CHAPTER 9

GRAPHICAL MODELS FOR PROTEIN FUNCTION AND STRUCTURE PREDICTION

MINGJIE TANG,[1,†] KEAN MING TAN,[2,†] XIN LU TAN,[2] LEE SAEL,[1,3]
MEGHANA CHITALE,[1] JUAN ESQUIVEL-RODRÍGUEZ,[1]
and DAISUKE KIHARA[1,3]
[1]Department of Computer Science, Purdue University, West Lafayette, Indiana
[2]Department of Statistics, Purdue University, West Lafayette, Indiana
[3]Department of Biological Sciences, Purdue University, West Lafayette, Indiana

9.1 INTRODUCTION

One of the central aims of bioinformatics is to reveal hidden structures of biological entities and systems, such as protein/gene sequences, protein structures, and interactions, in order to understand how they are constructed and related to each other. Unveiling structures and relationships will also enable prediction and classification of unknown data. For this task, various machine learning techniques have been introduced in the bioinformatics field. In fact, the two-decade history of bioinformatics can be viewed, in one sense, as the history of adopting and applying new algorithms to biological data. The role of predictive and classification algorithms in bioinformatics will only become more important as the number and types of data increase.

In the early 1990s, *dynamic programming* (DP) was successfully applied to protein and DNA sequence alignment [1, 2]. Pairwise sequence alignment tools later evolved into tools for sequence database search [3–5]. Computational sequence alignment and database search tools have revolutionized biological studies. Nowadays sequence alignment and sequence database searches are routinely used in every molecular biology laboratory for obtaining clues for gene function and functional sites. DP has been further applied for RNA secondary-structure prediction [6], protein three-dimensional (3D) structure comparison [7, 8], and protein 3D structure prediction [9–11]. In the late 1990s, the hidden Markov model (HMM) became very popular partly due to the publication of a seminal book, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids* [12]. HMM can handle a

[†]These two authors have equal contribution.

Biological Knowledge Discovery Handbook: Preprocessing, Mining, and Postprocessing of Biological Data, First Edition. Edited by Mourad Elloumi and Albert Y. Zomaya.

variance of biological sequences and structures in a probabilistic fashion and thus allows more sensitive searches of similarities in proteins. HMM has been widely applied for protein secondary-structure prediction [13, 14], transmembrane region prediction [15, 16], and 3D structure prediction [17, 18]. The artificial neural network (NN) [19] is another example that was widely applied in various prediction and classification problems in bioinformatics [20–22]. The support vector machine (SVM), which was introduced in the field around the beginning of this century, has gradually taken over the NN in various prediction methods [23–25]. Of course, DP, HMM, NN, and SVM are only a few examples among many others [26–30] of successfully applied machine learning techniques in bioinformatics.

In this chapter, we review bioinformatics applications of two emerging graphical models, the *Markov random field* (MRF) and the *conditional random field* (CRF). The main advantage of these two methods is that they can represent dependencies of variables using graphs. Since many biological data can be described as graphs, both methods have gained increasing attention in the bioinformatics community. We first briefly describe the MRF and the CRF in comparison with the HMM. What follows are applications of the two graphical models, focusing on gene prediction, protein function prediction, and protein structure prediction. These applications benefit from the graphical models by being able to represent dependencies between graph nodes, which contributed to improvement of prediction accuracy.

9.2 GRAPHICAL MODELS

A graphical model is able to represent complex joint distributions of a large number of variables compactly using a set of local relationships specified by a graph. It provides us a convenient way to understand and express complicated joint distributions. Each node in the graph represents a random variable and nodes are connected by edges, which describe the dependency between the variables.

In this section, we will discuss some basics of the graphical model. First, we discuss the directed graphical model, that is, the edges of the graphs have a particular directionality indicated by arrows (Bayesian networks or belief networks). Then, we proceed to undirected graphical models such as MRFs and CRFs. In the following section, we mainly consider the supervised learning process. For the set of variables (X, Y), X represents the input variables that are observed, and Y is the output variable which we want to predict.

9.2.1 Directed Graphical Model

A directed graph G is a set of vertices and edges $G(V, E)$ where $V = \{V_i\}$ is a set of nodes and $E = \{(V_i, V_j)\}$ is a set of edges with directions from V_i to V_j. We assume G is acyclic. Each V_i represents a random variable. Each node V_i has a set of parent nodes V_{parents}. The structure of the graph defines the conditional independence relationship among the random variables. The joint probability of all variables V can be calculated as the product of the conditional probability of each variables conditioned on its parents as presented in the following equation

$$\Pr(V) = \prod_{V_i \subset V} \Pr(V_i | V_{\text{parents}}) \tag{9.1}$$

The most widely used directed graphical model is the Naive Bayes model [31] (Figure 9.1). It predicts a single class variable y given a vector of features $x = (x_1, x_2, \ldots, x_n)$. For example, if all the features have discrete binary values $\{0, 1\}$, then a general distribution over x corresponds to a table of 2^n numbers of possible combinations of features for

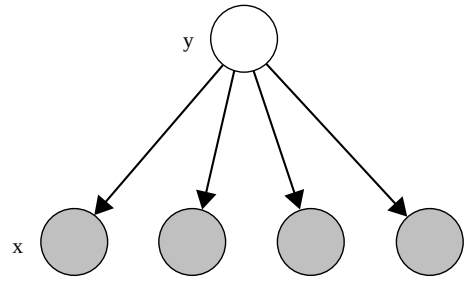

FIGURE 9.1 Naive Bayes model.

each class. Since the dimension of the number of possible combinations of features increases significantly as n becomes large, modeling $\Pr(X)$ will suffer from the curse of dimensionality. However, the conditional independence property in the graphical model enables us to get the joint distributions easily.

The naive Bayes has the conditional independence assumption that every attribute x_i is conditionally independent of other attributes x_j on the class label. The formula for the naive Bayes model and the graphical representation follows:

$$\Pr(x_1, x_2, \ldots, x_n, y) = \Pr(y) \prod_{i=1}^{n} \Pr(x_i|y) \tag{9.2}$$

The conditional independence assumption is one of the basic ideas of a graphical model because it can reduce the number of parameters to be estimated. The conditional independence is usually called D-separation for the joint distribution in a graph model [32].

9.2.2 Undirected Model (Markov Random Field)

An undirected graphical model can also be represented by $G = (V, E)$, except that the edges in E are undirected. The widely used undirected graph model is the MRF. The MRF allows one to incorporate local contextual constraints in labeling problems in a principled manner and is applied in vision, bioinformatics, and many other fields. It uses a concept called the Markov blanket of a node, which assumes that the node is conditionally independent from all the other nodes but conditioned only on the set of neighboring nodes [33]. By the conditionally independent property in the MRF, the joint probability for graph nodes (variables) is factorized. For example, we consider two nodes X_i, X_j in the graph, and they are not directly connected by edges in the graph (Figure 9.2). Then the conditional distribution for X_i and X_j would be illustrated by

$$\Pr(x_i, x_j| \sim \{x_i, x_j\}) = \Pr(x_i| \sim \{x_i, x_j\})\Pr(x_j| \sim \{x_i, x_j\} \tag{9.3}$$

where $\sim\{x_i, x_j\}$ is the set of nodes in the graph with the node $\{x_i, x_j\}$ removed. By this conditional independence property, we can represent the joint distribution in terms of cliques in the graph. A clique is a fully connected graph where every node has links to all the other nodes in the graph. The size of the clique can vary from binary to the entire graph in consideration. We can also define an undirected graphical model as the set of all distributions:

$$\Pr(x, y) = \frac{1}{z} \prod_{A} \psi_A(X_A, Y_A) \tag{9.4}$$

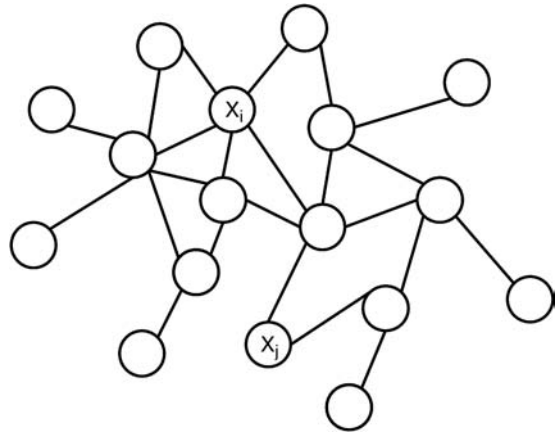

FIGURE 9.2 Markov random field.

where ψ_A is the potential function defined on the clique A in the graph and Z is a normalized factor defined by

$$Z = \sum_{X,Y} \prod_{A} \psi_A(X_A, Y_A) \tag{9.5}$$

where Z ensures that the distribution $\Pr(x, y)$ is correctly normalized. The normalization constant is one of the major limitations of the undirected graphs. For example, suppose we have a model with M discrete nodes each of which has K states. Evaluation of the normalization term involves summing over the K_M state, which is exponential with the size of the model. The model parameter is learned from the potential functions ψ [34]. The potential function ψ takes positive values. Traditionally, we express the potential function, which is strictly positive as exponentials, so that

$$\psi = \exp[-E(X, Y)] \tag{9.6}$$

where $E(X, Y)$ is called an energy function and the exponential representation is called the *Boltzmann distribution.* The joint distribution is defined as the product of potentials, and so the total energy is obtained by adding the energies of each of the maximal cliques.

9.2.3 Discriminative versus Generative Model

In the framework of supervised learning, a new test data set can be classified using either a generative model or a discriminative model [35, 36] (Figure 9.3).

9.2.3.1 Generative Model The first step in constructing a generative model is to find the prior distribution $\Pr(Y_k)$ and the likelihood function or conditional distribution $\Pr(X|Y_k)$ for every k, where k is the possible classes for the label. To predict the label for a new observation X, we need to calculate the conditional probability of a particular class Y_k given the observation X. This can be obtained using the Bayes rule:

$$\Pr(Y_k|X) = \frac{\Pr(Y_k)\ \Pr(X|Y_k)}{\Pr(X)} \tag{9.7}$$

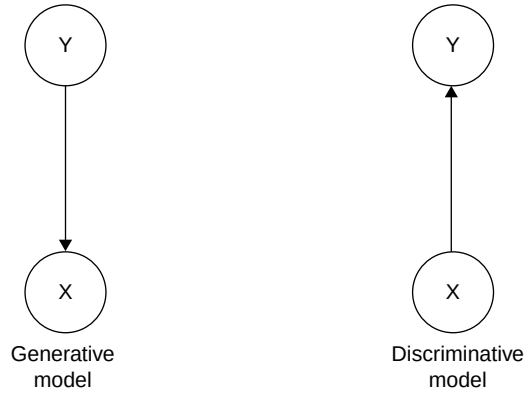

FIGURE 9.3 Graph representation of generative model and discriminative model.

where $\Pr(X)$ is a normalizing constant and is obtained by marginalizing Y for $\Pr(X, Y)$. Then, we can assign our output to the class that yields the highest posterior probability given an observation and make a decision by using a user-specified cost function. This is a generative model because we explicitly model the distribution of the observation, X.

9.2.3.2 Discriminative Model A generative model such as the naive Bayes, HMM, and MRF computes the joint distributions $\Pr(X, Y)$. However, the difficulty lies primarily in computing $\Pr(X)$, especially when X is in high dimension and when most of the features are correlated. As we addressed in the naive Bayes model, we assume that features are conditionally independent given the class label. However, this assumption may hurt the performance of the generative model on a general application as compared to a discriminative model such as the logistic regression model [35].

For sequential data such as gene or protein structure, a generative model like HMM has the assumption that Y_i only depends on feature X_i. However, since some of the features might be missing in the training data, it is therefore necessary to explore and use X_i,'s neighbors' information. The generative model fails to capture this information. Moreover, usually the purpose of a model is to predict Y rather than modeling the distribution $\Pr(X)$. Hence, computing the joint distribution for X and Y would not be necessary to infer the posterior probability of Y.

A discriminative model builds a function that maps variable X directly to one decision boundary described by a posterior probability $\Pr(Y|X)$. The discriminative model does not need to make strict independence assumptions on the observations, and it is able to choose features in an arbitrary way. Also, the transition probability between labels may depend not only on the current observation but also on past and future observations.

9.2.4 Sequential Model

Classical classifiers such as naive Bayes and logistic regression predict only a single class of variable given an observation. These models fail when it comes to prediction of a label sequence given an observation sequence because both models assume that the predicted labels are independent from each other. It is called structure prediction or a sequential model because of the topological structure of input and output data. In bioinformatics,

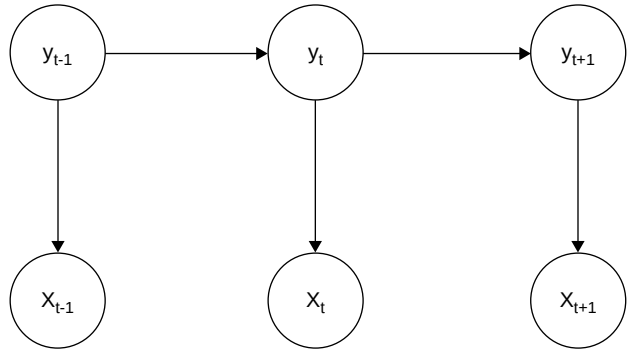

FIGURE 9.4 Hidden Markov model.

DNA sequences, protein sequences, and protein 3D structures are typical structure data. Two widely developed graph models, the HMM and the CRF, will be discussed in the following sections.

9.2.4.1 Hidden Markov Model The HMM [37] has been used extensively in various scientific fields for segmenting and predicting label sequences for a given observation sequence. In Figure 9.4, X is the observed sequence and Y is the label sequence, which is sometimes called the hidden state. The subscript t indicates the position of the label and the observation sequence. The arrows between hidden states represent the translation possibility $\Pr(y_t|y_{t-1})$, and the arrow between a hidden state and an observation represents the emission possibility $\Pr(x_t|y_t)$. There are three assumptions made in the HMM:

1. *Markov Assumption* The current state (label) is only dependent on the previous state. This means that each state y_t is independent from all other states, $y_1, y_2, \ldots, y_{t-2}$ given its previous state y_{t-1}.
2. *Stationary Assumption* The state transition probability is independent of the actual position at which the transition takes place.
3. *Independence Assumption* The current input x_t is statistically independent from the previous output x_{t-1} given the current state y_t.

In addition, each observation x_t depends only on the current state y_t. Based on these assumptions, we then have Equation (9.8) as the joint probability:

$$\Pr(Y, X) = \prod_{t=1} \Pr(y_t|y_{t-1})\Pr(x_t|y_t) \tag{9.8}$$

Here, $\Pr(x_t|y_t)$ is the transition matrix of our observation and $\Pr(y_t|y_{t-1})$ is the state transition matrix.

There are three fundamental questions in the HMM [37].

1. How do we estimate the parameters?
2. Which HMM is most probably used to generate the given sequence? This is also called the evaluation process.
3. Given an observation sequence, what is the most likely label sequence which has generated the observation sequence?

The HMM is a supervised learning method. In other words, given a set of training data sets, we have to optimize the parameters (transition and confusion matrices). We can consider the HMM model as an extension of the mixture Gaussian model by introducing the transition matrix between different hidden states. To optimize parameters, the expectation–maximization (EM) algorithm has been used extensively during the past decades [34]. As for the second question, one can use the forward algorithm to calculate the probability of an observation sequence [37]. The third question is to predict the label sequence for a new observation based on the learned parameters. In this case, we want to find label sequence Y^* that maximizes the conditional probability of the label sequence given an observation sequence:

$$y^* = \arg\max_Y \Pr(y|x) \tag{9.9}$$

The above equation is obtained using the Bayes rule [Equation (9.7)]. The probability $\Pr(X)$ does not depend on Y, so we can drop it. Now, the objective is to find the maximum of $\Pr(X, Y)$. Using a naive approach, we can list all the possible label sequences and see which one has the highest posterior probability. The time complexity is $O(S^T)$, where S is the number of possible labels and T is the total number of states in the sequence. But this task can be performed efficiently using a dynamic programming technique known as the Viterbi algorithm [37].

9.2.4.2 Weakness of Hidden Markov Model

As mentioned before, HMM is a generative model. It models the joint distributions of both observation sequences and label sequences. However, in most cases, our interest lies primarily in predicting a label sequence given an observation sequence.

In addition, the HMM assumes that each observation x_t is only dependent on its label y_t, and, hence, is incapable of modeling multiple features from the observation. However, in real-world applications, observation sequences tend to be dependent and are best represented in terms of multiple interacting features and long-range dependencies between observation elements.

9.2.5 Maximum-Entropy Markov Models and Label Bias Problem

Maximum-entropy Markov models (MEMMs) is a form of conditional model used to label sequential data given an observation (Figure 9.5). While the HMM is a generative model,

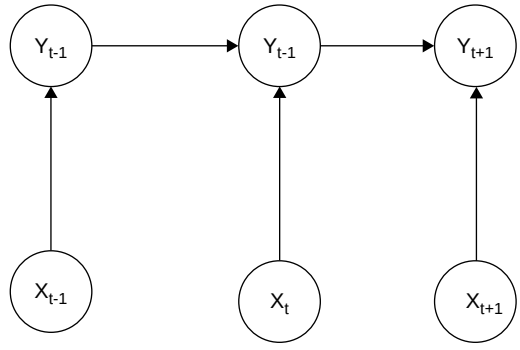

FIGURE 9.5 Maximum-entropy Markov models.

the MEMM is a discriminative model. Hence, the MEMM does not need to model the joint distributions of both observation and label sequences. Also, unlike the HMM, the MEMM is able to model multiple features from the observation using the function f_k in Equation (9.10). The MEMM had been applied to various fields for labeling sequential data. These include but are not limited to part-of-speech (POS) tagging and segmentation of text document problems [38].

$$\Pr(y_t|y_{t-1}, x_t) = \frac{1}{Z(x_t, y_{t-1})} \exp\left(\sum_{k=1}^{K} \lambda_k f_k(x_t, y_t, y_{t-1})\right) \tag{9.10}$$

Equation (9.10) is the state observation transition function using in the MEMM for time $(t-1)$ to t. This equation provides the probability of current state y_t given the previous state y_{t-1} and observation x_t. Computation of the likelihood of the whole state sequence involves multiplying the above probability of each of the time points together. Here, f_k is the kth feature function and λ_k is the weight assigned to f_k.

Although the MEMM has the advantageous characteristics over the HMM, the MEMM has an undesirable property called the label bias problem [39]. This is due to the per-state normalization at each transition. Hence, the model will tend to favor the label with fewer outgoing transitions. The details about the label bias problem can be found in a paper by Lafferty et al. [39]. More information on the MEMM can be found in a paper by McCallum et al. [38]. The conditional random field was proposed to solve the label bias problem and also inherit all the good properties of the MEMM.

9.2.6 Conditional Random Field

The CRF was first proposed by Lafferty et al. [39] in the context of segmentation and labeling of text sequences. The CRF models the posterior distribution $p(Y|X)$ directly. Because the CRF is a undirected graph model (Figure 9.6), it does not have arrows between the observation and hidden state as in the HMM. It has been proven to be very effective in many applications with structured outputs, such as information extraction, image processing, and parsing [40–42]. The CRF is able to model the conditional probability of a label sequence with nonindependent and interacting features of the observation sequence due to its conditional nature.

Since the CRF does not have the assumption for the distribution of the inputs $P(X)$ and finds the decision boundary directly, it can be regarded as an extension of logistic regression

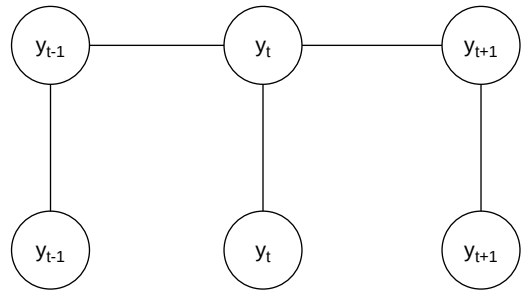

FIGURE 9.6 Conditional random field.

to model sequential data. The relationship among the CRF, logistic regression, and naive Bayesian model can be found in some previous works [35, 36].

Formally, we define $G = (V, E)$ as an undirected graph such that $Y = (Y_u)_{u \in V}$, and Y is indexed by the vertices of G. The couple (X, Y) is said to be a conditional random field if, when conditioned on X, the random variable y_t obeys the Markov property with respect to the graph: $\Pr(y_t | y_{S-\{t\}}, X) = \Pr(y_t | y_{Nt}, X)$, where $S\backslash\{t\}$ is the set of all the nodes in the graph except the node t and N_t is the set of neighbors of node t in G [39]. The conditional distribution for Y given X is

$$\Pr(Y|X) = \frac{1}{Z} \sum_{t \in \text{Nodes}} \exp\big[\sigma(y_t|X) + \theta(y_t, y_{ng}|X)\big] \tag{9.11}$$

where t is a node in the graph G and $\sigma(y_t|X)$ is an association potential function that reflects the feature X that has an influence on the label y_t. It can be a logistic regression model, a Bayesian model, a SVM, or a kernel regression model. The second term $\theta(y_t, y_{ng}|X)$ is the *association potential function* between y_t and its neighbors y_{ng}. These neighborhood nodes y_{ng} can be nodes in a clique of the graph in which the size of the clique can be arbitrary. The association potential function shows the influence among related predicted labels. The potential functions defined in Equation (9.11) transform the traditional logistic regression that fits a single scalar to fitting and labeling sequential data. Therefore, the CRF is similar to the logistic regression for classification where y is now sequential data, which can be discrete or continuous. The variable Z is the normalization factor given by

$$Z = \sum_{Y} \sum_{t \in \text{Nodes}} \exp\big[\sigma(y_t|X) + \theta(y_t, y_{ng}|X)\big] \tag{9.12}$$

So far we defined the graph as a linear chain in Figure 9.6. However, the structure of the graph G can be arbitrary provided that the conditional independence in the label sequences is correctly modeled. However, due to the high computational cost for training a general CRF model, the most widely used graph structure is the linear chain CRF as shown in Figure 9.6. The maximum clique among the label sequence is then a pairwise clique. From Equation (9.2), the joint distribution over the labels Y given the observation sequence X can be written as

$$\Pr(Y|X) = \frac{1}{Z(x)} \exp\left(\sum_{c \subset C} \sum_{k=1}^{K} \big[\lambda_k f_k(y_{t-1}, y_t, X) + \mu_k g_k(y_t, X)\big] \right) \tag{9.13}$$

Here f_k corresponds to θ and g_k corresponds to σ in Equation (9.11). Since the loss function [Equation (9.13)] is convex, it will always give us the global maximum if we optimize the function [39]. In the remainder of this chapter, Equation (9.13) is simplified by substituting the feature functions $f_k(y_{t-1}, y_t, X)$ and $g_k(y_t, X)$ to $f_k(X, i, y_{t-1}, y_t)$. Therefore, we obtain the equation

$$\Pr(Y|X) = \frac{1}{Z(X)} \prod_{t} \exp\left(\sum_{k=1}^{K} \lambda_k f_k(X, t, y_{t-1}, y_t) \right) \tag{9.14}$$

While the MEMM normalizes the probability on a per-state basis, the CRF normalizes the probability over the whole sequence. Hence, the MEMM will tend to be biased toward

states with fewer outgoing transitions (the label bias problem). To the extreme, the MEMM will ignore the observation if there is only one single outgoing transition [39].

We note that every HMM can always be written as a special instance of the linear chain CRF. We define a single feature function for each of $\Pr(y_t|y_{t-1})$ and $\Pr(x_t|y_t)$ as

$$f_k(y_{t-1}, y_t, x) = \begin{cases} 1 & \text{if } x_{t-1} = y' \text{ and } y_t = y \\ 0 & \text{otherwise} \end{cases} \tag{9.15}$$

$$g_k(y_t, x) = \begin{cases} 1 & \text{if } y_t = y' \text{ and } x_t = x \\ 0 & \text{otherwise} \end{cases} \tag{9.16}$$

Here, the parameters λ_k and μ_k corresponding to these features are equivalent to the logarithms of the HMM transition and emission probabilities $\Pr(y_t|y_{t-1})$ and $\Pr(x_t|y_t)$. However, the CRF is a more general model because it allows arbitrary dependencies on the observation sequence. In general, a feature function can be any mapping $F_j : X \times Y \to R$.

Often, feature functions are just binary indicators (absence/not). For example, let us consider a simple DNA functional site model by considering motifs using the CRF. Because the CRF modeling is a supervised learning process, we use training data of DNA sequences with sequence patterns X and its related label Y. In order to capture the dependencies on the observation sequence patterns in one sequence, we can define one of feature functions as

$$f_1(y_{t-1}, y_t, x) = \begin{cases} 1 & \text{if } x_t = \text{CGCC}, y_t = \text{motif 1} \\ 0 & \text{otherwise} \end{cases} \tag{9.17}$$

$$f_2(y_{t-1}, y_t, x) = \begin{cases} 1 & \text{if } x_t = \text{TATA}, y_t = \text{motif 2}, y_{t-1} = \text{motif 1} \\ 0 & \text{otherwise} \end{cases} \tag{9.18}$$

For the feature functions f_1 and f_2, the corresponding weights λ_1, λ_2 reflect the probability for the observed sequence patterns and related sequence label. For example, whenever the CRF sees the sequence TATA, it will prefer to label the state as motif 1 because we observe TATA with a higher weight for motif 1.

9.2.6.1 Parameter Estimation

Maximum-likelihood estimation is widely used to learn the parameters in the CRF. Maximum-likelihood estimation chooses values of the parameters such that the logarithm of the likelihood, known as the log likelihood, is maximized [39]. The log likelihood for the CRF can be written as

$$l(\lambda) = \sum_{i=1}^{N} \log \Pr(y^{(i)}|x^{(i)}) \tag{9.19}$$

where N is the number of training sequences and λ is the vector of the parameters. Since typically there are thousands of parameters used in a CRF model, the CRF may be overfitted to the training data set. To avoid overfitting, a regularization factor is usually added to the log likelihood. Substituting Equation (9.13) into Equation (9.19) and adding a Gaussian regularization, we obtain the expression

$$l(\lambda) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k\left(y_t^i, y_{t-1}^i, x_t^i\right) - \sum_{i=1}^{N} \log Z(X^i) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2} \tag{9.20}$$

Note that the subscript i denotes the ith data samples, t is the position in a data, and k is the kth feature function. In order to maximize the log-likelihood function [Equation (9.20)], we differentiate the log-likelihood function with respect to the parameter λ_k, which yields

$$\frac{\partial l(\lambda)}{\partial \lambda_k} = \sum_{i=1}^{N}\sum_{t=1}^{T} f_k\left(y_t^i, y_{t-1}^i, x_t^i\right) - \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{y,y'} f_k\left(y, y', x_t^i\right) P\left(y, y' | x^{(i)}\right) - \sum_{k=1}^{K}\frac{\lambda_k}{\sigma^2} \quad (9.21)$$

Because Equation (9.21) does not have closed-form solutions, we cannot obtain the λ_k value by simply setting Equation (9.21) equal to zero. Thus, we need to take a different strategy to obtain the parameter values. For example, Lafferty et al. used the gradient ascent approach [39]:

$$\lambda^{n+1} = \lambda^n + \alpha\nabla \quad (9.22)$$

where α is the scaling factor and ∇ is the first deviant of the log-likelihood function [Equation (9.21)] for λ. However, the conventional approach requires too many iterations to be practical. The quasi-Newton method converges much faster than the original approach based on iterative scaling [36, 43]:

$$\lambda^{n+1} = \lambda^n - H^{-1}\nabla \quad (9.23)$$

where H is the Hessian matrix whose elements comprise the second derivatives of $l(\lambda)$ with respect to the components of λ. Other parameter learning methods can be found in some tutorials [36, 43].

9.2.6.2 Inference Given the learned model, we are interested in labeling an unseen instance given an observation sequence and computing the marginal distribution. This is essentially the same as the HMM inference problem, because both of them aimed to find the maximal posterior possibility label sequence. In order to find the most probable label for a given observation sequence, we can use the Viterbi algorithm similar to the HMM. There are two major approaches to compute the marginal probability or the conditional probability: the exact inference and approximate inference methods.

The elimination algorithm is the basic method for the exact inference. The main idea is to efficiently marginalize out all the irrelevant variables using factored representation of the joint probability distribution. Forward–backward algorithms such as those used for the HMM can be applied to the linear chain CRF as well. The algorithm is further described in detail in papers by Hoefel and Elkan [44] and Lafferty et al. [39].

The computational complexity of the exact inference algorithms increases exponentially when the size of the cliques are very large. Hence, approximation methods are preferred in many practical applications. These approximation methods include the sampling method [45], the variation methods [46], or the loopy belief propagation [47]. A very efficient approach for high-dimensional data is the Markov chain Monte Carlo, including Gibbs sampling and Metropolis–Hastings sampling [48, 49]. The loopy belief propagation applies the original belief propagation algorithm to graphs even when they contain loops [50].

9.2.6.3 Conditional Random Field versus Markov Random Field Kumar and Hebert [41] compared two major differences between the CRF and the original MRF framework. First, in the conditional random fields, the association potential function at any state is a function of all the observations X while in the MRF (with the assumption of conditional

independence of the data) the association potential is a function of data only at that state X_t. Second, the interaction potential for each pair of nodes in MRFs is a function of only labels, while in the conditional models it is a function of labels as well as all the observations X. The interaction among labels in MRFs is modeled by the term $P(Y)$, which can be seen as a prior under the Bayesian framework. However, these label interactions do not depend on the observed data X. This difference plays a central role in modeling arbitrary interactions in sequential data. Above all, the MRF and HMM belong to the generative model which is based on modeling the joint distributions $\Pr(X, Y)$, while the CRF is a extension of the discriminative model such as the maximum-entropy model (usually regarded as logistic regression). The discriminative model is based on modeling the conditional distribution $\Pr(Y|X)$ directly.

9.2.6.4 General Conditional Random Field In the previous sections, we have mainly discussed the learning and inference steps for the linear chain CRF. However, those methods can also be extended to general graphs by considering the long-range dependencies over labels. Several researchers have recently demonstrated that certain CRF graph structures are useful for handling special applications. For example, the Skip-Chain CRF takes into account the probabilistic dependencies between long-distance mentions for information extraction [51]. Liu et al. [52] extended the linear chain CRF to predict a type of protein structure [52]. In addition, several researchers modified the original CRF and proposed to integrate other techniques, for example, the Bayesian CRF [53], the boosting [54], and the neural network [55]. Other research directions include development of the semisupervised learning CRF in order to deal with insufficient training label data. Some recent developments of the semisupervised learning methods can be found in recent papers [56, 57].

9.2.7 Summary of Models and Available Resources

Each model has its strengths and drawbacks and one needs to examine which model (or if any model) is suitable for a particular problem. In Table 9.1, we summarize the pros and cons of the models discussed above.

There are several useful information sources and software for the MRF and CRF. For the MRF, some books, such as *Markov Random Field Modeling in Image Analysis* [58] and *Image Processing: Dealing with Texture* [59], provide details of the method. In addition, we provide a list of resources on the Internet we found useful (Table 9.2).

9.3 APPLICATIONS

In this section, we discuss some applications of the MRF and CRF on gene prediction, protein function prediction, and protein structure prediction.

9.3.1 Gene Prediction Using Conditional Random Fields

One of the first steps in gene analysis is to determine the coding regions given DNA sequences, which is called the gene prediction problem or the gene-finding problem. Generally gene prediction considers prediction of protein-coding genes, which are the sections of DNA that are translated to produce proteins [60]. According to the central dogma of molecular biology in eukaryotes, a DNA sequence first undergoes a transcription process

TABLE 9.1 Summary of Models

	Pros	Cons
Generative model	Based on the joint distribution, it is easy to calculate the posterior distribution.	Does not need to generate the joint distributions for variables when it comes to making a prediction.
Discriminative model	Do not need to model the joint distributions of the features and variables. It is able to calculate the posterior possibility directly. It just needs to estimate the parameters of the model rather than the whole distributions of data. The model has been extended to different ways by new regulation function, kernel methods, and others.	Could not compute the joint distribution of variables and features.
Naive Bayes model	Although this classifier has a strong assumption of independence of features, this model has been successfully applied in many applications.	Because the feature independence assumption is not promised in most of the applications, this classifier is not always suitable for such data.
Markov random field	Widely used in the computation vision and bioinformatics for modeling data naturally modeled as a graph.	It has the same drawback of generative model and the computation cost is high in most of cases, especially when the size of cliques gets larger.
Hidden Markov model	Suitable and applied for many sequential data, for example, in the bioinformatics, natural language processing, and computation visions.	It assumes that features are independent with each other and often difficult to model (train parameters) distant dependency of hidden states.
Maximum-entropy Markov models	It is proposed to handle the drawbacks of HMM.	Has the label bias problem.
Logistic/linear regression	Do not have assumption of feature independence and can handle nonlinear relation between features.	It does not model dependency between labels.
Conditional random field	It successfully deals with the drawbacks of HMM (assumption of independence of features) and MEMM (the label bias problem). Recently successfully applied in bioinformatics domain.	It still faces the problem of size of training data as the logistic regression. The available software package of CRF is hard to run on different data sets.

in which intergenic regions of the sequence are removed to produce premessenger RNAs (pre-mRNAs). A pre-mRNA is subject to gene splicing, where introns are removed producing a messenger RNA (mRNA). Finally, a mRNA is translated into a protein (amino acid) sequence, where identification of the coding region and *untranslated regions* (UTRs) is involved [61]. Thus, in most gene predictions, the aim is to accurately predict the

TABLE 9.2 Online Resources for MRF and CRF

Title and Web Site	Contents and Comments
Tutorials and Lectures	
Introduction to MRF and its application on computation vision https://engineering.purdue.edu/~bouman/ece641/mrf_tutorial/	Basic examples of graph model and the related source code will be useful to understand the graph model.
Collection for CRF http://www.inference.phy.cam.ac.uk/hmw26/crf/	Links to tutorials, papers, and software for CRF.
Video tutorial for CRF (Charles Elkan) http://videolectures.net/cikm08_elkan_llmacrf/	Basic tutorial for beginner to understand logistic regression model and CRF.
Software	
MRF source code collection http://www.cs.cmu.edu/~cil/v-source.html	It provides the collection of software on computation vision based on MRF.
MRF minimization http://vision.middlebury.edu/MRF/code/	It provides several benchmark software packages to compare MRF application on image segmentation and photomontage problems.
Collection of CRF program http://www.cs.ubc.ca/~murphyk/Software/CRF/crf.html	This website includes several 1D CRF model implementation packages.
Mallet-CRF http://crf.sourceforge.net/	Graphical Models in Mallet (GRMM) developed by the UMASS research group supports arbitrary factor graphs, which subsume Markov random fields, Bayesian networks, and conditional random fields.
CRFsuite http://www.chokkan.org/software/crfsuite/	Written in C++ and implemented in Windows 32, Linux platform.
CRF++ http://crfpp.sourceforge.net/	CRF++ is an open-source implementation of conditional random fields (CRFs) for segmenting/labeling sequential data.

intergenic regions, exons, introns, and UTRs of the target DNA sequence to determine the protein-coding regions in the DNA sequence.

In gene prediction, variants of the CRF, including the semi-Markov CRF–based method, was found to outperform many generative models (e.g., HMM-based methods) [62–65].

In a generative model such as the HMM, the joint probability of the observed sequence **x** and hidden gene structure y is modeled. The joint probability and the parameter θ that is chosen to maximize the joint probability in the training data set are used to predict the genes by selecting the path of hidden labels **y** that maximize the joint probability given a new sequence **x**. To improve gene prediction accuracy, one approach is to incorporate additional information that can come from a multiple-sequence alignment and/or experiential data such as the express sequence tag (EST) alignment information [66]. Unlike the generative models that need to model the relationship between the different types of information, which can quickly become too complex, CRF models can treat different types of information as features without modeling the dependency between the features making leverage of different sources of information possible [62]. Below we describe four variants of the CRF-based gene prediction method.

Culotta et al. first applied the linear chain CRF model in gene prediction and showed that multiple features can be incorporated in the gene prediction [60]. In their model, a base in the DNA sequence is classified as a coding region, an intron, or an intergenic region. To incorporate restrictions in the model, a finite-state machine model is used between label nodes (**y**): the starting state, three coding states, and three intron states. Edges between the states model the restrictions. Coding regions must start with the three bases ATG (in the case of human genes) and end with TAA, TGA, or TAG. The features that they use include (1) 11 base features that incorporate the statistics of amino acid sequences by examining the identity of the previous five bases from the current base position in the given DNA sequence; (2) four histogram features which measure the frequencies of the base conjunctions and disjunctions in a sliding window of size 5; and (3) five homology features, such as the number of hits obtained through a BLAST sequence database search. Here, we will not discuss their CRF model since they used the typical CRF setting, which we extensively described in earlier sections.

Bernal et al. developed an ab initio gene prediction method based on the semi-Markov CRF CRAIG (CRF-based ab initio Genefinder) [63]. An ab initio gene prediction method only considers the information that comes directly from the target DNA sequence and does not use homology information (multiple-sequence alignment). Although ab initio methods have generally a lower accuracy than methods that use homology information, ab initio methods are useful when no homology information is found. CRAIG also uses a finite-state machine to model the gene structure states (**y**): the initial state, the end state, the intergenic state, the start state of a coding region, the end state of a coding region, three exon states, and six intron states. The state machine of eukaryotic genes is shown in Figure 9.7. Instead of associating each label y_j to each base x_j, GRAIG associates DNA sequence **x** to the labeled segment of sequence **s** by relaxing the requirements of the Markov property (semi-Markov property). The gene prediction, given a target sequence **x**, finds the best scoring segmentation where the score, which is analogous to conditional probability $\Pr(y|\mathbf{x})$ in the typical CRF model, is defined as (Equation 2 in the original paper)

$$S_w(\mathbf{x}, \mathbf{s}) = \sum_{j=1}^{Q} w_j \cdot f(s_j, \text{lab}(s_{j-1}), \mathbf{x}) \tag{9.24}$$

That is, DNA sequence $\mathbf{x} = x_1, \ldots, x_p$ is associated with the likely sequence segments $\mathbf{s} = s_1, \ldots, s_Q$, where each segment has information of a starting position $\text{pos}(s_j) = p_j$, a length of segment $\text{len}(s_j) = l_j$, and a state label $\text{lab}(s_j) = y_j$, that is, $s_j = \langle p_j, l_j, y_j \rangle$.

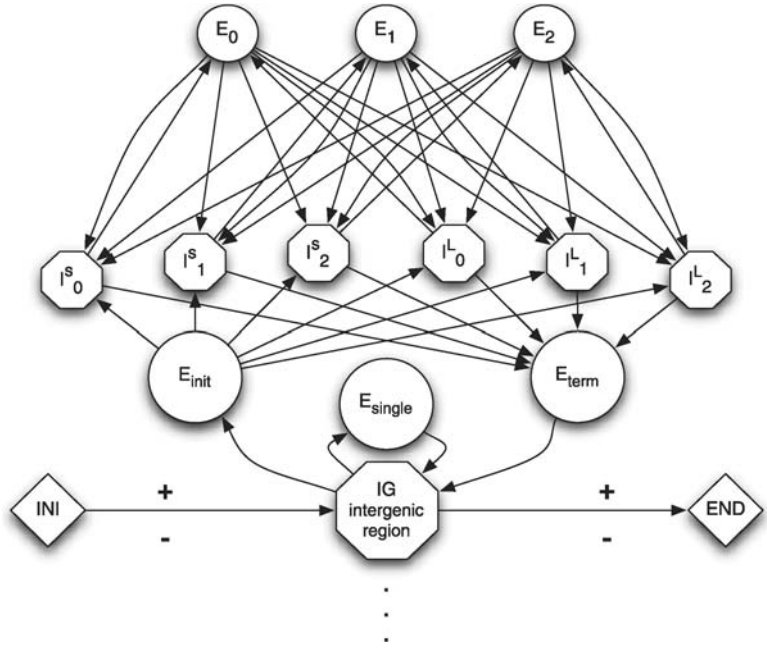

FIGURE 9.7 State machine of eukaryotic genes. The states represent the genomic regions and the directed edges between the states represent the biological signals. The short introns are denoted by I^S and long introns are denoted by I^L. States for exons start with E with subdivision of type of exons in subscript. (Reprinted with permission from [63].)

Each segment is described with a set of features that are computed from the current segment, the label of a previous segment, and the input sequence around the start position of the segment. The complete list of 19 features, such as length and sequence patterns of each state, can be found in Tables 6 and 7 of the original paper by Bernal et al. [63]. The weights of the features are learned using an online approach, where iteratively updating the weights as the input sequences is done one by one. Inference for a sequence $\mathbf{x}$ is computed with a variant of the Viterbi algorithm, which finds the label $\mathbf{s}$ that maximizes Equation (9.24).

Conrad is another semi-Markov CRF-based de novo gene prediction method [62, 67]. De novo methods use multiple-sequence alignments of the input DNA sequence. Conrad, like CRAIG, uses labeled segments S, that is, instead of associating a label to each bases in a sequence, a label $\mathbf{s}$ is represented as a segment of DNA sequence, $s_i = (t_i, u_i, v_i)$, where t_i is the start position, u_i is the end position, and v_i is the label of the segment. Each segment is assumed to only interact with its immediate neighbors such that feature sum F_j can be written as the sum of the Q number of local features f_j:

$$F_j(\mathbf{x}, \mathbf{s}) = \sum_{i=1}^{Q} f_j(v_{i-1}, t_i, u_i, v_i, \mathbf{x}) \tag{9.25}$$

Conrad combines the probabilistic features that are adopted from phylogenetic generalized HMM (GHMM) models [68], which are done by converting the GHMM model to an

equivalent semi-Markov CRF model, with additional nonprobabilistic binary features. The components of the probabilistic features include (1) five reference features that model the nucleotide compositions of homologues sequences; (2) three length features that model the length distributions of introns, exons, and intergenic regions; (3) a transition feature that measures the frequencies of types of transitions; (4) eight boundary features; and (5) five phylogenetic features that come from data of multiple species. Nonprobabilistic features, or discriminative features, used include (1) six *gap features* which reflect insertion or deletion patterns that come from a multiple-sequence alignment; (2) three *footprint features* for each aligned sequence which give information of the position at which each sequence is aligned; and (3) nine EST features that come from EST alignments. Two methods are used to train the weights of the features: the condition maximum likelihood (CML) and the maximum expected accuracy (MEA). The CML indirectly optimizes the accuracy by maximizing the likelihood of correct segments over the training data set, whereas the MEA tries to directly optimize the accuracy of the inferred segment. The CML finds the weight $w_{\text{CML}} = \arg_w \max[\log(\Pr_w(\mathbf{y}^0, \mathbf{x}^0)]$ on the training data $(\mathbf{y}^0, \mathbf{x}^0)$ by computing the gradient using dynamic programming. The MEA optimizes the weights using an objective function A, which is defined as the expected value of the similarity function S of the distribution of segments define by the SMCRF:

$$S(\mathbf{y}, \mathbf{y}^0, \mathbf{x}^0) = \sum_{i=1}^{n} s(y_{j-1}, y_i, y_i^0, \mathbf{x}^0, i)$$
$$A(\mathbf{w}) = E_w(S(\mathbf{y}, \mathbf{y}^0, \mathbf{x}^0)). \tag{9.26}$$

and the weights $w_{\text{MEA}} = \arg_w \max[A(w)]$ are obtained by computing a gradient-based function optimizer.

The last method, CONTRAST [65], is another de novo gene prediction method that uses the CRF model. CONTRAST is composed of local and global components. For the local components, boundaries of coding regions are learned using SVMs. The global components are composed of the gene structure that integrates the boundary information learned by SVM with additional features from multiple-alignment results. The gene structure of CONTRAST consists of one intergenic node, three starting-exon nodes, three single-exon nodes, three internal exon nodes, three terminal exon nodes, and three intron nodes. The constraints are posed with directed edges between the nodes. CONTRAST uses nonprobabilistic binary features that are categorized as (1) label transition features; (2) sequence-based features including features based on target sequence, a multiple alignment, and EST alignments; and (3) coding region boundary features that come from outputs of the SVM classifier using alignment information. The weights of the features are optimized using a gradient-based optimization algorithm and the RPROP algorithm [69]. The score of a labeling is computed as $\mathbf{w} \cdot \mathbf{F}(\mathbf{x}, \mathbf{y})$ where $\mathbf{F}(\mathbf{x}, \mathbf{y}) = \sum f(y_{i-1}, y_i, i, \mathbf{x})$. CONTRAST takes a slightly different approach of training the weights and predicting the labels as compared with traditional CRF models. In prediction, selected labels maximize the weighted difference between the expected number of true-positive and false-positive coding region boundary predictions, which can be computed using the forward and backward algorithms. The weight learning is done by optimizing the expected boundary accuracy in the training set, which is done using the gradient–based optimization algorithm.

9.3.2 Protein Function Prediction Using Markov Random Fields

The large growth in high-throughput genomic and proteomic data has spurred the need for advanced computational function prediction methods that can elucidate protein function by using this system-level information [70–74]. In this section we describe the computational methods that provide insight into functional annotations of proteins using the MRF.

Traditionally protein function is predicted using sequence information only [3, 75, 76]. In recent years high-throughput experimental technologies have provided us with rich information sources, such as protein–protein interaction (PPI) data, microarray expression data, and synthetic lethality of genes. These new information sources provide the context of the whole genome to study the function of proteins. Using these various features, machine learning approaches have been used to infer the function of new proteins. Deng et al. [77, 78] first proposed the use of the MRF for modeling the probability that a protein has a certain function by capturing the local dependency of protein function on its neighbors in the PPI network. They make use of the "guilt-by-association" rule in the neighborhood of a protein and apply the local Markov property that the function of a protein is independent of all the other proteins given its neighbors in the PPI network.

9.3.2.1 Markov Random Field for Function Prediction Deng et al. [78] defined a Gibbs distribution over the PPI network of yeast, which was obtained from the Munich Information Center for Protein Sequences (MIPS) database [79]. They considered only one annotation category at a time and provided the probability that an unknown protein in the network has that function of interest F using the MRF. Given a PPI network with N proteins as nodes, out of which $P_1, \ldots, P_n$ are the n unannotated proteins and $P_{n+1}, \ldots, P_{n+m}$ are the m proteins whose annotation is known. The random variable X_i will be 1 if the ith protein has the annotation category F, else it will be zero. Thus, we obtain a vector X consisting of labeling $X_1 = \lambda_1, \ldots, X_n = \lambda_n$ and $X_{n+1} = \mu_{n+1}, \ldots, X_{n+m} = \mu_{n+m}$. Figure 9.8 shows the structure of the MRF. Let π be the fraction of proteins having annotation F. Let O_{ij} be 1 if there is interaction between proteins i and j, otherwise it is 0, and Nei(i) be the proteins interacting directly with P_i in the PPI network. Since interacting proteins are more likely to have the same function as noninteracting ones, they define the potential function $U(x)$ as

$$U(x) = -\alpha N_1 - \beta N_{10} - \gamma N_{11} - N_{00} \tag{9.27}$$

where N_1 is the number of nodes having functional annotation F, N_{11} is the number of interacting pairs where both partners have annotation F, N_{10} is the number of interacting pairs in the network where one interacting partner has F and the other does not have F, N_{00} is the number of interacting pairs in the network where both the interacting partners do not have F as their annotation and α is defined as $\log[\pi/(1-\pi)]$. With the general MRF theory we can write the global Gibbs distribution [Equation (9.28)] using the potential function $U(x)$ where $Z(\Theta)$ is the partition function and $\Theta = \{\alpha, \beta, \gamma\}$:

$$\Pr(X|\theta) = \frac{1}{Z(\theta)} \exp[-U(x)] \tag{9.28}$$

To predict the unknown labels $(X_1, \ldots, X_n)$, we need samples from the posterior distribution $\Pr(X_1, \ldots, X_n | X_{n+1}, \ldots, X_{n+m})$. These samples can be obtained by applying

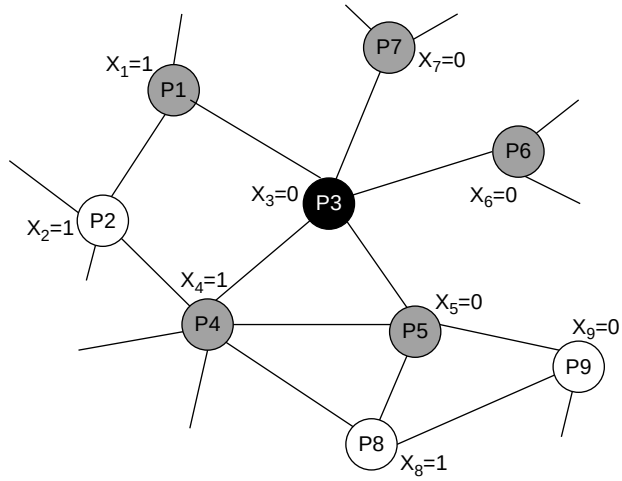

FIGURE 9.8 Markov random field based on PPI network. Shown is a PPI network with nodes as proteins and edges showing interactions between them. The MRF structure used by Deng et al. is based on the random variable X corresponding to each protein, which takes values one or zero depending on whether the protein has the annotation under consideration or not. The network also contains nodes whose X_i value is not known and thus will be predicted using the MRF framework. The node P_3 is shown in black and its neighbors, for example, Nei(P_3), are shown in gray. The edge P_1P_2 is counted as N_{11}, the edge P_1P_3 is counted as N_{10}, and the edge P_3P_5 is counted as N_{00}.

the Gibbs sampling strategy where the samples from the conditional distribution in Equation (9.29) approximate the samples from the joint distribution:

$$\begin{aligned}\Pr(X_i = 1|X_{[-i]}, \theta) &= \frac{\Pr\left(\left(X_i = 1, X_{[-i]}\right)|\theta\right)}{\Pr\left(\left(X_i = 1, X_{[-i]}\right)|\theta\right) + \Pr\left(\left(X_i = 0, X_{[-i]}\right)|\theta\right)} \\ &= \frac{\exp\left[\alpha + (\beta - 1)\, M_0^{(i)} + (\gamma - \beta)\, M_1^{(i)}\right]}{1 + \exp\left[\alpha + (\beta - 1)\, M_0^{(i)} + (\gamma - \beta)\, M_1^{(i)}\right]} \end{aligned} \tag{9.29}$$

Here $X_{[-i]} = (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_{n+m})$ and $M_{(i)}^0$, $M_{(i)}^1$ are the number of interacting partners of protein Pr_i having label 0 and 1, respectively. Assuming that the parameters Θ are obtained using the method described in the next part, it first sets initial X_i labels using the probability π. The labels are then updated repeatedly from the ones in the previous stage for a burn-in period using Equation (9.29). Then, according to Gibbs, sampling the labels X_i obtained in the lag period approximates the probability of observing function F in protein P_i.

Since the partition function Z in Equation (9.28) is the function of Θ, it is difficult to use maximum-likelihood estimation. Hence they used the quasi-likelihood method to estimate the parameter Θ. This method is based on the standard linear logistic regression model and treats all observations independently. They obtained the logistic model [Equation (9.30)]

by simplifying Equation (9.29) and the parameters were estimated using the subnetwork of known proteins:

$$\log \frac{\Pr\big(X_i = 1 | X_{[-i]}, \theta\big)}{1.0 - \Pr\big(X_i = 1 | X_{[-i]}, \theta\big)} = \alpha + (\beta - 1)\, M_0^{(i)} + (\gamma - \beta)\, M_1^{(i)} \tag{9.30}$$

Overall the method proposed by Deng et al. is novel because it applies the Markov property of the local functional dependence of the protein on its neighborhood to the PPI networks. They compared the performance of this method with the neighbor counting and the chi-square method in terms of receiver operating characteristic (ROC) curves. It was observed that for a given specificity their method was more sensitive than others. Also, they have observed that the sensitivity of the method increased when the unknown proteins have more interacting partners. Even though the method improves over the existing network-based prediction methods, there are some limitations, for example, the method treats the functions independently of each other and processes one function at a time. There are relationships across the functional categories of the Gene Ontology [80] used here that can change the probabilities of a particular protein having annotations of interest.

In the next example, Letovsky and Kasif [81] use the MRF model based on the same idea as Deng et al. that the graph neighbors are more likely to share the same annotations as opposed to the nonneighbors in the graph. They use the binomial model to describe the neighborhood function to quantify the probability of a node having a particular annotation given the annotations of its neighbors. Similar to Deng et al.'s method, this method also considers a separate MRF for each annotation label, and term dependencies are not explicitly considered. For each protein P_i a variable $L_{i,t}$ is defined which is 1 or 0 depending on whether protein P_i has label t or not. Using the Bayes rule [Equation (9.31)], they computed the label probability $\Pr(L_{i,t} = 1 | N_i, K_{i,t})$ that protein i has label t given that it has N_i neighbors and $K_{i,t}$ out of those neighbors are annotated with the label t. Subscripts are excluded in later descriptions of this equation:

$$\Pr(L_{i,t} | N_i, K_{i,t}) = \frac{\Pr(K_{i,t} | L_{i,t}, N_i) \cdot \Pr(L_{i,t})}{\Pr(K_{i,t} | N_i)} \tag{9.31}$$

where $f = \Pr(L)$ is computed as the frequency of term t in the network and $\bar{f}$ is given by $1 - f$. The main distinction of this model as compared with the previous method by Deng et al. [Equation (9.29)] is that it computes $\Pr(K|L, N)$ based on the binomial distribution $\mathrm{Bin}(N, K, \Pr_1)$. The idea behind this is that if labels are randomly assigned they will follow a binomial distribution where $\Pr_1$ is the probability that an interacting partner is labeled with t given that the protein is labeled with t. Similarly, $\Pr(K|L, N)$ is the probability of observing the given number of neighbors with label t given that the protein does not have label t. It is given by $\mathrm{Bin}(N, K, \Pr_0)$ where $\Pr_0$ is the probability that an interacting partner is labeled with t given that the protein is not labeled with t. Thus we have $P(K|N) = f \cdot P(K|L, N) + \bar{f} \cdot P(K|\bar{L}, N)$, which in turn simplifies Equation (9.31) and gives the probability of protein i having label t as shown in the equation.

$$\Pr(L|N, K) = \frac{\lambda}{1 + \lambda} \quad \text{where} \quad \lambda = \frac{f \cdot \Pr_1^K \cdot \Pr_1^{N-k}}{\bar{f} \cdot \Pr_0^K \cdot \Pr_0^{N-k}} \tag{9.32}$$

With the MRF framework based on the neighborhood conditional probability function defined in Equation (9.32), Letovsky and Kasif [81] used heuristic belief propagation to

label unannotated proteins. Initially the proteins are assigned a label based on f; then in the first step using Equation (9.32) label probabilities are estimated for unlabeled nodes. Then in the second step the adjusted probabilities in the first step are used, and to avoid invalid self-reinforcement, they apply a threshold of 0.8 to select and label nodes. Then these two steps are repeated until no more labels can be assigned. The GRID database [82] has been used for PPI data, and SGD [80] yeast Gene Ontology (GO) annotations have been used as labels. The data set consisted of 4692 labeled and 2573 unlabeled proteins. With a specific jackknife procedure they have observed that the method has 98.6% precision and 21% recall when a threshold of 0.8 was used for probability.

Deng et al. [78] did not consider unannotated proteins for which the label is unknown when training the regression model for parameter estimation. Instead, Kourmpetis et al. [83] have used the same model and applied the adaptive Markov chain Monte Carlo (MCMC) to draw samples from the joint posterior of X, Θ. They have shown that, when compared for 90 GO terms, their method gives area-under-the curve (AUC) value 0.8195 as compared with Deng et al. (0.7578) [78] and Letovsky et al. (0.7867) [81]. Kourmpetis et al. used the MRF model with the same potential function [83] as Deng and colleagues and initialized the parameters Θ and labels X using quasi-likelihood estimation and Gibbs sampling, respectively. Then, in the given iteration t the parameters are updated using a differential evolution Markov chain, a type of adaptive MCMC, conditioned on current labels, and the labels X^t conditioned on the current parameters Θ^t for unannotated nodes are updated by using Gibbs sampling. They repeated this procedure until convergence. When comparing the methods they mentioned that both Deng et al. [78] and Kourmpetis et al. [83] could estimate the intercept parameter α well, but the interaction parameters $\beta - 1$ and $\gamma - \beta$ in Equation (9.30) were better estimated by Kourmpetis et al. and led to the better performance.

9.3.2.2 Integrating Multiple Data Sources

After developing the MRF model based on PPI data, Deng et al. have further developed an integrated model that can incorporate information from PPI networks, expression profiles of genes, protein complex data, and domain information by weighing different data sources to compute the posterior probability that a protein has a particular function [84]. Here we will describe how the potential function described in Equation (9.27) is modified to take multiple data sources into account. The prior probability π_i that a protein P_i has a function of interest (i.e., $X_i = 1$) is different for each protein and is computed based on the protein complex data shown in Equation (9.33). If the protein belongs to multiple complexes, the maximum prior from any complex is used:

$$\pi_i = \Pr(X_i = 1|\text{Complex}) = \frac{\text{Proteins having the function within complex}}{\text{Known proteins within complex}} \tag{9.33}$$

Now to integrate the pairwise associations between proteins obtained from multiple sources, they converted each association into a network Net_i. For the expression data they connected proteins having expression correlation above a fixed threshold, and for a genetic interaction network they connected interacting genes based on mutation analysis data. Assuming that there are L such networks, they write the network parameters from Equation (9.27) for each network l as $(N_{10}^{(l)}, N_{11}^{(l)}, N_{00}^{(l)})$. The potential function after incorporating multiple data sources is shown in Equation (9.34) with α_i defined as $\log[\pi_i/(1 - \pi_i)]$ and parameters

$\beta_l, \gamma_l, 1 \leq l \leq L$. The joint probability distribution over all networks is given by Equation (9.35).

$$U(x) = -\sum_{i=1}^{n+m} x_i \alpha_i - \sum_{l=1}^{L} \left(\beta_l N_{10}^{(l)} + \gamma_l N_{11}^{(l)} + N_{00}^{(l)} \right) \tag{9.34}$$

$$\Pr\{\text{labeling, networks}\} = \frac{1}{Z(\theta)} \exp[-U(x)] \tag{9.35}$$

To include the dependency of the protein's function on the domain composition (a protein can consist of multiple functional regions called domains), they considered a set of domains $D_1, \ldots, D_M$ for each protein. Now P_i has domain composition d_i given by $(d_{i1}, d_{i2}, \ldots, d_{iM})$, where d_{ij} is 1 if the domain j is present in the protein P_i and 0 otherwise. Probabilities $\Pr_{1m}$ and $\Pr_{0m}$ are conditional on $d_m = 1$ (i.e., the protein has the domain d_m) given that the protein has or has not the given function, respectively. They considered all domains to be independent and gave the probability of the protein having the domain structure d given the function of interest as shown in Equation (9.36). Similarly, the probability that the protein has the domain structure d given it does not have the function is shown in Equation (9.37).

$$\Pr_1(d) = \prod_{m=1}^{M} \Pr_{1m}^{dm} (1 - Pr_{1m})^{1-dm} \tag{9.36}$$

$$\Pr_0(d) = \prod_{m=1}^{M} \Pr_{0m}^{dm} (1 - Pr_{0m})^{1-dm} \tag{9.37}$$

where M is the total number of domains in consideration. Based on the given label assignment, the probability of domain features is obtained as per Equation (9.38), which is used to modify the joint probability distribution shown in Equation (9.39) to describe the MRF based on multiple data sources:

$$\Pr\{\text{domain features}|\text{labeling}\} = \prod_{i:Xi=1} \Pr_1(d_i) \cdot \prod_{i:Xi=0} \Pr_0(d_i) \tag{9.38}$$

$$\begin{aligned} \Pr\{\text{labeling networks domain features}\} &= \Pr\{\text{labeling networks}\} \\ &\quad \cdot \Pr\{\text{domain features}|\text{labeling}\} \end{aligned} \tag{9.39}$$

In Equation (9.38), X_i is 1 or 0 depending on whether the protein P_i has the function of interest or not. Equation (9.38) expresses the probability of the domain features of all the proteins given the presence or absence of the function of interest. The first term on the right-hand side of Equation (9.39) comes from the MRF [Equation (9.35) and the second term is Equation (9.38)].

To estimate the parameters Θ [all α's, β's, and γ's in Equation (9.34)] they used the quasi-likelihood method, the same as the one described before in this section, and Gibbs sampling was used to assign functional labels to unknown proteins. To gauge the improvement in prediction accuracy, they computed the precision and recall curve of the method using just the PPI data and also with different data sources integrated. For a precision value of 57%, using only PPI data obtained the recall of 57%, which was increased to 87% when all the data sources were used together.

9.3.3 Application to Protein Tertiary Structure Prediction

Protein tertiary structure prediction remains one of the most challenging tasks in bioinformatics. Graphical models have not been applied much in this area, but there are a few of notable works. We discuss three applications of the MRF and CRF below.

9.3.3.1 Side-Chain Prediction and Free-Energy Estimation The native structure of a protein has the minimum free energy among the alternative conformations, which is defined as $G = E - TS$, where E represents the enthalpy of the system, T represents the temperature, and S denotes the entropy of the system. Thus, the aim of structure prediction of a protein is to find the structure for the protein sequence that has the minimum free energy. However, computation of the entropy is a costly process mainly because the entropy calculation requires exploring a large number of alternative conformations of the protein that make the solution infeasible.

Kamisetty et al. [85] applied the MRF to compute the free energy of protein side-chain conformations. A protein structure is represented as a set of three-dimensional atomic coordinates of the backbone atoms X_b and side-chain conformations X_s, $X = \{X_b, X_s\}$. The distribution of the conformations can be expressed as the probability of the side-chain conformations (rotamers) with respect to the backbone structure:

$$\Pr(X = x|\Theta) = \Pr(X_b = x_b) \Pr(X_s = x_s|X_b, \Theta) \tag{9.40}$$

When the backbone structure X_b is fixed, we only take the latter term into account. Assume Θ represents all the model parameters. The left panel in Figure 9.9 shows the MRF for a part of a protein structure where atoms are connected by edges if they are closer than a certain predefined distance whose actual value can vary depending on the application. When the backbone structure is fixed, the MRF represents the probability distribution of the side-chain rotamers given the backbone. To compute the probability of a particular structure, a factor graph was defined to express the relationship between different atoms. The factor graph expresses the functional form of the probability distribution (Figure 9.9). Interactions are

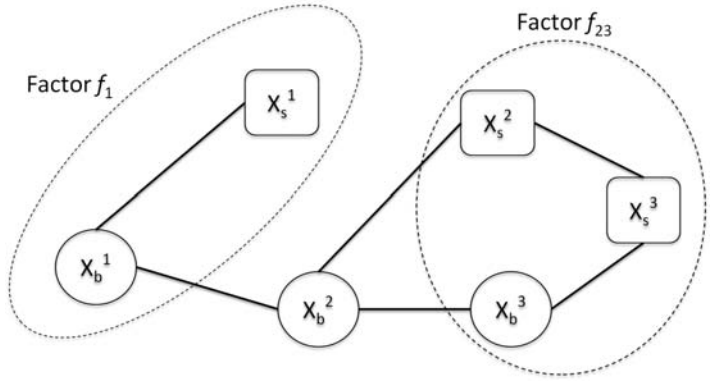

FIGURE 9.9 A factor graph from a backbone and side-chain representation is created by grouping pairwise interactions. The conjunction of all the interactions is equivalent to the functional form of the probability distribution.

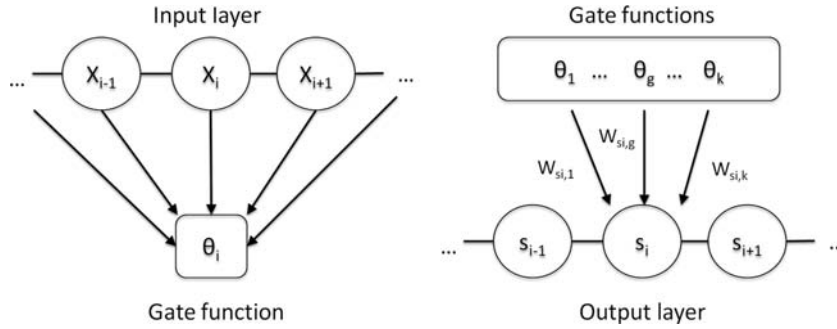

FIGURE 9.10 Conditional neural field for protein tertiary-structure prediction. The input layer x represents the sequence profile and the predicted secondary structure of q query protein and the output layer codes the angle distribution of each position.

restricted only to pairwise interactions (side chain to side chain, backbone to backbone, and side chain to backbone). In the factor graph those interactions are expressed as a series of local relationships that allow the translation from a standard backbone and side-chain representation (see Figure 9.10).

For example, simpler factors like f_1 can represent the relation between both backbone and side-chain atoms labeled x_1 (ψ_2 terms will be explained in the following paragraph). More complex relations such as f_{23} can denote backbone–backbone and side chain–side chain interactions between x_2 and x_3 (ψ_1 and ψ_3 terms, explained next). Each factor encodes the relationship between side-chain atoms by computing the pairwise potentials functions for each type of interaction:

$$\begin{aligned}\psi_1\left(X_s^{i_p}, X_s^{j_q}\right) &= \exp\left(-E(x_s^{i_p}, x_s^{j_q})/k_BT\right)\\ \psi_2\left(X_s^{i_p}, X_b^{j}\right) &= \exp\left(-E(x_s^{i_p}, x_b^{j})/k_BT\right)\\ \psi_3\left(X_b^{i}, X_b^{j}\right) &= 1\end{aligned} \tag{9.41}$$

All atoms in the model are numbered. Let i and j denote the number assigned to two atoms that have some form of interaction either directly through the backbone or by side-chain interactions and i_p and j_q denote rotamers p and q of residues i and j. The first equation expresses the potential function of two rotamers p in residue i and q in residue j while the second equation specifies the potential function of the rotamer p of the residue i and the backbone residue j. The third function is set to 1 because we assume that the backbone is fixed. These equations model the probability according to the Boltzmann distribution, where E defines the interaction energy between two elements using a linear approximation to the repulsive van der Waals force. Also, T is temperature and k_B is the Boltzmann constant. In terms of potential functions, the probability of a side chain given a backbone can be expressed as

$$\Pr(X_s|X_b) = \frac{1}{Z}\prod_{c\in C(G)}\psi_c(x_1, x_2) \tag{9.42}$$

Using the factors group, Equation (9.42) can be restated as

$$\Pr(x_s) = \frac{1}{Z} \prod_{f_a \in F} f_a\left(x_s^a\right) \tag{9.43}$$

The MRF model can be used to estimate the probability of a particular side-chain conformation. In addition, it can also be used to estimate the free-energy. As described before, the complexity of the free-energy computation comes from the large number of possible states. The way to solve this problem is to break up the factor graph into region R, which contains several factors f_R and variables x_R that can be used to compute the free energy of each region via estimates of marginal probabilities. Once the region-based estimates are computed, they can be added to produce an overall estimation. The model has been used to calculate entropy estimates that can help determine native structures from incorrect ones. The authors tested their model using 48 immunoglobin-related data sets, each containing an average of 35 decoys per data set. The native structure was ranked at the top in 42 of 48 data sets.

9.3.3.2 CRF Model for Protein Threading The threading is a protein structure prediction method aimed at identifying protein structures that fit a query protein sequence from a database of protein structure templates [9, 10]. It employs a scoring function to evaluate fitness between the query sequence and a template structure which typically combines a scoring term for assessing protein sequence similarity and several terms for scoring amino acid propensity for structural environments in the template, such as the secondary structure or exposure/burial status. Using the scoring function, the query sequence is aligned with all templates in the database which are then sorted by score to identify the most well fitting template for the sequence. This problem can be modeled using the CRF [86].

In an alignment of the query sequence and a template structure, each alignment position will be assigned a label that represents a state from the set $X = \{M, I_s, I_t\}$, where M indicates that the position aligns a residue in the sequence with one in the template (matching state), I_s indicates that an insertion event occurs in the target sequence, and I_t represents an insertion in the template. The scoring function evaluates states in the alignment. For a target s and a template t, an alignment $a = \{a_1, a_2, \ldots, a_L\}$ denotes a particular assignment of labels, where each $a_i \in X$. The conditional probability of alignment a given the target and the template protein is defined as

$$\Pr_\Theta(a|s,t) = \frac{\exp\left(\sum_{i=1}^{L} F(a,s,t,i)\right)}{Z(s,t)} \tag{9.44}$$

where $\Theta = (\lambda_1, \lambda_2, \ldots, \lambda_p)$ are the model parameters and Z is a normalization factor, taking into account all possible alignments over s and t:

$$Z(s,t) = \sum_a \exp\left(\sum_{i=1}^{L_a} F(a,s,t,i)\right) \tag{9.45}$$

where F stands for the combination of features at a particular position i in the alignment. Its functional form is as follows:

$$F(a,s,t,i) = \sum_k \lambda_k e_k(a_{i-1}, a_i, s, t) + \sum_l \lambda_l v_l(a_i, s, t) \tag{9.46}$$

Both functions e and v represent feature functions that model two types of relations. The feature function e models an edge feature, the state transition between $i-1$ and i, which depends on the features at these two positions. The second feature function, v, a label feature, models the relation between the state assigned to i and the features of the residue at the position. Note that both feature functions can be nonlinear, making the model richer in terms of the complexity of relations that can be modeled. In the CRF by Peng and Xu, predicted solvent accessibility, PSIPRED secondary structures, and PSI-BLAST sequence profiles were used as feature terms. The model is trained using the gradient tree boosting technique. Instead of visualizing function F as a linear combination of conventional terms, we can think about F being a linear combination of regression trees. The main advantage of this approach is that more complex features can be introduced and only the important ones emerge as part of the result. In their study, they chose 30 protein pairs from the PDB for training and 40 for validation, with an average size of 200 residues and sequence identity of less than 30% between them. They showed that their alignment accuracy improves with respect to existing methods, CONTRAlign and SP3/SP5, by at least 5%, being able to correctly align around 75% of the residues exactly and about 90% if a four-offset success criterion was used. They were also able to improve or get similar results in fold recognition, with respect to existing methods like RAPTOR and HHpred. The improvement is not as significant in some cases, but they show considerable increase at the superfamily level (around 10% of correctly identified residues).

9.3.3.3 Ab Initio Protein Structure Prediction

Xu et al. further developed an ab initio protein structure prediction method that employs CRF combined with the neural network [87]. The proposed framework was named the conditional neural field. The proposed method expresses the main-chain conformation as a distribution of angles at each residue position and represents nonlinear relations between features and the main-chain angles using the neural network.

A tertiary structure of the main chain of a protein can be uniquely determined by specifying a (hinge) angle θ at each residue at position i which is formed by positions $i-1$, i, and $i+1$ and τ, a pseudo–dihedral angle calculated from positions $i-2$, $i-1$, i, and $i+1$. The conformations of residues in a protein can be expressed as a probability distribution of those angles. The probability distribution of the unit vector of each pair of angles (θ, τ) for a residue position can be modeled using the Fisher–Bingham (FB5) distribution. The probability density function FB5 is given by

$$f(u) = \frac{1}{c(\kappa, \beta)} \exp\left\{\kappa\gamma_1 \cdot u + \beta\left[(\gamma_2 \cdot u)^2 - (\gamma_3 \cdot u)^2\right]\right\} \tag{9.47}$$

where u is a unit vector variable of the angles, the function c is a normalizing constant, κ determines the concentration of the angle distribution, and β is the ellipticity of the equal-probability contours. The γ values define the mean direction and the major axes. Using Equation (9.47), a protein main-chain structure can be modeled as a series of FB5 distributions, each determining likely directions that can be taken in order to place the next backbone residue. In order to limit the number of possible FB5 distributions that can be used at each position, 100 representative distributions were selected from a nonredundant protein set they analyzed. Once 1 out of the 100 distributions is assigned to a particular residue, concrete values of θ and τ can be sampled.

The FB5 distribution of angles was considered as a "label" and the probability of having each distribution for each position in a query protein sequence was estimated using the CRF. Two features were considered in the potential function of the CRF, namely, PSI-BLAST sequence profile and predicted secondary structure. In the conditional neural field, a hidden layer of nonlinear combinations was introduced (Figure 9.10). The hidden layers use a gate function $G_\theta(x) = 1/[1 + \exp(-\theta^T x)]$, with θ the parameter vector and x the feature vector.

The probability of a label assignment S (of FB5 distributions) given a sequence profile M and its secondary structure X was calculated as

$$\Pr_\Lambda(S|M, X) = \frac{\exp\left[\sum_{i=1}^{N} F(S, M, X, i)\right]}{Z(M, X)} \tag{9.48}$$

where $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_p\}$ are model parameters and the function Z is a normalization factor. The function F consists of first- and second-order feature functions (e_1, e_2) that describe the neighbor dependencies as well as a label feature function v that represents a linear combination of K gate functions with parameters w and f. The concrete form F for the features at position i is given by

$$F(S, M, X, i) = e_1(s_{i-1}, s_i) + e_2(s_{i-1}, s_i, s_{i+1}) + \sum_{j=i-w}^{i+w} v(s_{i-1}, s_i, M_j, X_j) \tag{9.49}$$

where

$$v(s_{i-1}, s_i, X, M) = \sum_{g=1}^{K} w_{s_{i-1}, s_i, g} G_{\theta_g}(f(X, M, i)) \tag{9.50}$$

This represents a linear combination of a series of gate functions G.

The benchmark study was performed on a set of about 3000 nonredundant proteins. Although they did not compare their results against other existing methods, they reported that there was an improvement with respect to their previous CRF-only method [88], showing that the introduction of the neural field layer contributed to better predictions. Additionally, the authors participated in the Critical Assessment of techniques for protein Structure Prediction in 2010 (CASP8) [89] to test their performance at the blind prediction competition. Their team was ranked notably high among the participants in CASP8. There was an average improvement of 10% in terms of TM-score/GDT-TS as compared with the CRF-only method.

9.4 SUMMARY

In this chapter, we introduced graphical models focusing on the Markov random field and the conditional random field. The latter half of this chapter was used to describe the recent applications of the graphical models in three important problems in bioinformatics, gene finding, protein function prediction, and protein structure prediction. We expect to see an increasing number of applications in coming years, especially in the protein structure prediction, since this area is not yet much explored by the graphical models, and moreover, the current applications have already produced promising results. As more complex biological

data become available that are suitable to represent using networks, these graphical models has become more important and useful in the bioinformatics field.

ACKNOWLEDGMENTS

This work was supported in part by the National Institute of General Medical Sciences of the National Institutes of Health (R01GM075004) and the National Science Foundation (DMS0800568, EF0850009, IIS0915801).

REFERENCES

1. S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.
2. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.
3. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215:403–410, 1990.
4. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.
5. W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, 85:2444–2448, 1988.
6. R. Nussinov and A. B. Jacobson. Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc. Natl. Acad. Sci USA*, 77:6309–6313, 1980.
7. C. A. Orengo and W. R. Taylor. SSAP: Sequential structure alignment program for protein structure comparison. *Methods Enzymol.*, 266:617–635, 1996.
8. I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.*, 11:739–747, 1998.
9. H. Chen and D. Kihara. Effect of using suboptimal alignments in template-based protein structure prediction. *Proteins*, 79:315–334, 2011.
10. J. Skolnick and D. Kihara. Defrosting the frozen approximation: PROSPECTOR—A new approach to threading. *Proteins*, 42:319–331, 2001.
11. Y. Matsuo and K. Nishikawa. Protein structural similarities predicted by a sequence-structure compatibility method. *Protein Sci.*, 3:2055–2063, 1994.
12. R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, 1998.
13. K. Asai, S. Hayamizu, and K. Handa. Prediction of protein secondary structure by the hidden Markov model. *Comput. Appl. Biosci.*, 9:141–146, 1993.
14. P. Lio, N. Goldman, J. L. Thorne, and D. T. Jones III. PASSML: Combining evolutionary inference and protein secondary structure prediction. *Bioinformatics*, 14:726–733, 1998.
15. A. Krogh, B. Larsson, G. von Heijne, and E. L. Sonnhammer. Predicting transmembrane protein topology with a hidden Markov model: Application to complete genomes. *J. Mol. Biol.* 305: 567–80, 2001.
16. E. L. Sonnhammer, H. G. von, and A. Krogh. A hidden Markov model for predicting transmembrane helices in protein sequences. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 6:175–182, 1998.

17. R. Karchin, M. Cline, Y. Mandel-Gutfreund, and K. Karplus. Hidden Markov models that use predicted local structure for fold recognition: Alphabets of backbone geometry. *Proteins*, 51: 504–14, 2003.
18. A. Hildebrand, M. Remmert, A. Biegert, and J. Soding. Fast and accurate automatic structure prediction with HHpred. *Proteins*, 77 (Suppl. 9):128–132, 2009.
19. C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1995.
20. D. T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292:195–202, 1999.
21. B. Rost, P. Fariselli, and R. Casadio. Topology prediction for helical transmembrane proteins at 86% accuracy. *Protein Sci.*, 5:1704–1718, 1996.
22. P. Fariselli, F. Pazos, A. Valencia, and R. Casadio. Prediction of protein–protein interaction sites in heterocomplexes with neural networks. *Eur. J Biochem.*, 269:1356–1361, 2002.
23. A. J. Bordner and A. A. Gorin. Protein docking using surface matching and supervised machine learning. *Proteins*, 68:488–502, 2007.
24. J. Guo, H. Chen, Z. Sun, and Y. Lin. A novel method for protein secondary structure prediction using dual-layer SVM and profiles. *Proteins* 54:738–743, 2004.
25. S. Hirose, K. Shimizu, S. Kanai, Y. Kuroda, and T. Noguchi. POODLE-L: A two-level SVM prediction system for reliably predicting long disordered regions. *Bioinformatics*, 23:2046–2053, 2007.
26. P. Larranaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J. A. Lozano, R. Armananzas, G. Santafe, A. Perez, and V. Robles. Machine learning in bioinformatics. *Brief. Bioinformatics*, 7:86–112, 2006.
27. A. L. Tarca, V. J. Carey, X. W. Chen, R. Romero, and S. Draghici. Machine learning and its applications to biology. *PLoS Comput. Biol.*, 3:e116, 2007.
28. Y. Saeys, I. Inza, and P. Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23:2507–2517, 2007.
29. C. J. Needham, J. R. Bradford, A. J. Bulpitt, and D. R. Westhead. A primer on learning in Bayesian networks for computational biology. *PLoS Comput. Biol.*, 3:e129, 2007.
30. D. Che, Q. Liu, K. Rasheed, and X. Tao. Decision tree and ensemble learning algorithms with their applications in bioinformatics. *Adv. Exp. Med. Biol.*, 696:191–199, 2011.
31. N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.
32. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.
33. M. I. Jordan. *Learning in Graphical Models*. MIT Press, Cambridge, MA, 1998.
34. C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, Secaucus, NJ, 2006.
35. A. L. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and Naive Bayes. *Adv. Neural Inform. Process. Syst. (NIPS)*, 14, 2002.
36. C. Sutton and A. McCallum. An Introduction to conditional random fields. *Found. Trends Mach. Learn.*, 4:267–373, 2011.
37. L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77:257–286, 1989.
38. A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. *International Conference on Machine Learning*, San Francisco, CA, 2000, pp. 591–598.
39. J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th International Conference on Machine Learning*, San Francisco, CA, 2001, pp. 282–289.

40. F. Sha and F. Pereira. Shallow parsing with conditional random fields. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language*, Edmonton, Canada, 2003, pp. 213–220.
41. S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. Paper presented at the IEEE International Conference on Computer Vision, Nice, France, 2003, pp. 1150–1157.
42. B. Wellner, A. McCallum, F. Peng, and M. Hay. An integrated, conditional model of information extraction and coreference with application to citation matching. Paper presented at the Conference on Uncertainty in Artificial Intelligence (UAI), Arlington, Virginia, 2004, pp. 593–601.
43. S. V. N. Vishwanathan, N. N. Schraudolph, M.W. Schmidt, and P. Kevin. Accelerated training of conditional random fields with stochastic gradient methods. Paper presented at the International Conference on Machine Learning, New York, NY, 2006, pp. 969–976.
44. G. Hoefel and C. Elkan. Learning a two-stage SVM/CRF sequence classifier. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM'08)*, Napa Valley, California, October 26–30, 2008.
45. K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of Uncertainty in Artificial Inteligence*, San Francisco, CA, 1999, pp. 467–475.
46. E. P. Xing, M. I. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. Paper presented at the Conference on Uncertainty in Artificial Intelligence (UAI), San Francisco, CA, 2003, pp. 583–591.
47. C. Sutton and T. Minka. Local training and belief propagation. MSR-TR-2006-121, 1-10. Microsoft, Seattle, WA, 2006.
48. P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.
49. M. Welling and G. E. Hington. A new learning algorithm for mean field Boltzmann machines. In *Proceedings of the International Conference on Artificial Neural Networks*, Lausanne, Switzerland, 2002, pp. 351–357.
50. J. Yedidia, W. T. Freeman, and Y. Weiss. Advances in neural information processing systems. *Adv. Neural Inform. Process. Syst.*, 13:689–695, 2000.
51. C. Sutton and A. McCallum. Collective segmentation and labeling of distant entities in information extraction. UMass-TR-04-49, 1-8. University of Massachusetts, Amherst, 2004.
52. Y. Liu, J. Carbonell, P. Weigele, and V. Gopalakrishnan. Protein fold recognition using segmentation conditional random fields (SCRFs). *J. Comput. Biol.*, Barbados, 13:394–406, 2006.
53. Y. Qi, M. Szummer, and T. Minka. Bayesian conditional random fields. Paper presented at the Conference on Artificial Intelligence and Statistics, Barbados, 2005.
54. A. Torralba, W. T. MurFreeman, and K. P. Murphy. Using the forest to see the trees: A graphical model relating features, objects and scenes. *Commun. ACM*, 53:107–114, 2010.
55. X. He, R. S. Zemel, and M. A. Carreira-Perpinan. Multiscale conditional random fields for image labeling. Paper presented at IEEE International Conference on Computer Vision and Pattern Recognition, Washington DC, 2004, pp. 695–702.
56. P. Liang, M. I. Jordan, and D. Klein. Learning from measurements in exponential families. Paper presented at the International Conference on Machine Learning, Montreal, Canada, 2009, pp. 641–648.
57. Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. *Adv. Neural Inform. Process. Syst.*, 17:529–536, 2005.
58. S. Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer, London, UK, 2009.

59. M. Petrou and P. G. Sevilla. *Image Processing: Dealing with Texture*. Wiley, Hoboken, NJ, 2006.
60. A. Culotta, D. Kulp, and A. McCallum. Gene prediction with conditional random fields. UM-CS-2005-028. University of Massachusetts, Amherst, 2005.
61. C. Mathé, M.-F. Sagot, T. Schiex, and P. Rouzé. SURVEY AND SUMMARY: Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res.* 2002, 30:4103–4117.
62. D. DeCaprio, J. P. Vinson, M. D. Pearson, P. Montgomery, M. Doherty, and J. E. Galagan. Conrad: Gene prediction using conditional random fields. *Genome Res.*, 17:1389–1398, 2007.
63. A. Bernal, K. Crammer, A. Hatzigeorgiou, and F. Pereira. Global discriminative learning for higher-accuracy computational gene prediction. *PLoS Comput. Biol.*, 3:e54, 2007.
64. M. K. Doherty. Gene prediction with conditional random fields. UM-CS-2005-028. University of Massachusetts, Amherst, 2007.
65. S. S. Gross, C. B. Do, M. Sirota, and S. Batzoglou. CONTRAST: A discriminative, phylogeny-free approach to multiple informant de novo gene prediction. *Genome Biol.*, 8:R269, 2007.
66. C. Wei and M. R. Brent. Using ESTs to improve the accuracy of de novo gene prediction. *BMC Bioinformatics*, 7:327, 2006.
67. J. P. Vinson, D. DeCaprio, M. D. Pearson, S. Luoma, and J. E. Galagan. Comparative gene prediction using conditional random fields. *Adv. Neural Inform. Process. Syst.*, 19:1441–1448, 2007.
68. R. H. Brown, S. S. Gross, and M. R. Brent. Begin at the beginning: predicting genes with 5′ UTRs. *Genome Res.*, 15:742–747, 2005.
69. M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. *IEEE Int. Conf. Neural Networks*, 1:586–591, 1993.
70. R. Sharan, I. Ulitsky, and R. Shamir. Network-based prediction of protein function. *Mol. Syst. Biol.*, 3:88, 2007.
71. T. Hawkins, M. Chitale, S. Luban, and D. Kihara. PFP: Automated prediction of gene ontology functional annotations with confidence scores using protein sequence data. *Proteins*, 74:566–582, 2009.
72. X. M. Zhao, L. Chen, and K. Aihara. Protein function prediction with high-throughput data. *Amino Acids*, 35:517–530, 2008.
73. A. Vazquez, A. Flammini, A. Maritan, and A. Vespignani. Global protein function prediction from protein-protein interaction networks. *Nat. Biotechnol.*, 21:697–700, 2003.
74. D. Kihara. *Protein Function Prediction for Omics Era*. Springer, London, 2011.
75. M. Chitale, T. Hawkins, and D. Kihara. Automated prediction of protein function from sequence. In J. Bujnicki (Ed.), *Prediction of Protein Strucutre, Functions, and Interactions*. Wiley, Hoboken, NJ, 2009.
76. M. Chitale, T. Hawkins, C. Park, and D. Kihara. ESG: Extended similarity group method for automated protein function prediction. *Bioinformatics*, 25:1739–1745, 2009.
77. M. Deng, T. Chen, and F. Sun. An integrated probabilistic model for functional prediction of proteins. *J. Comput. Biol.*, 11:463–475, 2004.
78. M. Deng, K. Zhang, S. Mehta, T. Chen, and F. Sun. Prediction of protein function using protein-protein interaction data. *J. Comput. Biol.*, 10:947–960, 2003.
79. H. W. Mewes, D. Frishman, U. Guldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Munsterkotter, S. Rudd, and B. Weil. MIPS: A database for genomes and protein sequences. *Nucleic Acids Res.*, 30:31–34, 2002.
80. M. A. Harris, et al. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.*, 32:D258–D261, 2004.
81. S. Letovsky and S. Kasif. Predicting protein function from protein/protein interaction data: A probabilistic approach. *Bioinformatics*, 19 (Suppl. 1):i197–i204, 2003.

82. C. Stark, B. J. Breitkreutz, A. Chatr-Aryamontri, L. Boucher, R. Oughtred, M. S. Livstone, J. Nixon, A. K. Van, X. Wang, X. Shi, T. Reguly, J. M. Rust, A. Winter, K. Dolinski, and M. Tyers. The BioGRID Interaction Database: 2011 update. *Nucleic Acids Res.*, 39:D698–D704, 2011.

83. Y. A. Kourmpetis, A. D. van Dijk, M. C. Bink, R. C. van Ham, and C. J. ter Braak. Bayesian Markov random field analysis for protein function prediction based on network data. *PLoS ONE*, 5:e9293, 2010.

84. M. Deng, Z. Tu, F. Sun, and T. Chen. Mapping Gene Ontology to proteins based on protein-protein interaction data. *Bioinformatics*, 20:895–902, 2004.

85. H. Kamisetty, E. P. Xing, and C. J. Langmead. Free energy estimates of all-atom protein structures using generalized belief propagation. *J. Comput. Biol.*, 15:755–766, 2008.

86. J. Peng and J. Xu. Boosting protein threading accuracy. *Lect. Notes Comput. Sci.*, 5541:31, 2009.

87. F. Zhao, J. Peng, and J. Xu. Fragment-free approach to protein folding using conditional neural fields. *Bioinformatics*, 26:i310–i317, 2010.

88. F. Zhao, S. Li, B. W. Sterner, and J. Xu. Discriminative learning for protein conformation sampling. *Proteins*, 73:228–240, 2008.

89. J. Moult, K. Fidelis, A. Kryshtafovych, B. Rost, and A. Tramontano. Critical assessment of methods of protein structure prediction—Round VIII. *Proteins*, 77 (Suppl. 9):1–4, 2009.