Using Non-Local Information in Semi-Markov Conditional Random Fields for Biomedical Named Entity Recognition

Abstract

This paper presents an extension semi-CRFs for Named Entity Recognition (NER) which can incorporate long-distance information. We introduce a simple modification of the label set to transfer information about distant entities. Although, in theory, one can train and inference the model within the framework of 1^{st} order semi-CRF models, the straight-forward implementation of the model suffers from a prohibitive computational cos. In this paper, we propose a general framework which enables us to use the state information more efficiently, and conduct a filtering which significantly reduces the candidate states. This framework allows us to use a rich set of features that can capture various characteristics of biomedical NER. Experimental results that our model achieves an F-score of 71.48% on the JNLPBA 2004 shared task without using

external resources and post-processing techniques. We also examine the effect of filtering and long-distance information in experience.

1 Introduction

The rapid increase of information in the biomedical domain has emphasized the need for automated information extraction techniques. In this paper we focus on the Named Entity Recognition (NER) task, which is the first step to tackle more complex tasks such as relation extraction and knowledge mining.

Biomedical NER is harder than general NER tasks. For example, the best F-score in the shared task of biomedical NER in COLING 2004 JNLPBA (Kim et al., 2004) was 72.55% (Zhou and Su, 2004), while the best performance at MUC-6, where systems tried to identify general named entities such as person or organization names, was an accuracy of 95% (Sundheim, 1995). The difficulty in biomedical NER lies in the following characteristics of biomedical named entities. Firstly, named entities tend to have many notational variants. For example, "*NF-kappa B*", "*NF kappa B*" and "*nuclear factor kappa B*" refer to the same entity. We therefore cannot take a dictionary-based approach directly. Secondly, the same term can convery different meanings depending on the context. We should also note that inter-annotator agreement is considerably lower than in general NER tasks. Krauthammer (2004) reported that the inter-annotator agreement rate of human experts was 77.6% when the semantic classes were gene, protein and mRNA, which may sugget the

upper-bound of F-score in a biomedical NER task is around 80%.

In biomedical NER tasks, many of the previous work are based on machine learning techniques. Hidden Markov Model (HMM) were once popular in NER task (Bikel et al., 1997). Recently, Kou et al., (2005) has proposed dict-HMMs, in which dictionary information is encoded as state transition. Maximum Entropy Markov Models (MEMMs) are commonly used for NER and can integrate overlapping features such as word orthographic information and contextual words around the target entities. Among these methods, conditional random fields (CRFs) (Lafferty et al., 2001) have achieved good results in many sequence labeling tasks (Kim et al., 2005; Finkel et al., 2004), presumably because they are free from the label bias problem and can use bi-directional sequential information, which are not directly captured by MEMMs.

Sarawagi and Cohen (2004) have recently introduced semi-Markov conditional random fields (semi-CRFs). They are defined on semi-Markov chains and attach labels to subsequences of a sentence, rather than to the tokens. The semi-Markov formulation allows one to easily construct entity-level features. Since the features can capture the whole characteristics of a subsequence, we can use, for example, the length of an entity or dictionary features which measure the similarity between a candidate segment and the closest element in the dictionary.

In this paper, we present an extended version of

semi-CRFs, in which each entity's probability is conditioned on the preceding states. The preceding states do not have to be adjacent to the entity. We achieve this by modifying the labels for "O" ("O" means outside of the named entity) so that they transfer the information about the preceding states. We also show that we can conduct efficient training and inference by using "feature forest" (Miyao and Tsujii, 2002), with which we pack the states that share the previous information. However, the straight-forward implementation of this framework still suffers from a prohibitive computational cost, because the number of label is increased by the modification, and not a few named entities are long such as eight or ten words, which makes it difficult to enumerate all entity candidates in training and inferencing. We therefore introduce a filtering method that significantly reduces the number of candidate entities by using a "light-weight" classifier. This enables us to construct semi-CRF models for the tasks where entity names are not necessarily short and many class-labels exist at the same time.

2 CRFs and Semi-CRFs

CRFs are undirected graphical models that encode a conditional probability distribution using a given set of features. CRFs allow both discriminative training and bi-directional flow of probabilistic information along the sequence. In NER, we usually use linear-chain CRFs, which define the conditional probability of a state sequence **y** = y_1, ..., y_n given the ob-

served sequence **x** = x_1,...,x_n as:

$$p(\mathbf{y}|\mathbf{x},\lambda) = \frac{1}{Z(\mathbf{x})}\exp(\Sigma_{i=1}^{n}\Sigma_j\lambda_j f_j(y_{i-1},y_i,\mathbf{x},i)), \quad (1)$$

where $f_j(y_{i-1}, y_i, \mathbf{x}, i)$ is a feature function and $Z(\mathbf{x})$ is the normalization factor over all state sequences for the observed sequences. The model parameters are a set of real-valued weights $\lambda = \{\lambda_k\}$, each of which represents the weight of a feature. All feature functions are real-valued and can use adjacent label information.

Semi-CRFs are actually a restricted version of order-L CRFs in which all the labels in a chunk are the same. We follow the definitions in (Sarawagi and Cohen, 2004). Let $\mathbf{s} = \langle s_1, ..., s_p\rangle$ denote a segmentation of x, where a segment $s_j = \langle t_j, u_j, y_j\rangle$ consists of a start position t_j, an end position u_j, and a label y_j. We assume that segments have a positive length which is shorter than or equal to a pre-defined upper bound L ($t_j \leq u_j$, $u_j - t_j + 1 \leq L$) and completely cover the sequence **x** without overlapping, that is, **s** satisfies $t_1 = 1$, $u_p = |x|$, and $t_{j+1} = u_j + 1$ for $j = 1, ..., p-1$. Semi-CRFs define a conditional probability of a state sequence **y** given a observed sequence **x** as:

$$p(\mathbf{y}|\mathbf{x},\lambda) = \frac{1}{Z(\mathbf{x})}\exp(\Sigma_j\Sigma_i\lambda_i f_i(s_j)), \quad (2)$$

where $f_i(s_j) := f_i(y_{j-1}, y_j, \mathbf{x}, t_j, u_j)$ is a feature function and $Z(\mathbf{x})$ is the normalization factor as one in CRFs. The inference problem for semi-CRFs can be solved by using a semi-Markov analog of the usual Viterbi algorithm. The computational cost for semi-CRFs in training and inference is L times larger than CRFs.

3 Using non-Local Information in Semi CRFs

In CRFs and semi-CRFs, we can only use the information on the previous label when defining the features on a certain state or entity. In NER tasks, however, information about a distant entity is often more useful than the information about the preceding state. For example, consider the sentence "... *including Sp1 and CP1.*" where the correct labels of "*Sp1*" and "*CP1*" are both "protein". If the model can use such information, it can classify "*CP1*" as "protein" using the information about "*Sp1*" being "protein", which is not adjacent to "*CP1*." On the other hand, since, in many cases, the previous label of a named entity is "O", which indicates the outside of named entities, information about adjacent labels do not provide useful information [1].

In order to incorporate such non-local information in to semi-CRFs, we propose a simple approach. We convert the label of "O" to "O-protein", "O-DNA" etc, depending on the preceding named entity. Figure 1 shows an example of this conversion. , in which the three labels for the 2nd, 3rd and 4th states are converted from "O" to "O-protein". When we define the features for the 5th state, we can use the information about the preceding entity "protein" by looking at the 4th states. Since this modification changes only the label set, we can do this within the same framework of Semi-CRF models. This modification adds extra information for each state which are not necessarily used in defining each states at

[1] 98.0%, labels of the previous labes of the named entity is "O" in the training data of the shared task in the JNLPBA 2005

defining the features of them, and can be used to transfer any information about the preceding entitles. We should note that not only the number of the kind of information, but also the way of discarding of information determines the computational cost. This will be discussed in 4.1.

In previous work, such long-distance information is usually employed at a post-processing stage. This is because the use of long-distance dependency violates the locality of the model and prevents us from using dynamic programming techniques in training and inference. Skip-CRFs are a direct implementation of long-distance to the model. However, they need to determine the structure for propagating long distance information in advance (Sutton and McCallum, 2004). In a recent study by Finkel et al., (2005), long-distance information is encoded as an independence model, and the inference is performed by Gibbs sampling, which enable us to use the state-of-the-art factored model and train efficiently, but the inference needs much computational cost.

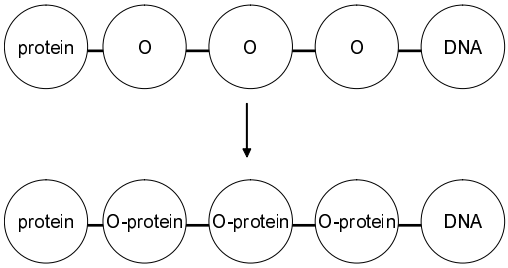

Figure 1: Modification of "**O**" (other labels) to transfer the previous label information.

4 Reduction of Training/Inference Cost

The enrichment of labels and the use of the previous and current label increase the computational cost. If we use a label set consisting of "O" and five non-"O" labels, we have ten labels after the conversion. Moreover, we use the previous and current state in

defining features, the possibility of the number of states which have the same start and end positions are 100 (10^2), this is about 17 times larger than the case that just use current state which needs only 6 states.

The straight-forward implementation of this modeling in semi-CRFs requires a prohibitive computational cost. In biomedical documents, there are quite a few entity names which consist of many words (8-word-names are not rare). This makes it difficult for us to use semi-CRFs, even a original ones, for biomedical NER because we have to set L to be eight or larger, where L is the upper bound of the length of possible chunks in semi-CRFs. Moreover, in order to take into account the dependency between named entities of different classes appearing in a sentence, we need to incorporate multi-class into a single probabilistic model. For example, in the shared task in COLING 2004 JNLPBA (Kim et al., 2004) the number of labels are six ("protein", "DNA", "RNA", "cell_line", "cell_type" and "other"). This also increase the computational cost when we adopt the 1^{st} order semi-CRF.

To realize the model, we propose two methods. The first is using the *feature forest* (Miyao and Tsujii, 2002), which is in short, employing dynamic programming at training "*as much as possible*", and the second is employing a filtering method using a light-classifier to remove unnecessary state candidates.

4.1 Feature Forest

In estimating semi-CRFs, we can use an efficient dynamic programming algorithm, which is similar to

the forward-backward algorithm. The proposal here is a more general framework for estimating sequential conditional random fields.

This framework is based on *the feature forest model*, which was originally proposed for disambiguation models for parsing. A feature forest model is a maximum entropy model defined over *feature forests*, which are abstract representations of an exponential number of sequence/tree structures. A feature forest is an "and/or" graph; in Figure 4.1, circles represent "and" nodes (*conjunctive nodes*) while boxes denote "or" nodes (*disjunctive nodes*). Feature functions are assigned to conjunctive nodes. Each sequence in a feature forest is obtained by choosing a conjunctive node for each disjunctive node. For example, Figure 4.1 represents $3 \times 3 = 9$ sequences, since each disjunctive node has three candidates. It should be noted that feature forests can represent an exponential number of sequences with a polynomial number of conjunctive/disjunctive nodes. If probabilistic events, in our case, sequences of named entity tags, are represented by feature forests, a maximum entropy model of the whole sequence can be estimated using a dynamic programming algorithm.

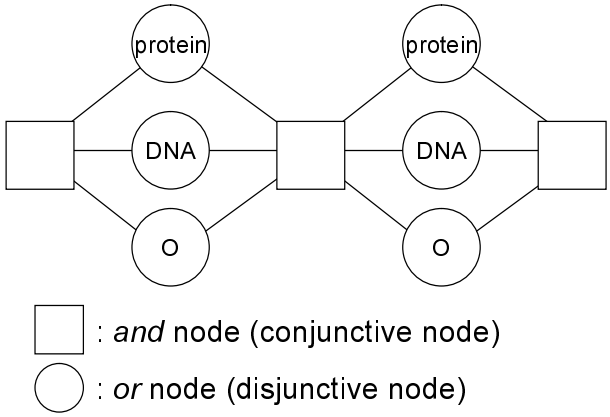

Figure 2: Example of feature forest.

Hence, our concern here is to represent all possi-

ble tag sequences with compact feature forests. Our strategy is to pack "equivalent" states as far as possible. "Equivalent" states mean that they yield equivalent feature functions. When different states yield equivalent feature functions, they may be packed into one node in a feature forest.

For example, suppose the task of tagging "PROTEIN", "DNA", "O-PROTEIN", or "O-DNA", where the latter two tags are "O" tags while distinguishing previous named entity tags. When we simply apply a 1^{st} order semi-CRF, we must distinguish states that have different previous states (Figure 3, left). However, when we want to distinguish "previous named entity tags" rather than the immediate previous states, feature forests can represent these events more compactly. The right figure in Figure 2 shows that the disjunctive nodes following "PROTEIN" and "O-PROTEIN" nodes are packed into one. This is because they share the equivalent information: the previous named entity tag was "PROTEIN". This means that these states yield equivalent feature functions. By this method, we can pack states by ignoring unnecessary information (such as whether the previous state was "O"), and will obtain a more compact representation of named entity sequences.

Another advantage of using feature forests is that we can filter out states beforehand to reduce the size of feature forests. This is because the dynamic programming algorithm of feature forest models is applicable to feature forests with any shape. For example, we can remove "unlikely" states from feature

forests, and this will reduce the training cost. We discuss a method of filtering in the following section.

4.2 Filtering with naive Bayes classifier

We introduce a filtering method to remove low-probability candidate states. This is the first step of our systems. After this filtering step, we construct semi-CRFs on the remaining candidate states. Therefore, the aim of this filtering is to reduce the number of candidate states without producing wrongly removed correct entities. This idea is similar to the method proposed by Tsuruoka and Tsujii, (2005) for chunk parsing, in which implausible phrase candidates are removed beforehand.

We construct a binary naive Bayes classifier using the same training data as those for semi-CRFs. In training and inference, we enumerate all possible chunks (the max length of a chunk is L as in semi-CRFs) and then classify those to correct ones or not. Table 1 lists the features used in the naive Bayes classifier.

Since the purpose of the filtering is to reduces the computational cost, rather than to improve the performance, we chose the acceptance rate, which determines the correct entities to remove the entity, so that the recall of filtering results would be high.

4.3 Features

Table 2 lists the features used in the classifier in semi-CRFs. We give a detailed description about some of the features. "**Length**" and **Length and EndWords** captures the tendency of the length of a named entity which cannot encoded in token-level

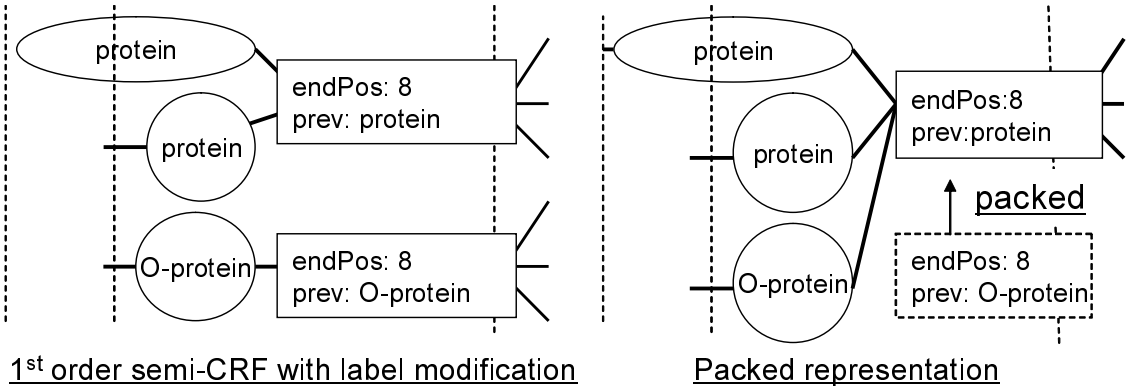

Figure 3: Example of featureforest representation of various semi-CRFs. Left: 1^{st} order semi-CRFs with label modification. The states whose have same end position and label are packed. Right: 1^{st} order semi-CRFs with label modification. The states whose have same end position and prev label information are packed. Since "O-PROTEIN" transfers "PROTEIN" information, which is same as the "PROTEIN" label states, they are packed together.

features. '**Wordshape** are the features indicate capitalization, digitalization and word formation information. "**Previous Label**" is the previous named entity labels (not "O"). For example, when the labels of the chunk sequence are "protein" "others" "others" "DNA", the "**Previous Label**" feature of the 4th entity is "protein". "**Prev State and Prev Word**" are features that capture especially conjunction words such as "*and*" or ", (comma)". For instance, "*OCIM1 and K562*" which "*OCIM1*" and "*K562*" both are assigned cell_line labels. Even if the classifier can determine only "*OCIM1*" as a cell_line , this feature helps to infer that the "*K562*" is assigned as CELL_LINE.. "**Count feature**" is a

Table 1: Features used in the naive Bayes Classifier for the entity candidate: w_s, w_{s+1}, ..., w_e. sp_i is the result of shallow parsing at w_i.

Feature Name	Example of Features
Start/End Word	w_s, w_e
Inside Word	w_s, w_{s+1}, ... , w_e
Context Word	w_{s-1}, w_{e+1}
Start/End SP	sp_s, sp_e
Inside SP	sp_s, sp_{s+1}, ..., sp_e
Context SP	sp_{s-1}, sp_{e+1}

feature which captures the tendency that named entities repeatedly appear in a same sentence.

5 Experiments

5.1 Experimental Setting

Our experiments were performed on the training and evaluation set provided by the shared task in COLING 2004 JNLPBA (Kim et al., 2004). The training data used in this shared task came from the GENIA version 3.02 corpus. In the task there are five semantic labels: protein, DNA, RNA, cell_line and cell_type. The training set consists of 2000 abstracts from MEDLINE, and the evaluation set consists of 404 abstracts. We divided the original training set to 1800 abstracts and 200 abstracts, and the former is used as the training data and the latter is used as the development data. For semi-CRFs, we used

amis[2] for training semi-CRF with feature-forest. We used *GENIA taggar*[3] for POS-tagging and shallow-parsing.

Table 3 shows the detail of the task setting. Table 4 shows the length distribution of the named entities in the training set. We set $L = 10$ for training and evaluation, where L is the upper bound of the length of possible chunks in semi-CRFs. Notice that there are many long entities in the training set.

Table 4: Length distribution of entities in the training set

Length	# entity	Ratio
1	21646	42.19
2	15442	30.10
3	7530	14.68
4	3505	6.83
5	1379	2.69
6	732	1.43
7	409	0.80
8	252	0.49
>8	406	0.79
total	51301	100.00

5.2 Results

We first evaluate the effect of the filtering in the final performance. In this experiment, we cannot examine the performance without filtering using all the training data, because training on all the training data without filtering required much larger memory resources (estimated about 80G Byte) than our experimental environment. We thus compared the result of the classifiers with and without the filtering us-

[2] http://www-tsujii.is.s.u-tokyo.ac.jp/ yusuke/amis/

[3] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/

ing only 2000 sentences as training data. Table 6 shows the result of the total system with different filtering thresholds. The difference between the two filtering threshold is about 1.4% in F-score, this is near to the number of the falsely removed positive entities in the filtering phase (Table 5), which indicates that the result of filtering phase and labeling phase are independent. We also found that the precision of filtering is higher than one without filetring. This result can be explained by that the naive Bayes classifier in filtering phase uses all training data, so it can remove the false positive entities which cannot be detected by semi CRFs using limited training data. So this improvement may not be expected in the setting where we use all training data in comparison with of the filtering methods.

Table 5 shows the filtering performance on the training and evaluation data. The naive Bayes classifiers effectively reduced the number of candidate states with very little falsely removed correct entities.

We next evaluate the effect of long-distance information in the final performance. Table 5.2 shows the result of the classifier performance with previous state information and without. This results indicate that previous information improves the performance.

Table 8 shows the result of the overall performance of our best setting, which uses state features and 1.0^{-15} acceptance rate for filtering. This result is similar to the results of other systems, that is, the performance of cell_line is not good, and the perfor-

mance of the right boundary identification is better than that of the left boundary identification.

Table 7: Overall performance on the evaluation data

development Set			
	Recall	Precision	F-score
Baseline	71.55	78.01	74.64
+ Prev State	72.09	78.47	75.14
Evaluation Set			
	Recall	Precision	F-score
Baseline	72.59	70.16	71.36
+ Prev State	72.65	70.35	71.48

Table 9 shows a comparison between our system and other state-of-the-art systems. Our system has achieved a comparable performance to the state-of-the-art without using external resources and conducting pre/post processing. For example, Zhou et. al (2004) utilize the gazetteers, abbreviation information. Kim et. al (2005) used original Genia corpus to employ other semantic classes information for identification term boundary. Finkel et. al (2004) used gazetters, web-querying, the surrounding abstract, frequency counts from BNC corpus. Settle (2004) used semantic domain knowledge of 17 kinds of lexicons. Our approach and exploitation of external resources/knowledge do not conflict but are com-

Table 9: Comparison with other systems

System	Recall	Precision	F-score
Zhou et. al (2004)	75.99	69.42	72.55
Our system	72.65	70.35	71.48
Kim et.al (2005)	72.77	69.68	71.19
Finkel et. al (2004)	68.56	71.62	70.06
Settles (2004)	69.0	70.0	69.5

plementary ones. We will examine the combination of these techniques as a future work.

6 Conclusion

We presented a probabilistic model that incorporates long-distance dependencies into the semi-CRFs. We also presented a filtering method and an efficient training method which enable us to use not only semi-CRFs which include long named entities, but also non-local information. Our system achieved 71.48% F-score without gazetters, post-processing and external resouces, which is even close to the best performance system which utilize external resources and rule based post-processing. In the next stage of our research, we will make more general probilistic model which can incorporate non-local information. We also hope to apply one method to shallow parsing, in which the entity may be long and local information is important.

References

D. M. Bikel, R. Schwartz, and R. M. Weischedel. 1997. Nymble: a high-performance leraning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*.

Jenny Finkel, Shipra Dingare, Huy Nguyen, Malvina Nissim, Gail Sinclair, and Christopher Manning. 2004. Exploiting context for biomedical entity recognition: From syntax to the web. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. 2004. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of the International Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-04)*, pages 70–75.

Seonho Kim, Juntae Yoon, Kyung-Mi Park, and Hae-Chang Rim. 2005. Two-phase biomedical named entity recognition using a hybrid method. In *Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP-05)*.

Zhenzhen Kou, William W. Cohen, and Robert F. Murphy. 2005. High-recall protein entity recognition using a dictionary. *Bioinformatics 2005 21*.

Micahel Krauthammer and Goran Nenadic. 2004. Term identification in the biomedical literature. *Jornal of Biomedical Informatics*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning 2001*.

Yusuke Miyao and Jun'ichi Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proceedings of Human Language Technology Conference (HLT 2002)*.

Sunita Sarawagi and William W. Cohen. 2004. Semimarkov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*.

Beth M. Sundheim. 1995. Overview of results of the muc-6 evaluation. In *Sixth Message Understanding Conference (MUC-6)*, pages 13–32.

C. Sutton and A McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *ICML workshop on Statistical Relational Learning*.

Yoshimasa Tsuruoka and Jun'ichi Tsujii. 2005. Chunk parsing revisited. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT 2005)*.

GuoDong Zhou and Jian Su. 2004. Exploring deep knowledge resources in biomedical name recognition. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*.

Table 2: Feature templates used in recognition chunk. **s** := w_s w_{s+1} ... w_e is the target chunk where w_s and w_e represent the words at the beginning and the ending of the target chunk respectively. p_i is the part of speech tag at w_i. sc_i is the shallow parse result at w_i.

Feature Name	Example of features
Whole chunk	w_s + w_{s+1} + ... + w_e
Word/POS/SC with Position	BEGIN + w_s, END + w_e, IN + w_{s+1}, ..., IN + w_{e-1}, BEGIN + p_s,...
Word/POS/SC End Bi-grams	w_{e-1} + w_e, $p_{e-1} + p_e$, sc_{e-1} + sc_e
Context Uni-gram/Bi-gram	w_{s-1}, w_{e+1}, w_{s-2} + w_{s-1}, w_{e+1} + w_{e+2}, w_{s-1} + w_{e+1}
Length, Length and End Word	$\|\mathbf{s}\|$, $\|\mathbf{s}\|$+w_e
Word shape	$WS(w_s + w_{s+1} + ... + w_e)$
Prev State /and Prev Word	$PrevState$, $PrevState$ + w_{s-1}
Count Feature	the frequency of $w_s w_{s+1}..w_e$ in a sentence is more than one
Prefix/Suffix of Chunk	2/3-gram character prefix of w_s, 2/3/4-gram character suffix of w_e

Table 3: The number of each type of named entity in the shared task data of COLING 2004 JNLPBA

	protein	DNA	RNA	cell_type	cell_line	ALL
Training Set	30269	9533	951	6718	3830	51301
Evaluation Set	5067	1056	118	1921	500	8662

Table 5: Filtering results using the naive Bayes classifier. p is the acceptance rate for filtering

data	# entity candidate	# remaining candidate	reduction ratio	recall
training ($p = 1.0^{-12}$)	4179662	505538	0.14	0.984
training ($p = 1.0^{-15}$)	4179662	725227	0.20	0.993
development ($p = 1.0^{-12}$)	418626	57960	0.14	0.985
development ($p = 1.0^{-15}$)	418626	82788	0.20	0.994

Table 6: Performance with filtering at the development data. $(< 1.0^{-12})$ indicate the acceptant rate for filtering is 1.0^{12} and $(1.0)^{15}$ as well.

	Recall	Precision	F-score	Memory Usage (MB)	Training Time (s)
Small Training Data = 2000 sentence					
Filtering ($< 1.0^{-12}$)	64.22	70.62	67.27	600	1080
Filtering ($< 1.0^{-15}$)	65.34	72.52	68.74	870	2154
Without filtering	65.77	72.80	69.10	4238	7463
All Training Data = 16713 sentence					
Filtering ($< 1.0^{-12}$)	70.05	76.06	72.93	10444	14661
Filtering ($< 1.0^{-15}$)	72.09	78.47	75.14	15257	31636
Without filtering	Not available			Not available	

Table 8: Performance of our system on the test set

	Fully Correct			Right Correct	Left Correct
Class	Recall	Precision	F-score	F-score	F-score
protein	77.74	68.92	73.07	79,97	77.94
DNA	69.03	70.16	69.59	76.47	72.46
RNA	69.49	67.21	68.33	76.67	70.83
cell_type	65.33	82.19	72.80	81.38	73.61
cell_line	57.60	53.14	55.28	65.26	58.35
overall	72.65	70.35	71.48	78.91	75.19