CHAPTER 3

BIOLOGICAL TERM RECOGNITION

The huge bio text corpus which has resulted from the dramatically increasing biological information provides a rich source of biological information upon which to develop automated knowledge extraction methods [43]. Hence, there are many research directions which have been developed to handle biological extraction in order to extract relations such as Applied Bayesian and Classical Inference: the case of the Federalist papers [44], Biobibliometrics [45], Information Hyperlinked over Proteins (iHOP) which provides the network of genes and proteins as a natural way of accessing the millions of abstracts [46] and Mining literature for protein-protein interactions [47].

Nevertheless, this is a complicated to recognize the biological names [48] because one word or phrase could refer to many entities depending on its context. Many biological named entities have various spelling forms, their compound names can be very long and different abbreviations are frequently used in the biological domain, all of which makes it difficult to recognize them precisely. Thus, Name Entity Recognition (NER) is a fundamental task which helps to specify the best boundary of biological terms from the literature. One way that NER can be approached is by considering it as a sequence labeling task [49 and 50]. This chapter will focus on biological term recognition. For bio-text mining the named entity recognition is considered as the biological labeling sequence problem which is explained in the next section also included the motivation of this work.

3.1 Introduction

Since 2001, the task of labeling sentences has been investigated. It is a useful preprocessing step for higher natural language processing tasks. In the biological domain, the biological named entity recognition can be thought of as a sequence segmentation problem [51, 10], where each word is a token in a sequence to be assigned a label such as protein, DNA, RNA, cell-line, cell-type [18, 52], see **Figure 3.1** for example

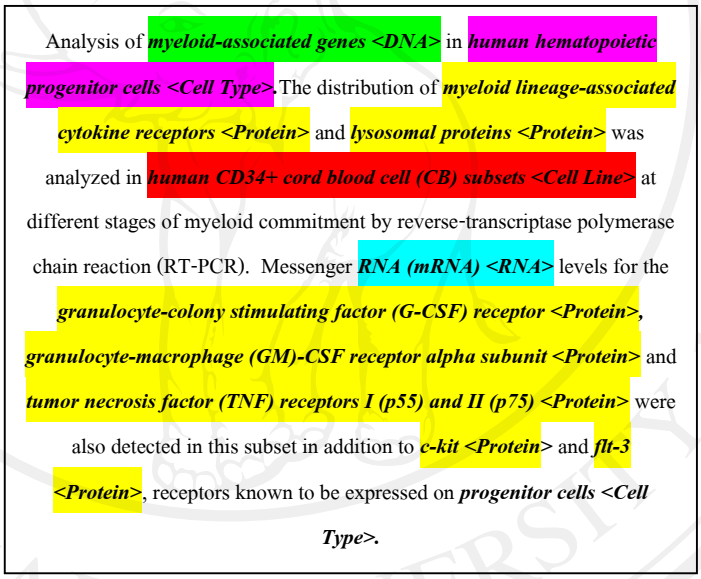


Figure 3.1 Sentence with <DNA>, <RNA>, <Protein>, <Cell Line>, and <Cell Type> tags generated by Biological Named Entities Recognition

NER in biological has been studied for approximately ten years [53]. Biological named entities recognition remains a challenging task and active area of research for many reasons including:

- There is no a complete dictionary for most types of biological named entities [54].

- The same word or phrase can refer to different entities depending upon context.
- Many biological named entities have various spelling forms.
- Sometimes biological named entities are very long. These factors highlight the difficulties for identifying the boundary of named entities.
- Named entities may be cascaded. One named entities may be embedded in another named entities. More efforts must be made to identify this kind of named entities.
- Abbreviations are frequently used in biological domain. Since abbreviations often do not have enough evidence for certain named entities class, it is difficult to classify them correctly.

To address the issues presented above, many machine learning models have been developed that can deal with NER [55]. The benefit of these kinds of models is that they do not require a dictionary in memory and they can automatically form rule sets by learning from a large text corpus. NER tools are a prerequisite for many applications working on text, such as information retrieval, information extraction or document classification [52] and the state of the art is around F-score of 77.57%. Several techniques have been used for the NER task which use the three most important and effective approaches [1]. Rule-based methods are quite fast and there is no need to store a dictionary in memory but these methods can be difficult to find a perfect grammar. Dictionary-based methods yield relatively high accuracy and still solve unknown word problems by updating the dictionary, nevertheless the general disadvantage of this approach lies in the necessary dictionary update and curation when new protein or gene

names are invented [56]. Machine learning is the alternative way so that there is no need for a dictionary in memory and it can automatically form a rule set by learning from a large text corpus, the disadvantage is that it requires a large corpus and learning time plus a good learning algorithm [54]. Machine Learning based approaches are divided into two main categories: Classifier-based (Decision trees, naïve Bayes and Support Vector Machine:SVMs) [57] and Markov model based (Hidden Markov Models:HMM, Maximum Entropy Markov Models:MEMM and Conditonal Random Fields : CRFs) [58, 59]. This chapter will explain the biological named entity recognition model with graphical models.

3.2 Biological Named Entity Recognition with Graphical Models

This section is about the review of three of the most important and effective named entity recognition models that are often applied to biological terms recognition problems: Hidden Markov Model (HMM), Maximum Entropy Markov Model (MEMM), Support Vector Machine (SVM), and Conditional Random Fields (CRFs)**.**

3.2.1 Hidden Markov Model (HMM)

Hidden Markov model (HMM) is a powerful tool for representing sequential data [58]. It has been successfully applied to: Part-of-speech tagging:

<PRP>He</PRP><VB>books</VB><NNS>tickets</NNS>,

Named entity recognition: <ORG>Mips</ORG> Vice resident

<PRS>JohnHime</PRS>

Information extraction: <TIME>Afterlunch</TIME> meet<LOC>under the oak tree</LOC>

The purpose of HMM is to find the most likely tag sequence $Y^n = y_1 y_2 \dots y_n$ for a given token sequence $X^n = x_1 x_2 \dots x_n$ that maximizes$P(Y_n|X_{n,})$. In a token sequence X^n, the token x_i is defined as $x_t = (f_t, w_t)$,where w_i is the word and f_i is the feature set related to the word.

In tag sequence, Y_n, each tag consists of the parts:

Boundary category, which denotes the position of the current word in NE,

Entity category, which indicates the NE class. This estimates the following joint probability of the current token x_t and label y_t conditioned on the previous label y_{t-1} and previous two tokens x_{t-1} and x_{t-2}: $P(x_t, y_t|y_{t-1,}x_{t-1,}x_{t-1,})$

Same as in Biological Named Entity Recognition, given a sequence of tokens (observations): … "p53 protein suppresses mdm2 expression"…and a trained HMM:

Figure 3.2 State transition of Hidden Markov Model (HMM)

Find the most likely state sequence: (Viterbi) $argmaxY_n P(y_n|x_{n,})$

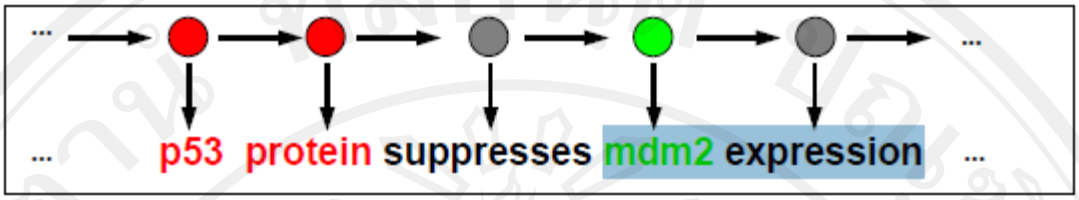


Figure 3.3 Example of Named Entity Recognition of HMM

Any words, which are said to be generated by the designated "protein name" state, are extracted as a protein name: p53 protein.

3.2.2 Maximum Entropy Markov Model (MEMM)

A limitation of HMM is that it is hard to extend them to allow multiple features of observations, rather than atomic observations themselves. An alternative to the HMM was proposed in which the transition and observation probability matrices are replaced by maximum entropy classifiers for each state [59]. Many classification tasks are most naturally handled by representing the instance to be classified as a vector of features. The state and observation transition functions into a single maximum entropy model for each state are combined. The condition the tag sequence was assigned to a sentence on such things as part–of–speech tags, phrasal tags, and predicate verbs. The maximum entropy distribution is a conditional exponential model of the form:

$$P_{s'}(s|o) = \frac{1}{Z(o,s')} \exp\left(\sum_i \lambda_i f_i(o,s)\right) \quad (3.1)$$

where λ_i are the feature weights that need to be estimated from the training data, and $Z(o,s')$ is a normalization factor to ensure P is a probability distribution.

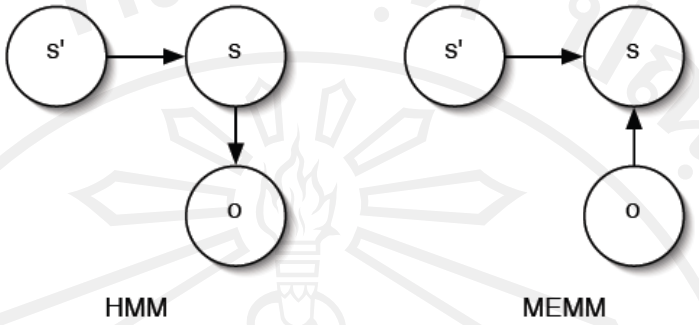


Figure 3.4 MEMM states are conditioned on the previous state and the observation.

3.2.3 Conditional Random Field Model (CRFs)

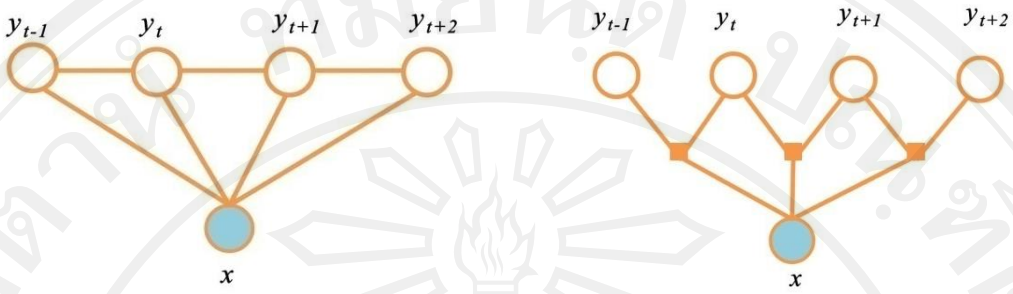
The general concepts of the conditional random fields method are described below:

1) Principles of Conditional Random Fields (CRFs)

First introduced by Lafferty et al. (2001) [60], the conditional random fields (CRFs) method is a probabilistic model for computing the probability $p(Y|X)$ of an event occurring for the conditional probability distribution over the possible output or label $y = (y_1, y_2, \dots, y_n) \in Y$ given observations $x = (x_1, x_2, \dots, x_n) \in X$. For regulatory element prediction, a specific case of the CRF called a linear-chain CRF is used.

2) Linear-chain CRFs

Linear-chain CRFs are a form of CRFs whose structure is like a linear chain model describes the output variables as a sequence. The features of a linear-chain CRFs (as represented in **Figure 3.5 (b))** are evaluated over all of X and consecutive hidden variables (y_{t-1}, y_t).



(a) (b)

Figure 3.5 Graphical structure of the linear chain CRFs for sequence labeling [61]

Graphically, (a) nodes represent the individual y_t, as well as all of X. The graph's edges represent dependencies among the observations and hidden variables where features are defined over fully-connected subgraphs (b) or cliques

Because the CRF directly models the conditional probability of a hidden sequence Y given observations X, $P(Y|X)$, it is referred to as a discriminative model [61]. In a CRF, this conditional probability can be formulated as

$$P(Y|X) = \frac{1}{Z_\theta(X)} \exp(\theta \cdot F(Y, X)) \tag{3.2}$$

$Z_\theta(X)$ is the normalization factor from Lafferty et al. (2001) [92],

$$Z_\theta(X) = \sum_y \exp\left(\sum_j \theta_j F_j(Y, X)\right) \tag{3.3}$$

For a linear-chain CRF, the j^{th} feature sum F_j is defined to be the feature function f_j and g_j evaluated over the entire sequence:

$$F_j(Y, X) = \sum_j \lambda_j f_j(y_{i-1}, y_i, \boldsymbol{x}, i) + \sum_j \mu_j g_j(y_i, \boldsymbol{x}, i) \tag{3.4}$$

Where $f_j(y_{i-1}, y_i, \boldsymbol{x}, i)$ is a transition feature function for the space of all possible observation sequences and the labels at positions i and $i-1$ in the label sequence, $g_j(y_i, \mathbf{x}, i)$ is a state feature function of label sequence at position i and the observation sequence. The index j in f_j and g_j is a feature serial number to represent different features. The parameter $\theta = \{\lambda, \mu\}$ with λ_j and μ_j correspond to features f_j and g_j, respectively, and they are trained to maximize the conditional likelihood of the training data.

3) CRF training

The parameters θ of a linear-chain CRF are trained by using a gradient descent algorithm. Gradient descent algorithms use an objective function F_θ and its gradient with respect to θ to iteratively increase the value of the objective function. L-BFGS, a limited-memory variant of the traditional Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, is one gradient descent algorithm [62]. The BFGS method computes an approximate Hessian matrix at each step, using these second derivatives to find a new set of feature weights which are used to compute the objective value and its gradient. The L-BFGS algorithm was chosen because it has been shown to perform well on CRFs [63].

Assuming the training data $\{(x^{(k)}, y^{(k)})\}$ is independent and identically distributed for the product of (3.8) over all training sequences, as a function of the parameters θ, this is known as the likelihood, denoted by $p(\{y^{(k)}\}|\{x^{(k)}\}, \theta)$. Maximum likelihood training chooses parameter values such that the logarithm of the likelihood, known as the log-likelihood, is maximized. For a CRF, the log-likelihood is given by

$$\mathcal{L}(\theta) = \sum_k [log \frac{1}{Z(\mathbf{x})} + \sum_j \theta_j F_j(x^{(k)}, y^{(k)})] \quad (3.5)$$

This concave function guarantees convergence to the global maximum. Differentiating the log-likelihood with respect to parameter θ_j gives

$$\frac{\partial \mathcal{L}(\theta)}{\theta_j} = E_{\tilde{p}(\mathbf{X},\mathbf{Y})}[F_j(\mathbf{y}, \mathbf{x})] - \sum_{\mathbf{k}} E_{p(Y|x^{(k)},\theta)}[F_j(\mathbf{y}, x^{(k)})] \quad (3.6)$$

where $\tilde{p}(\mathbf{x}, \mathbf{y})$ is the empirical distribution of training data and $E_p[\cdot]$ denotes the expectation with respect to distribution p. The setting this derivative to zero yields the maximum entropy model constraint which is the expectation of each feature with respect to the model distribution which is equal to the expected value under the empirical distribution of the training data.

It is not always possible to analytically determine the parameter values that maximize the log-likelihood by setting the gradient to zero and solving for λ since this does not always yield a closed form solution. Instead, the maximum likelihood parameters must be identified using an iterative technique such as iterative scaling [65, 65 and 66] or gradient-based methods [68]. Previously, Wallach H. (2002) [63] also proposed that the CRFs parameters can be estimated by two algorithms, based on Improved Iterative Scaling (IIS) and Generalized Iterative Scaling (GIS).

4) CRF inference

In linear-chain CRFs, a gradient-based training requires computing marginal distributions $P(y|x)$, and testing requires computing the most likely assignment $y^* = Argmax_y P(y|x)$. Then, the Viterbi algorithm is used infer a straight forward variant of that used for HMM to label an unseen instance. The inference task can be

performed efficiently and exactly by variants of the standard dynamic-programming algorithms for HMM. The review of HMM algorithms is described in more detail by L.R. Rabiner and B.H. Juang (1989) [58].

For each state y at position t. The most likely previous state is found by taking the evaluation of the best partial path to each of the possible previous states y_{t-1} with the evaluation of the edge (y_{t-1}, y_t). By combining results from the forward and backward recursions as described by A. Culotta and A. McCallum (2004) [67], to compute the global optimum value $V_t(y)$ of the best partial path to y at position t is thus defined by this which yields the Viterbi recursion:

$$V_t(y) = \begin{cases} \max_{y'} V_{t-1}(y') + \theta \cdot F(y, y', X, t), & if\ t > 0 \\ 0, & if\ t = 0 \\ -\infty, & if\ t < 0 \end{cases} \quad (3.7)$$

A back pointer from the segment is stored to this most likely previous state. When the algorithm reaches the end of the sequence, it then chooses the ending segment with the highest partial path probability (where the partial path is now the complete path), and follows its back pointers to recover the Viterbi path.

3.2.4 The comparison of HMM, MEMM and CRFs

From above, HMM is the generative model which is a model for randomly generating observable data, typically given some hidden parameters. It specifies a joint probability distribution over observation and label sequences. Generative models are used in machine learning for either modeling data directly while MEMM and CRFs are generative models which opposed to generative models, do not allow one to generate samples from the joint distribution of x and y. However, for tasks such

as classification and regression that do not require the joint distribution, discriminative models can yield superior performance. On the other hand, generative models are typically more flexible than discriminative models in expressing dependencies in complex learning tasks. In addition, most discriminative models are inherently supervised and cannot easily be extended to unsupervised learning. Application specific details ultimately dictate the suitability of selecting a discriminative versus generative model. HMM and MEMM are based on local normalization while CRFs is based on global normalization. The advantage of CRFs is their great flexibility to integrate a wide variety of arbitrary, non-independent features in the input [53] which results in the relaxation of the independence assumptions required by Hidden Markov Model (HMM) in order to ensure a tractable inference, CRFs still avoid the label bias problem that is the weakness of Maximum entropy Markov Model (MEMM) [66, 68]. CRFs have been shown to outperform both MEMM and HMM on a number of real-world sequence labeling tasks [55, 69]. These reasons refers that CRFs is correspond with the labeling sequence problem solution to generate the named entity recognition model.

There are the works about comparing HMM, MEMM and CRFs for disfluency detection. Y. Liu and et al. mention that find that the conditional modeling approaches (MEMM and CRFs) provide an elegant and successful way to model various and potentially correlated features. It thus avoids the use of ad-hoc rules used in the HMM and MEMM methods. In terms of effects of transcripts and corpora, found that that performance degrades substantially due to word errors in recognition output. They still mention that much work still remains to improve system performance for edit detection

[Y Liu]. Hence, to recognition biological term, this chapter will focus on CRFs to improve the named entity recognition model.

3.3 Biological Information Retrieval using Latent Semantic Indexing and Biological Named Entity Recognition

3.3.1 Introduction

The objective of this thesis focus on information extraction but in bio-text mining, the information retrieval is the techniques which necessary for information extraction to retrieve the relevance documents. Hence, this work will consider the information retrieval based on LSI and named entity recognition. Since the last chapter, the general terms couldn't help much for information retrieval. To study the effective of named entity recognition as feature which generate by CRFs can improve the information retrieval. The methods and dataset are described in the next section.

3.3.2 Methods and Dataset

This work is arranged more from the chapter 2. The CRFs part is applied to the experiment. The framework is designed as **Figure 3.6**

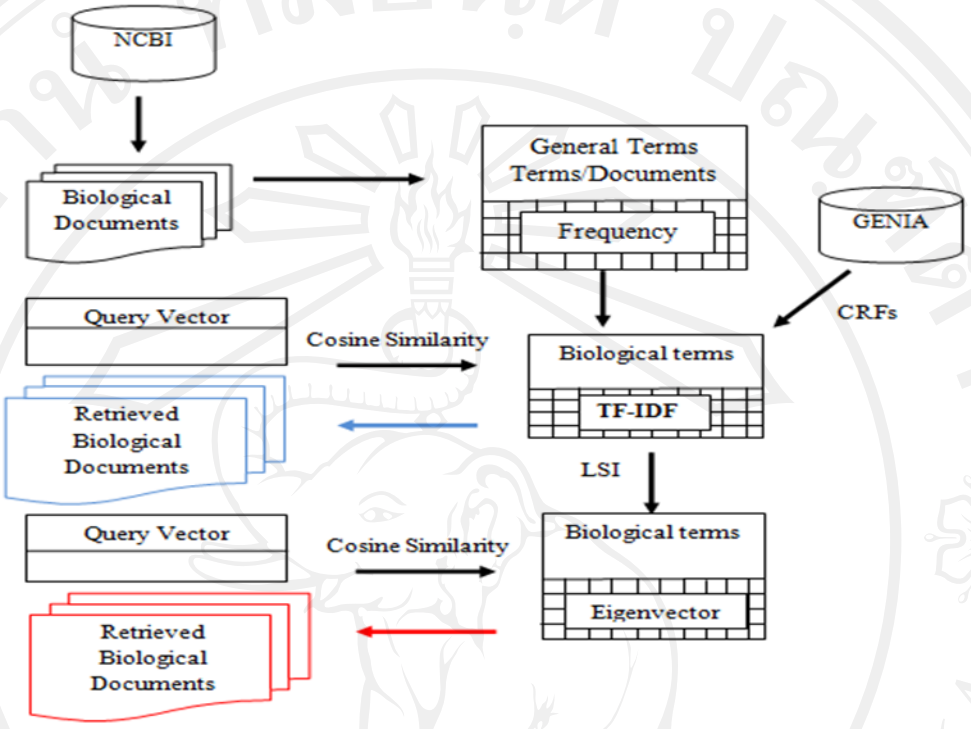


Figure 3.6 Framework of Biological Information Retrieval using Latent Semantic Indexing and Biological Named Entity Recognition

From **Figure 3.6**, The Framework is divided into two parts. First, the biological documents from NCBI corpus are labeled with CRFs model which generated by GENIA corpus. Then, the biological terms which labeled are scaled by TF-IDF. Another part, eigenvector is generated by LSI after scaled by TF-IDF. LSI is applied with CRFs. In the information retrieval, the models for both are tested by Cosine Similarity with query vector. The retrieved biological documents are returned. The methods and dataset in this work as follows,

1) Conditional Random Fields

One of the methods for performing such labeling and segmentation tasks is Conditional Random Fields (CRFs). CRFs is a probabilistic framework for labeling and segmenting sequential data based on the conditional approach [60]. In Biological Fields, the use of CRFs for Biological named entity recognition (NER) is prevalent [70, 10]. This method makes extensive use of a diverse set of features, including local features, full text features and external resource features [53]. The CRFs is an undirected graphical model [68], which is suited for sequence analysis.

Normally conditioned on x, the random variable represents observation sequences. Let $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ be an undirected graph such that there is a node $v \epsilon V$ corresponding to each of the random variables representing an element Y_v of Y. If each random variable Y_v obeys the Markov property with respect to G, then (X, Y) is a Conditional Random Field. In theory, the structure of graph G may be arbitrary, it represents the conditional independencies in the label sequence being modeled. However, when modeling sequences, the simplest and most common graph structure is a simple first-order chain, as illustrated in **Figure 3.7**

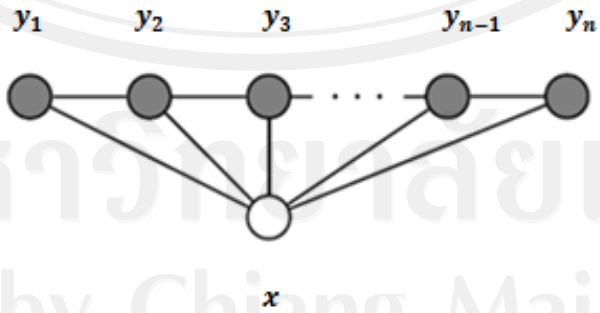


Figure 3.7 Graphical structures of chain-structured CRFs for sequences.

The variables corresponding to un-shaded nodes are not generated by the model.

Let I define the probability of a particular label sequence y given observation sequence x to be a normalized product of potential functions, each of the form

$$\exp(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i) \quad (3.8)$$

where $t_j(y_{i-1}, y_i, x, i)$ is a transition feature function of the observation sequence and the labels at positions i and $i-1$ in the label sequence; $s_k(y_i, x, i)$ is a state feature function of the label at position i and the observation sequence, λ_j and μ_k are parameters to be estimated from the training data. When defining feature functions, I construct a set of real-valued features $b(x, i)$ of the observation to expresses some characteristic of the empirical distribution of the training data that should also hold for the model distribution. An example of such a feature is

$$b(x,i) = \begin{cases} 1, & if\ the\ observation\ at\ position \\ & \quad is\ the\ word\ "Gene" \\ 0, & otherwise \end{cases} \quad (3.9)$$

Each feature function takes on the value of one of these real-valued observation features $b(x, i)$ if the current state (in the case of a state function) or previous and current states (in the case of a transition function) take on particular values. Therefore, all feature functions are real-valued. For example, consider the following transition function:

$$t_j(y_{i-1}, y_i, x, i) = \begin{cases} b(x,i), & if\ y_{i-1} = Protein \\ & \quad and\ y_i = Protein \\ 0, & otherwise \end{cases} \quad (3.10)$$

Let

$$s_k(y_i, x, i) = s_k(y_{i-1}, y_i, x, i) \quad (311)$$

and

$$F_j(y,x) = \sum_{i=1}^{n} f_i\,(y_{i-1}, y_i x, i) \quad (3.12)$$

where each $\sum_{i=1}^{n} f_i\,(y_{i-1}, y_i x, i)$ is either a state function $s_k(y_{i-1}, y_i, x, i)$ or transition function $t_j(y_{i-1}, y_i, x, i)$.

This allows the probability of a label sequence y given an observation sequence x to be written as:

$$p(y|x,\lambda) = \frac{1}{Z(x)} \exp(\sum_j \lambda_j F_j(y,x)) \quad (3.13)$$

A normalization factor is $Z(x)$.

2) Dataset

The dataset is gathered from the NCBI corpus and GENIA corpus. The biological documents or abstracts are used in testing process while biological terms are recognized by the CRFs model to label named entities types (Protein, DNA, RNA, Cell line and Cell type) which generated from GENIA corpus.

3.3.3 Results and Discussion

The performance is evaluated by precision and recall values. Then, the harmonic mean of precision and recall are weighted with the traditional F-measure. The performance of models comparison are shown in **Table 3.1**

Table 3.1 Performance of Models for Information Retrieval

	Recall (%)	**Precision (%)**	**F-Measure (%)**
CRF	70.00	72.50	71.23
LSI	65.00	67.00	65.98
CRF+LSI	75.40	76.50	75.95

According to experimental results, the performance of CRFs was about 71% (70% recall and 72.5% precision values). The CRFs method is suitable for finding important keywords in documents (decrease features), while the performance of LSI **(See Chapter 2**) was about 65% (67% recall and 67% precision values).The result showed that LSI methods help to find the latent concepts of the whole documents, not only important keywords. To improve the biological documents retrieval performance by CRFs and LSI, feature selection was generated by the CRFs model to find the names of biological entities and the relevance documents could be retrieved by LSI.

In this work, a method was proposed to improve biological information retrieval by using the CRFs model for finding some specific features and using LSI for retrieving the relevant documents. The performance of our proposed model depends on the given training data set. In the future, the method could possibly be improved for more effective feature selection to enhance the performance of the model.

3.4 Biological Named Entity Recognition using the Conditional Random Fields Technique

3.4.1 Introduction

The CRFs is another machine learning model which has been used to analyze biological text. Some such research topics have included forming NER and rich feature sets and identifying gene and protein mentions in text [9]. Biological and chemical named entity recognition with CRFs offers the advantage of dictionary feature which can integrate linguistic knowledge to identify biological named entities [53], extract semantic

biological relations from text [69], identify the results in biological abstracts [3], the name alias phenomenon and an open dictionary from the database term list SwissProt and the alias list LocusLink, the abbreviation resolution and in domain POS [48], system which makes extensive use of local and syntactic features within the text [72] and proposed a classifiers ensemble method [55]. These researches shows state of the art around 78%.

However, the problem of NER using CRFs remains a challenging task and is still an open and active area of research because of the system still difficult to recognize long, complicated NEs and to distinguish between two highly overlapped NE classes, such as cell-line and cell-type [1 and 3], suggested that should focus on improving the accuracy of detection of entities as well as entity boundaries, which will also greatly improve the relation extraction performance. This target work will focus on improving the CRFs performance by focusing on the potential function to generate feature for the model prediction.

3.4.2 Methods and Dataset

This method which uses in this work is CRFs. We will study the performance of CRFs via improving the potential function. Finally the models which generated from different potential function will be compared. This work is designed the framework as **Figure 3.8**

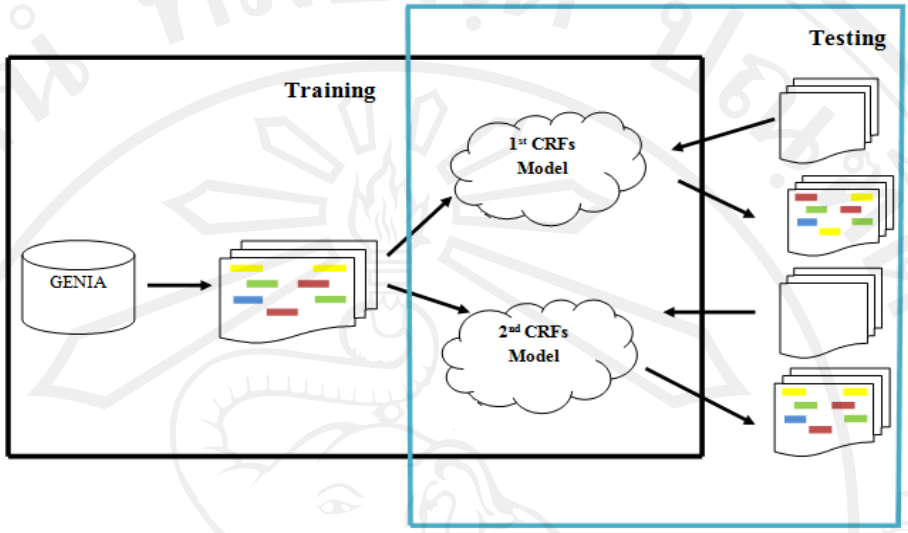


Figure 3.8 The framework of Biological Named Entity Recognition using the Conditional Random Fields Technique

1) Conditional Random Fields for Biological Named Entity Recognition

The CRFs method is an undirected graphical model [51] which is appropriate for sequence analysis [60] and can be used for Biological Named Entity Recognition (NER) [9]. The advantage of CRFs are in their great flexibility to integrate a wide variety of arbitrary, non-independent features from an input dataset and relax the independence assumptions required by Hidden Markov Model (HMM) in order to ensure that tractable inferences can be made [66]. Also, CRFs have been shown to prevent the label bias problem and can outperform HMM on a number of real-world sequence labeling tasks [68 and 69].

Let $\boldsymbol{x}$ be a random variable which represents a sequence of words, $\boldsymbol{y}$ be the labeling random variable associated with $\boldsymbol{x}$, $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a linear chain undirected graph

representing the conditional independencies of $\boldsymbol{y}$ given $\boldsymbol{x}$. For example, given a sequence of words $\boldsymbol{x}$ as in **Figure 3.9**

$\boldsymbol{x}$ =	Presence	of	beta	2-M	was	analyzed	by	immunohistochemistry	.
	|	|	|	|	|	|	|	|	|
$\boldsymbol{y}$ =	O	O	B-protein	I-protein	O	O	O	O	O

Figure 3.9 Example of Biological Named Entity Recognition

$\boldsymbol{y}$ is the corresponding label sequence predicted by the CRFs. In this example, the words "beta 2-M" at positions x_3 and x_4 are predicted as protein name; i.e. y_3 = "B-protein" and y_4 = "I-protein". "B-protein" indicates the beginning of the protein name, and "I-protein" indicates the consecutive compound protein name. The other positions x_t which have the label y_t = "O" indicate that the other x_t are plain words.

Conceptually, the conditional probability of a particular label sequence $\boldsymbol{y} = y_1, y_2, y_3, \ldots, y_T$ given observation sequence $\boldsymbol{x} = x_1, x_2, x_3, \ldots, x_T$ is defined by a normalized product of potential functions.

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{z(x)} \exp\left(\sum_t F(\boldsymbol{y}, \boldsymbol{x}, t)\right) \tag{3.14}$$

where $\boldsymbol{Z}(\boldsymbol{x})$ is a normalization factor, and $\boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x}, t)$is the potential function of $\boldsymbol{x}$ and $\boldsymbol{y}$ at position t.

The CRFs based NER model is trained with an available known $\boldsymbol{x}$ and $\boldsymbol{y}$ dataset, from which the conditional probability distribution($\boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})$) is inferred. In the prediction step, the label sequence $\boldsymbol{y}$ obtains the highest conditional probability. The NER model then

selects the maximum probability path $\boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})$ with a dynamic programming Viterbi-like algorithm [60], described as:

$$y^* = arg\ max_y p(\boldsymbol{y}|\boldsymbol{x}) \tag{3.15}$$

$$= arg\ max_y \exp\left(\sum_{t=1}^{T} F(\boldsymbol{y}, \boldsymbol{x}, t)\right)$$

- **Potential Functions of CRFs for Named Entity Recognition**

In this section, word surfaces and labels were used to be a feature selection and focus to define the potential function $F(\boldsymbol{y}, \boldsymbol{x}, t)$ of the CRFs for named entity recognition. The potential function for the 1st CRFs is described first, and then one for the second-order CRFs is extended from that formulation.

Let $\boldsymbol{x} = x_1, x_2, x_3, \ldots, x_T$ be a sentence (sequence of words), $\boldsymbol{y} = y_1, y_2, y_3, \ldots, y_T$ be the corresponding label sequence of $\boldsymbol{x}$, associated with class of word $y_t \in \{l_1, l_2, \ldots, l_M\}$, which could be "B-protein", "I-protein", "B-cell_line", "I-cell_line", "B-cell_type", "I-cell_type", "B-DNA", "I-DNA", "B-RNA", "I_RNA" and "O".

- **Potentials in 1st CRFs**

The potential function $F(\boldsymbol{y}, \boldsymbol{x}, t)$ for 1st CRFs is the combination of transition potential functions $f_{l',l}^{i}(y_{t-1}, y_t, \boldsymbol{x})$ and state potential functions $g_l^i(y_t, \boldsymbol{x})$:

$$F\ (\boldsymbol{y}, \boldsymbol{x}, t) = \sum_{l',l,i} \lambda_{l',l}^{i} f_{l',l}^{i}(y_{t-1}, y_t, \boldsymbol{x}) + \sum_{l,i} \mu_l^i\, g_l^i(y_t, \boldsymbol{x}) \tag{3.16}$$

for which $\lambda_{l',l}^{i}$ and μ_l^i represent the transition function and state function parameters and i is index of the parameters. $z(\boldsymbol{x}, t)$ is defined as;

$$z(\boldsymbol{x}, t) = \begin{cases} 1, \ if \ \boldsymbol{x} \text{ at position } t \pm 2 \text{ is equal to } words \\ 0, \ otherwise \end{cases} \quad (3.17)$$

From equation (3.23), $z(\boldsymbol{x}, t)$ represents whether $\boldsymbol{x}$ at positions *t*-2 to *t*+2 consist of word surfaces or biological terms. For example, in **Figure 3.**10, if t is 3, the words, "Presence of beta 2-M was" are considered. I consider a sliding window (size=5) which surround the position of interest *t*.

The transition potential function $f^i_{l',l}(y_{t-1}, y_t, \boldsymbol{x})$ characterizes whether at some specific position t; the y_{t-1},y_t are particularly labelled by l' and l respectively:

$$f^i_{l',l}(y_{t-1}, y_t, \boldsymbol{x}) = \begin{cases} z(\boldsymbol{x}, t) & if \ y_{t-1} = l', y_t = l \\ 0 & otherwise \end{cases} \quad (3.18)$$

where l' and l , associated with some specific *i*, could be "B-protein", "I-protein", "B-cell_line", "I-cell_line", "B-cell_type", "I-cell_type", "B-DNA", "I-DNA", "B-RNA", "I_RNA" or "O".

The state potential function $g^i_l(y_t, \boldsymbol{x})$ characterizes whether at some specific position t, y_t is particularly labelled by l.

$$g^i_l(y_t, x) = \begin{cases} z(\boldsymbol{x}, t) & if \ y_t = l \\ 0 & otherwise \end{cases} \quad (3.19)$$

- **Potentials in 2nd CRFs**

$F(\boldsymbol{y}, \boldsymbol{x}, t)$ is defined in equation (3.20) for 2nd CRFs as:

$$F(\boldsymbol{y},\boldsymbol{x},t)=\sum_{l',l,i}\lambda^{i}_{l',l}\,f^{i}_{l',l}(y_{t-1},y_t,\boldsymbol{x}) + \sum_{l,i}\mu^{i}_{l}\,g^{i}_{l}(y_t,\boldsymbol{x}) + \sum_{l'',l',l,j}\lambda^{j}_{l'',l',l}f^{j}_{l'',l',l}(y_{t-2},y_{t-1},y_t,x)+\sum_{l',l,j}\mu^{j}_{l',l}g^{j}_{l',l}(y_{t-1},y_t,x) \tag{3.20}$$

Where $f^{i}_{l',l}(y_{t-1},y_t,\boldsymbol{x})$ and $g^{i}_{l}(y_t,\boldsymbol{x})$ are defined by (3.18) and (3.19). Let *j* be the index of the parameters in the 2nd CRFs.

$$f^{j}_{l'',l',l}(y_{t-2},y_{t-1},y_t,\boldsymbol{x})=\begin{cases} z(x,t) & if\ (y_{t-2}=l'',y_{t-1}=l') \\ & and\ (y_{t-1}=l',y_t=l\,) \\ 0 & otherwise \end{cases} \tag{3.21}$$

$$g^{j}_{l',l}(y_{t-1},y_t,x)=\begin{cases} z(x,t) & if\ y_{t-1}=l',y_t=l \\ 0 & otherwise \end{cases} \tag{3.22}$$

For example, in the sequence of words in **Figure 3.10**, when $t=4$, x_4 = "2-M"; when *l*= "I-protein", l'= "B-protein" and l''= "O", $f^{i}_{l',l}(y_{t-1},y_t,\boldsymbol{x})$ in equation (3.18) is equal to 1 if the value of y_3 and y_4 are "B-protein" "I-protein". While $f^{j}_{l'',l',l}(y_{t-2},y_{t-1},y_t,\boldsymbol{x})$ in equation (3.21) is equal to 1 if the value of y_2, y_3 and y_4 are "O", "B-protein", and "I-protein" respectively**.**

2) Time complexity of CRFs

The inference for CRFs based on the Viterbi algorithm is quite efficient. The training process is much more expensive due to the heavy forward-backward computation to evaluate the log-likelihood function and its gradient vector for each iterative scaling step.

The time complexity of the training process is O(mNTQ2nS), in which m is the number of training iterations; N is the number of training data sequences; T is the average length of training sequences; Q is the number of class labels; n is the number of CRF features; and S is the searching time of L-BFGS optimization at each step. In practical implementation, the computational time should be larger due to many other operations such as numerical scaling (to avoid numerical problems), smoothing, and mapping between data formats. The time complexity of the 2nd CRFs is even much larger, O(mNTQ4nS), because the number of labels is now squared. When the number of labels is large, training CRFs on single computer is very time consuming. Consider the performance of the 2nd CRFs is efficiency for named entity recognition. These the reason while not to be the 3rd CRFs. It is not worth because the time consuming will be O(mNTQ8nS). The improvement time complexity is not the objective in this thesis.

3) Dataset

Dataset was from GENIA corpus or JNLPBA2004 [69] which composed of 2,000 PubMed abstracts with term annotation for training data. Evaluation data compose with 404 PubMed abstracts, with one file with term annotation and one without for each and evaluation tool updated evaluation tool. Use this tool to get the evaluation equivalent to that of the shared task.

3.4.3 Results and Discussion

The CRFs model was tested on the JNLPBA2004 shared task dataset, The corpus is composed of five classes: protein, DNA, cell_type, cell_line and RNA. The pattern labels of each class begin with “B-” and are followed with the class such as B-protein, B-

DNA. If the biological terms are composed of two or more words, the pattern is "I-" for example, protein name (Human T3 factors) the class labels are in this format.

Human B-protein

T3 I-protein

factors I-protein

The FlexCRFs package [83] was used to train and label the text sequences for the first- and 2nd CRF analysis.

The performance of the NER model was evaluated using standard precision, recall and F-values. Precision π is defined as the fraction of predictions, made by the model, that are correctly matched. Recall ρ is defined as the fraction of the manual labels that were correctly matched.

In the evaluation of the model, only the exactly matched results were considered for a chunk-based performance evaluation. The performances of the biological named recognition models were shown in **Tables 3.2 and 3.3**

Table 3.2 Performance of biological named entity recognition based on 1st CRFs

Label	Recall (%)	Precision (%)	F-Measure (%)
Protein	56.01	68.16	61.49
DNA	57.58	70.45	63.37
Cell Type	58.15	77.95	66.61
Cell Line	46.00	53.86	49.62
RNA	61.86	64.04	62.93
Average	**56.18**	**69.50**	**62.13**

The 1st CRFs model returned F-measure values with an average of 62.13%. The precision scores were higher than the recall scores by 13.32%. This shows that the model returns fewer false positives than false negatives. The F-measure value of Cell Type had the highest percentage, 66.61%, while Cell Line had the lowest at 49.62%. The characteristics of Cell Line and Cell Type were quite similar, although the size of the training and testing datasets for Cell Type were about two times larger than in than the Cell Line. Thus, some Cell Line terms were wrongly predicted to be Cell Type. Although the 1st CRFs model explained above performs well for biological term recognition of not so long ranges interactions, it fails to encode long-range interactions among states due to the limitation of the first order dependency (the current state depends only on one previous state). Therefore, the model was extended to consider the more general case of 2nd CRFs which incorporate the recognition of longer ranged correlations than in 1st CRFs. These results are shown in **Table 3.3**.

Table 3.3 Performance of biological named entity recognition based on 2nd CRFs

Label	Recall (%)	Precision (%)	F-Measure (%)
Protein	100.00	100.00	100.00
DNA	99.24	99.62	99.43
Cell Type	99.79	99.95	99.87
Cell Line	98.80	97.99	98.39
RNA	100.00	100.00	100.00
Average	**99.73**	**99.88**	**99.81**

From **Table 3.3**, the precision scores were still generally higher than the recall scores as with the 1st CRFs but with a much smaller gap (only about 0.15%). From the potential

function of 2nd CRFs, more potential functions could generate a more complex model than for those of the 1st CRFs for capturing longer ranged interactions.

For the recall value, the average of the recall improvement value was around 43.55%. This showed that a higher fraction of the manual labels were correctly matched.

In general, biological terms which represent Cell Line and Cell Type are similar type words. The model from the first-order CRFs at times returned wrong predictions for the Cell Type class. The potential function for the 2nd CRFs supported interactions which were long enough to discriminate between them. Generally, the 2nd CRFs were very efficient as shown in **Table 3.3** above, but the time complexity was the drawback. The time complexity of the 2nd CRFs was much larger than in the first order especially when the number of labels was large, it was very time consuming, but runtime improvement considerations were not within the scope of this work.

According to the potential function above, the results showed that 1st CRFs predict with very high performance 1-2 words but the 2nd CRFs could predict the co-occurrence of words which were suitable for biological named entity recognition and longer sets of words. An example which showed the differences between the 1st and 2nd CRFs models was as follows:

Biological Term	**Label**	**1st CRFs**	**2nd CRFs**
human	B-protein	O	B-protein
T3	I-protein	B-protein	I-protein
receptor	I-protein	I-protein	I-protein

The 1st CRFs predicted the word "human" to be a normal word and not a biological term while the words "T3 receptor" were correctly predicted. In contrast, the 2nd CRFs was able to group these words together correctly to form a single biological term. This showed that 1st CRFs could only predict single words or word pairs to form biological terms. Thus, it was possible to miss some biological terms which were composed of many words. The 2nd CRFs returned correct matches because of the potential function which focuses on the co-occurrence of words, and they were quite effective to predict the long-range relationships between words (biological name entity). That's the reason why 2nd CRFs predicted with such a high performance increase over the 1st CRFs.

The models were also compared with other existing methods tested on the same JNLPBA 2004 task set as is shown in **Table 3.4**.

Table 3.4 Comparison of biological named entity recognition performance with others existing models

Label	Recall (%)	Precision (%)	F-Measure (%)
1st CRFs	56.18	69.50	62.13
2nd CRFs	**99.73**	**99.88**	**99.81**
Deep knowledge resources[48]	75.99	69.42	72.55
Local and syntactic features [72]	68.60	71.60	70.10
CRF-Simple orthographic features[9]	70.00	69.0	69.50
Class-attribute stacking [55]	75.57	79.68	77.57

The performance of the 1st CRFs model was lowest. It was not powerful enough for accurate prediction when compared with other models. Z. GuoDong and S. Jian [48] have explored various deep knowledge resources such as the name alias phenomenon, they

used both a closed dictionary from the training corpus and an open dictionary from the database term list SwissProt and the alias list LocusLink, the abbreviation resolution and in domain POS. For these methods, their performance was 72.55%. Also, B. Settles [9] presented in detail a framework for recognizing multiple entity classes in biological abstracts with Conditional Random Fields. It showed that a CRFs-based model with only simple orthographic features could achieve reasonable performance values while only using semantic lexicons which did not affect performance.

The performance of this model was around 69.50%. J. Finkel, S. Dingare,H. Nguyen,M. Nissim,C. Manning, and G. Sinclair [72] have also proposed a system which makes extensive use of local and syntactic features within the text, as well as external resources including the web and gazetteers. It achieved an F-score of 70.10%.

In 2008, H.Wang, T. Zhao,H.Tan and S. Zhang [55] proposed a classifiers ensemble method which achieved an F-score of 77.57%. The performance analysis of the 2nd CRFs model showed that it model is a highly efficient method to approach name entity recognition.

A biological information extraction technique using the CRFs model has proposed to address NER as it is applied to recognize classes of words. The 1st and the 2nd CRFs were compared and showed that the later model provided the best NER results.

3.5 Comparison of Biological Named Entity Recognition Models Performance

3.5.1 Introduction

The recently year, there are many techniques to generate the NER models because of they are a prerequisite for many applications working on text, such as information retrieval, information extraction or document classification [52]. The NER models almost generated with machine learning techniques. The state of the art of NER models around F-score of 77.57% [55]. Hence, this work was focus on the comparison between biological NER models performance which generated with machine learning techniques such as Dictionary based and SVMs.

3.5.2 Methods and Dataset

This work, the training and testing dataset from GENIA corpus is used to generate the NER models with four methods (1st CRFs, 2nd CRFs, SVMs and Dictionary Based). The finally they are compared the prediction performance by recall, precision and F-Measure scale. The framework of this work is defined as **Figure 3.9**

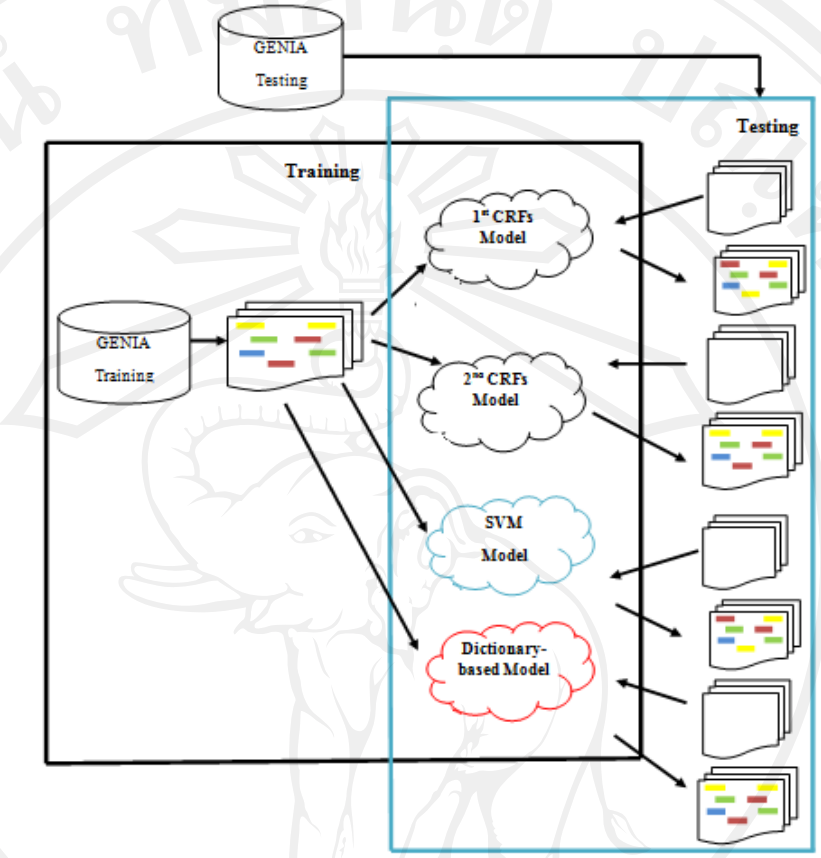


Figure 3.9 The framework of comparison of Biological Named Entity Recognition models performance

The concept of CRFs is described last section. This part will explain only the concepts of the Dictionary Based and SVMs as follow,

1) Dictionary Based

Dictionaries are intrinsically interesting and a study of their features can be of great usefulness to investigate the properties of the sequences they have been extracted from [73]. Given an unstructured text string x consisting of a sequence of tokens $x_1 \ldots x_n$ where each token is either a word or a delimiter. Let Y denote the set of entity types (such

as title, person-names and city-names) to be recognized from x. The entity recognition task is to segment x into a sequence s of segments $s_1 \ldots s_p$ where each segment s_j is either labeled with an entity type from Y or a special label other denoting none of entities. For ease of notation, we assume Y includes the other label. Thus, each segment s_j is associated with a start position t_j, an end position u_j and a label $y_j \in$ Y . Further, since adjacent segment about t_j and u_j always satisfy $1 \leq t_j \leq u_j \leq |$ x $|$, $t_j+1 = u_j + 1$; $u_p = |$ x $|$, and $t_1 = 1$. Entities were recognized in unstructured text based on various simultaneous clues in and around the proposed segment such as capitalization patterns and delimiters. Other clues like the implicit ordering of labels and matches with known list of entities are also utilized.

2) Support Vector Machines (SVMs)

As described in Chapter 2, Support Vector Machines (SVMs) are well-known for their good generalization performance for machine learning approaches to solve two-class pattern recognition problems. In the topics of NLP, SVMs have been applied to text classification, and are reported to have achieved high accuracy without falling into over-fitting even with a large number of words taken as the features [57]. Suppose a set of training data for a two-class problem is given as follows:

$\{(x_1, y_1), \ldots, (x_N, y_N)\}$, where $x_i \in R^D$ is a feature vector of the *i-th* sample in the training data and $y \in \{+1,-1\}$ is the class to which x_i belongs. In their basic form, an SVMs finds a linear hyperplane that separates the set of positive examples from the set of negative examples with *maximal margin* (the margin is defined as the distance of the

hyperplane to the nearest of the positive and negative examples). In a basic SVMs framework, the positive and negative examples are separated by a hyperplane written as: $(w.x) + b = 0 \;\; w \in R^N, b \in R$, SVMs find the optimal-hyperplane (optimal parameter w, b) which separates the training data into two classes precisely. The linear separator is defined by two elements: a weight vector w (with one component for each feature), and a bias b which stands for the distance of the hyperplane to the origin. The classification rule of a SVMs is:

$$sgn(f(x, w, b)) \tag{3.23}$$

$$f(x, w, b) = < w.x > + b \tag{3.24}$$

being x the example to be classified. In the linearly separable case, determining the maximal margin hyperplane (w, b) can be stated as a convex quadratic optimization problem with a unique solution: *minimize* $\|\mathbf{w}\|$, *subject to the constraints* (one for each training example):

$$y_i(< w.x_i > + b) \geq 1 \tag{3.25}$$

The SVMs model has an equivalent dual formulation, characterized by a weight vector α and a bias b. In this case, α contains one weight for each training vector, indicating the importance of this vector in the solution. Vectors with non null weights are called *support vectors*. The dual classification rule is:

$$f(x, \alpha, b) = \sum_{i=1}^{N} y_i \alpha_i < x_i.\mathrm{x} > + b \tag{3.26}$$

The α vectors can also be calculated as a quadratic optimization problem. Given the optimal α^* vector of the dual quadratic optimization problem, the weight vector w^*that realizes the maximal margin hyperplane is calculated as:

$$w^* = \sum_{i=1}^{N} y_i \alpha_i^* x_i \tag{3.27}$$

The b^* has also has a simple expression in terms of $w^*$and the training examples $(x_i, y_i)_{i=1}^N$ The advantage of the dual formulation is that efficient learning of non-linear SVMs separators can be acheived by introducing *kernel functions*. Technically, a *kernel function* calculates a dot product between two vectors that have been (non-linearly) mapped into a high dimensional feature space. Since there is no need to perform this mapping explicitly, the training is still feasible although the dimension of the real feature space can be very high or even infinite.

By simply substituting every dot product of x_i and x_j in dual form with any *kernel function* $K(x_i, x_j)$. SVMs can handle non-linear hypotheses. Among the many kinds of *kernel functions* available, the $d - th$ polynomial kernel is considered;

$$K(x_i, x_j) = (x_i . x_j + 1)^d \tag{3.28}$$

Use of $d - th$ polynomial kernel function allows to build an optimal separating hyperplane which takes into account all combination of features up to d.

The SVMs has advantage over conventional statistical learning algorithms, such as Hidden Markov Models, Maximum Entropy Models, from the following two aspects:

1) SVMs has high generalization performance independent of the dimension of the feature vectors. Conventional algorithms require careful feature selection, which is usually optimized heuristically, to avoid overfitting. Thus, it can more effectively handle the diverse, overlapping and morphologically complex Indian languages.

2) SVMs can carry out their learning with all combinations of given features without increasing computational complexity by introducing the *Kernel function*. Conventional algorithms cannot handle these combinations efficiently.

There are many toolkits developed based on SVMs such as the [1]YamCha toolkit which is the tool for detecting classes in documents and formulating the NER task as a sequential labeling problem; [2]TinySVM is an implementation of SVMs for the problem of pattern recognition. For these reasons, SVMs is a new generation learning algorithm based on recent advances in statistical learning theory, and has already been applied to a large number of real-world applications, such as text categorization and hand-written character recognition.

3.5.3 Results and Discussion

To examine the Biological Named Entity Recognition Models. This section is designed to evaluate the performance of the four models shown in **Table 3.5**.

[1]http://chasen-org/ taku/software/yamcha/
[2]http://cl.aist-nara.ac.jp/ taku-ku/software/TinySVM

Table 3.5 Comparison of Biological Named Entity Recognition Models Performance

Label	Recall (%)	Precision (%)	F-Measure (%)
1st CRFs	56.18	69.50	62.13
2nd CRFs	**99.73**	**99.88**	**99.81**
Support Vector Machines (SVMs)	61.97	71.15	66.24
Dictionary Based	55.16	51.46	53.25

The JNLPBA2004 corpus was a data set used to generate models. The CRFs models (1st CRFs and 2nd CRFs) were compared with the SVMs and Dictionary Based methods. The results showed that the 2nd CRFs method had the highest F-Measure at 99.81% while the SVMs and 1st CRFs were nearly equal to each other at 66.24% and 62.13% respectively. Dictionary based model returned the lowest results at 53.25%. The CRFs method was conclude that was suitable for biological name entity recognition.

In order to examine, how biological terms identification affects information extraction, the next chapter will compare results from biological named entity recognition models.

3.6 Conclusion

This chapter focuses on biological named entity recognition with machine learning. The CRFs was considered as the techniques to generate the model prediction. It's the graphical model which discriminative model and predict without the label bias problem. In process of information retrieval, the CRFs could help to improve the performance by identify feature selection and support LSI to find the latent concepts. But

the name of biological terms almost have long, the 1st CRFs was not support every case. The results of the experiment showed that the performance of the 2nd CRFs model returned the high performance than the others. It's suitable for identify the biological named entity recognition from literature which was necessary for the information extraction.