*	*BIES&IO*
IO	**0.0000**	**0.0000**	**0.0000**	**0.0000**
IOB2	0.2174	**0.0001**	**0.0000**	–
IOE2	–	**0.0075**	**0.0000**	–
BI	–	–	**0.0000**	–
IE	–	–	**0.0000**	–
IOBES	–	0.0970	**0.0000**	–
BIES	–	–	**0.0039**	**0.0219**

Table 11
The tagging results of two conventional models (*BIES* and *IO*) and a proposed model (*BIES&IO*). The number of named entities is shown in parenthesis.

BIES vs. *IO*	*BIES&IO*	
1. Agreed		
TP vs. TP (4139)	TP:99.42% (4115)	FN:0.58% (24)
TN vs. TN (–)	TN:–% (–)	FP: –% (65)
FP vs. FP (702)	FP:96.58% (678)	TN:3.42% (24)
FN vs. FN (1437)	FN:95.96% (1379)	TP:4.04% (58)
2. Disagreed		
TP vs. FN (456)	TP:91.23% (416)	FN:8.77% (40)
TN vs. FP (574)	TN:88.50% (508)	FP:11.50% (66)
FP vs. TN (397)	FP:82.12% (326)	TN:17.88% (71)
FN vs. TP (299)	FN:77.59% (232)	TP:22.41% (67)

human because it occurs almost four times more than *mouse* in the training data. On the other hand, the *IO* model, which correctly recognized this gene name, does not experience this problem because it can give the same *I* label to these words. We think that the *BIES&IO* model successfully recognized this gene name because it could exploit the features generated with the *IO* SR. There are similar cases where the *BIES&IO* and *IO* models correctly recognized gene names such as *serum insulin* and *type I and II collagen*, while the *BIES* model recognized only the last word, *insulin* and *collagen*. These last words often appear as gene names by themselves (33 among 44 times for *insulin* and 8 among 16 times for *collagen*). Therefore, the *BIES* model is likely to give the *S* label for these words.

However, incorporating the features of the *IO* model can cause difficulties in finding correct entity boundaries. For example, the *BIES* model correctly recognized gene names such as *Oshox1*, *phP1* and *Pms-*, whereas the *BIES&IO* and *IO* models recognized incorrect textual spans as *upstream Oshox1 binding sites*, *phP1 mutation* and *Pms*.

Next, we examined the effect of the proposed method based on the size of available training data. Models are trained on the first 10%, 20%, 40% and 100% of the original training data that is 15,000 sentences in total. Regularization parameters are tuned by using the last 10% of the original training data as the development data. For the models using 100% of the original training data, they are first trained on the first 90% portion for parameter tuning and the final models are trained on the full training data.

Fig. 2 shows the precision of the three proposed models (*IOB2+*, *IOBES+* and *BIES+*) and their counterpart model (*IOB2*, *IOBES* and *BIES*). The precision of a proposed model is almost identical to that of its counterpart model at each point. In addition, the models using more complex SRs achieve higher precision than the models using simpler ones regardless of application of the proposed method. This result shows that precision is mostly determined by the granularity (the number of segment labels) of the most complex SR employed by a model.

However, complex SRs can cause negative impact on recall. For example, in Fig. 3, the *BIES* model records the lowest recall when the size of training data is 10% and 20% of the original training data. The low recall of the *BIES* model at beginning is due to the insufficient training data considering that it achieves similar or higher recall than other two conventional models as the size of training data reaches 40%. A proposed model, *BIES+*, on the contrary, achieves almost highest recall from the beginning and outperforms all other models as the size of training data increases. Therefore, by using the proposed method, we cannot only take advantage of high discriminative power of complex SRs but also boost recall by incorporating simple SRs.

In Table 12, we compare the best proposed model (best PM) to the systems participated in the BC3GMR competition. The comparison is just for reference since BC2 systems exploit various techniques and external resources such as model ensemble, post-processing, abbreviation detection and resolution, semi-supervised learning, gazetteers and unlabeled data. This information is summarized in the last column of Table 12. The best PM is also compared with BANNER[11] (Leaman & Gonzalez,

[11] http://cbioc.eas.asu.edu/banner/.

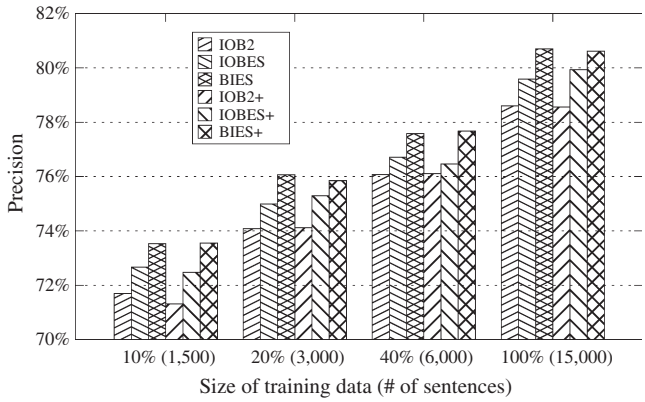

Fig. 2. The effect of the proposed method on precision based on the training data size.

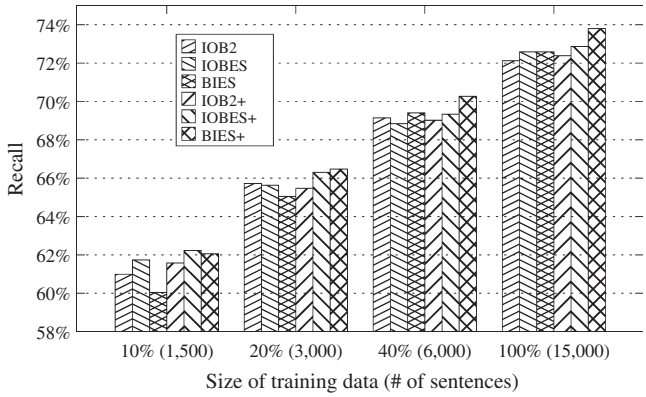

Fig. 3. The effect of the proposed method on recall based on the training data size.

Table 12
The performance comparison to the other systems based on the official evaluation. BC2-*x* means a system participated in the BC2GMR competition and ranked at the *x*th position. Add. tech. column shows additional techniques used for these systems, A: Abbreviation resolution, E: Ensemble classifier, G: Gazetteer, P: Post-processing, S: Semi-supervised method, and U: Unlabeled data.

Systems	Precision (%)	Recall (%)	F1-score (%)	Add. tech.
Li et al. (2009)	90.52	87.63	89.05	E, G, U
Hsu et al. (2008)	88.95	87.65	88.30	E, G
BC2-1st	88.48	85.97	87.21	G, P, S
BIES+ (best PM)	90.18	84.17	87.08	G
BC2-2nd	89.30	84.49	86.83	E, G, P
BIES (best BM)	90.58	83.26	86.77	G
BC2-3rd	84.93	88.28	86.57	E
BC2-6th	82.71	89.32	85.89	G, P
IOB2 (BM)	88.73	83.07	85.81	G
BANNER	87.18	82.78	84.93	A, P
BC2-7th	86.97	82.55	84.70	A, G

2008), a publicly available system for biomedical NER tasks, and two state-of-the art systems (Hsu et al., 2008; Li, Lin, & Yang, 2009). It is placed between the 1st and 2nd ranked BioCreative 2 systems. The overview paper of BioCreative 2 competition states that a difference of 1.23 or more in F1-score is statistically significant ($p < 0.05$). Therefore, we can conclude that our system rivals to the top performing system in the BioCreative 2 competition. Two recently proposed state-of-the-art systems (Li et al., 2009; Hsu et al., 2008) achieve higher performance than the best PM. They obtain such a high performance by combining the results of multiple NER models. The best component NER model in each state-of-the-art system achieves 86.20 and 87.12 in F1-score respectively. Therefore, we can say that the best PM achieves the state-of-the-art performance as a single NER model. In addition, there is a possibility that even better performance can be obtained by integrating the best PM into these systems.

Table 13
The performance on the CoNLL NER data.

Model	Precision (%)	Recall (%)	F1-score (%)	AFI	# of feat
IO	83.50	82.14	82.81	28.88	3.10 M
IOB2 (BM)	83.91	82.61	83.25	27.84	5.57 M
IOE2	83.85	82.38	83.11	27.79	5.57 M
IOBES	83.75	82.56	83.15	26.79	10.52 M
BI	83.73	82.56	83.14	26.01	6.19 M
IE (best BM)	83.77	82.86	83.31	25.46	6.19 M
BIES	83.45	82.67	83.06	23.02	12.38 M
IOB2+	84.30	82.99	83.64	28.35	8.67 M
IOBES+	84.34	83.18	83.76	27.75	24.76 M
BIES+ (best PM)	**84.35**	**83.50**	**83.92**	26.41	49.52 M
BIES& IO	83.93	83.07	83.50	25.60	15.47 M

The best PM achieves the highest precision, recall and F1-score.

While the proposed method produces a more desirable feature space for a model and improves its performance, the increase of the number of features inevitably slows down training speed. The last column in Table 9 shows the number of features for each model that is proportional to the training speed. The most complex model, *BIES+*, uses more than 60 million features; and the training speed is almost ten times slower than the *IOB2* baseline model. As a simple speed up technique, the *BIES&IO* model is trained with only two SRs, *BIES* and *IO*. Surprisingly, this model achieves comparable performance to the *BIES+* model with a relatively small increase of training time. Therefore, the *BIES&IO* model would be a good alternative to the conventional models when the training speed is important.

5.2. NER in the general domain

The proposed method is also evaluated on the CoNLL 2003 NER shared task data which is a general domain NER corpus. Features used in the study (Kazama & Torisawa, 2007) are adopted in this experiment. We used the POS and the chunking information originally provided in the CoNLL training data. However, gazetteers are not employed to observe the effects of our proposed method in isolation.

Table 13 shows the experimental results. The *IE* model achieves the best F1-score in this task. However, the difference compared to other models is not so significant, except the *IO* model. In addition, as a SR becomes more complex, the overall performance begins to decrease as shown with the *IOB2*, *IOBES* and *BIES* models. The size of the training data could be a reason because the number of named entities is quite small. For example, named entities of the *miscellaneous* class only appear 3,438 times, whereas the training data of the BioCreative 2 corpus has almost 18,000 named entities of the single class, *gene*. In addition, the average number of feature instances per feature (AFI) in the training data drops steeply as the granularity of a SR increases as shown in the fifth column.

When the proposed method is applied, the performance of the proposed models (*IOB2+*, *IOBES+*, *BIES+* and *BIES&IO*) consistently improves. Especially, the *BIES+* model achieves the best performance for the test data while its corresponding baseline model *BIES* records the worst. Since the results are very similar to that of the previous experiment, we omit the detailed analysis on this task.

6. Conclusion & future work

In this paper, we presented a feature generation method for incorporating multiple SRs into a single CRFs model. Our method creates a more desirable feature space; therefore, a model can exploit both features of complex SRs which provide high discriminative power and features of simple SRs which alleviate the problems that can be caused by the data-sparseness. Furthermore, we explained how a model computation after training can make the tagging speed of a model using the proposed method as fast as a model using a single SR.

The proposed method is evaluated on two NER tasks of biomedical and general domain corpora. The results demonstrated that our motivation of using multiple SRs is beneficial to better NER performance. In addition, we provided the results of the statistical significance test to show that the improvement is not by chance, and the detailed performance analysis to explain the effects of using multiple SRs for NER. Lastly, the evaluation on CoNLL NER corpus is also provided to show the domain independence of our proposed method.

Although many researches say that statistical NER systems have reached the plateau of performance, we think that still there is a room for meaningful improvement. Our method suggested one of such ways that use multiple perspectives for a problem. In addition, the proposed method is applicable to any segmentation tasks such as shallow parsing and word segmentation. We expect that the proposed method is also beneficial to these tasks too because the proposed model using multiple SRs exhibited better performance than the best conventional model.

References

Bodenreider, O. (2004). The unified medical language system (UMLS): Integrating biomedical terminology. *Nucleic Acids Research, 32*, D267–D270.

Hsu, C., Chang, Y., Kuo, C., Lin, Y., Huang, H., & Chung, I. (2008). Integrating high dimensional bi-directional parsing models for gene mention tagging. *Bioinformatics, 24*, i286–i294.

Kambhatla, N. (2006). Minority vote: At-least-N voting improves recall for extracting relations. In *Proceedings of COLING-ACL* (pp. 460–466).

Kazama, J., & Torisawa, K. (2007) Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 joint conference on EMNLP and CoNLL* (pp. 698–707).

Kudo, T., & Matsumoto, Y. (2001). Chunking with support vector machines. In *Proceedings of the 2nd conference on NAACL* (pp. 1–8).

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th ICML* (pp. 282–289).

Leaman, R., & Gonzalez, G. (2008). Banner: An executable survey of advances in biomedical named entity recognition. *Pacific Symposium on Biocomputing*, 652–663.

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

Li, Y., Lin, H., & Yang, Z. (2009). Incorporating rich background knowledge for gene named entity classification and recognition. *BMC Bioinformatics, 10*, 223.

Maglott, D., Ostell, J., Pruitt, K. D., & Tatusova, T. (2005). Entrez gene: Gene-centered information at NCBI. *Nucleic Acids Research, 33*, D54–D58.

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes, 30*, 3–26.

Okazaki, N. (2007). CRFsuite: A fast implementation of conditional random fields (CRFs).

Ratinov, L., & Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the 13th conference on CoNLL* (pp. 147–155).

Smith, L. et al (2008). Overview of biocreative II gene mention recognition. *Genome Biology, 9*, S2.

Sutton, C., & McCallum, A. (2007). An introduction to conditional random fields for relational learning. In L. Getoor & B. Taskar (Eds.), *Introduction to statistical relational learning*. MIT Press.

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 HLT-NAACL* (pp. 142–147).

Tjong Kim Sang, E. F., & Veenstra, J. (1999). Representing text chunks. In *Proceedings of the 9th conference on EACL* (pp. 173–179).

Tsuruoka, Y., & Tsujii, J. (2005). Bidirectional inference with the easiest-first strategy for tagging sequence data. In: *Proceedings of the conference on HLT and EMNLP* (pp. 467–474).

Xue, N. (2003). Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese*.

Zhao, H., Huang, C.-N., Li, M., & Lu, B.-L. (2006). Effective tag set selection in chinese word segmentation via conditional random field modeling. In *Proceedings of the 20th Asian pacific conference on language, information and computation* (pp. 87–94).