Kernel Conditional Random Fields: Representation, Clique Selection, and Semi-Supervised Learning

John Lafferty, Yan Liu and Xiaojin Zhu

February 5, 2004
CMU-CS-04-115

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Kernel conditional random fields are introduced as a framework for discriminative modeling of graph-structured data. A representer theorem for conditional graphical models is given which shows how kernel conditional random fields arise from risk minimization procedures defined using Mercer kernels on labeled graphs. A procedure for greedily selecting cliques in the dual representation is then proposed, which allows sparse representations. By incorporating kernels and implicit feature spaces into conditional graphical models, the framework enables semi-supervised learning algorithms for structured data through the use of graph kernels. The clique selection and semi-supervised methods are demonstrated in synthetic data experiments, and are also applied to the problem of protein secondary structure prediction.

This research was partially supported by the National Science Foundation (NSF), under grants IIS-0205456 and IIS-0312814.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, or the U.S. government.

Keywords: Sequence analysis, graphical models, conditional models, kernels, semi-supervised learning, protein secondary structure

1 Introduction

Many classification problems involve the annotation of data items having multiple components, with each component requiring a classification label. Such problems are challenging because the interaction between the components can be rich and complex. In text, speech, and image processing, for example, it is often useful to label individual words, sounds, or image patches with categories to enable higher level processing; but these labels can depend on one another in a highly complex manner. For biological sequence annotation, it is desirable to annotate each amino acid in a protein with a label, with the collection of labels representing the global geometric structure of the molecule. Here the labels in principle depend on the physical characteristics of the molecule and its ambient chemical environment. In each case, classification tasks naturally arise which clearly violate the assumption of independent and identically distributed instances that is made in the majority of classification procedures in statistics and machine learning. It is therefore of central importance to extend recent advances in classification theory and practice to structured, non-independent data classification problems.

Conditional random fields (Lafferty et al., 2001) have been proposed as an approach to modeling the interactions between labels in such problems using the tools of graphical models. A conditional random field (CRF) is a model that assigns a joint probability distribution over labels conditional on the input, where the distribution respects the independence relations encoded in a graph. In general, the labels are not assumed to be independent, nor are the observations conditionally independent given the labels, as is assumed in generative models such as hidden Markov models. The CRF framework has already been used to obtain promising results in a number of domains where there is interaction between labels, including tagging, parsing and information extraction in natural language processing (Collins, 2002, Sha and Pereira, 2003, Pinto et al., 2003) and the modeling of spatial dependencies in image processing (Kumar and Hebert, 2003). In all of this work, however, conditional random fields are based on explicit feature representations.

We present an extension of conditional random fields that permits the use of implicit features spaces through Mercer kernels. Such an extension is motivated by the significant body of recent work that has shown kernel methods to be extremely effective in a wide variety of machine learning techniques; for example, they enable the integration of multiple sources of information in a principled manner. Our introduction of Mercer kernels into conditional graphical models is also motivated by the problem of semi-supervised learning. In many domains, the collection of annotated training data is difficult and costly, as it requires the efforts of expert human annotators, while the collection of unlabeled data may be relatively easy and inexpensive. The emerging theme in recent research in semi-supervised learning is that kernel methods, in particular those based on graphical representations of unlabeled data, form a theoretically attractive and empirically promising set of techniques for combining labeled and unlabeled data (Belkin and Niyogi, 2002, Chapelle et al., 2002, Smola and Kondor, 2003, Zhu et al., 2003).

In Section 2 we formalize the learning problem in terms of *graphical processes*, a form of stochastic process where random variables are indexed by graphs. A version of the classical representer theorem of Kimeldorf and Wahba (1971) is given which shows that the solution to a regularized risk minimization problem necessarily leads to a representation in terms of the kernel over the data. However, unlike the classical result, for kernel conditional random fields the dual parameters depend on all potential assignments of labels to cliques in the graph, not only the observed labels. The resulting framework is closely related to the methods proposed recently by Taskar et al. (2003); the representer theorem derived here provides additional justification for these methods.

This generalized representer theorem motivates the need for algorithms to derive sparse representations, since the full representation has parameters for each labeled clique in the graphs appearing in the training data. In Section 3 we present a greedy algorithm for selecting a small number of representative cliques. This "clique selection" algorithm parallels the "import vector selection" algorithms of kernel logistic regression (Zhu and Hastie, 2001), and the feature selection methods that have been previously proposed for conditional random fields using explicit features (McCallum, 2003).

In Section 4 we explain how kernels for semi-supervised learning can be incorporated into our framework to combine labeled data with unlabeled data. The ideas and methods are demonstrated on a synthetic data set in Section 5, where the effects of the underlying graph kernels, clique selection, and sequential modeling can be clearly seen. In Section 6 we report the results of experiments using kernel CRFs for protein secondary structure prediction. This is the task of mapping primary sequences of amino acids onto a string of secondary structure assignments, such as helix, sheet, or coil. It is widely believed that secondary structure can contribute valuable information to discerning how proteins fold in three dimensions. We compare kernel conditional random fields, estimated using clique selection, against support vector machine classifiers, with both methods using kernels derived from position-specific scoring matrices (PSI-BLAST profiles) as input features. In addition, we give results for the use of graph kernels derived from the PSI-BLAST profiles in a transductive, semi-supervised framework for estimating the kernel CRFs. The paper concludes with a brief discussion in Section 7.

2 Representation

Before proceeding with formalism, we give some intuition for what our framework is intended to capture. Our goal is to annotate structured data, where the structure is represented by a graph. Labels are to be assigned to the nodes in the graph in order to minimize some loss function, such as 0-1 error; the labels come from a small set $\mathcal{Y}$, for example, $\mathcal{Y} = \{\texttt{red}, \texttt{blue}, \texttt{green}\}$. Each vertex in the graph is associated with a feature vector $\boldsymbol{x}_v \in \mathcal{X}$. In image processing, the feature vector at a node might include a pixel intensity, as well as average pixel intensities smoothed over neighboring regions using wavelets. In protein secondary structure prediction, each node might correspond to an amino acid in the protein, and the feature vector at a node may include an amino acid histogram of all protein fragments in a database which closely match the given protein at that node.

We consider models of the joint assignment of labels to a graph. A bit more abstractly, recall that a stochastic process is a family of random variables $\xi_t \in \mathbb{R}^m$ indexed by $t \in T$; when $T = \mathbb{R}$, the index is often thought of as time. In standard kernel machines such as Gaussian processes, one considers a stochastic process indexed by feature vectors $\boldsymbol{x} \in \mathbb{R}^n$, defining a covariance kernel $E[\langle \xi_{\boldsymbol{x}}, \xi_{\boldsymbol{x}'} \rangle] = K(\boldsymbol{x}, \boldsymbol{x}')$. In our framework, we shall adopt a generalization, which might be called a *graphical process*—a stochastic process indexed by graphs together with their feature vectors, $(\mathfrak{g}, \boldsymbol{x})$. Thus, we consider random variables $\xi_{(\mathfrak{g},\boldsymbol{x})} \in \mathbb{R}^m$ together with certain covariance kernels $E[\langle \xi_{(\mathfrak{g},\boldsymbol{x})}, \xi_{(\mathfrak{g}',\boldsymbol{x}')} \rangle] = K((\mathfrak{g}, \boldsymbol{x}), (\mathfrak{g}', \boldsymbol{x}'))$ for the purpose of predicting labels on graphs.

2.1 Cliques and labeled graphs

Let $\mathfrak{G}$ denote a collection of finite graphs. For example, $\mathfrak{G}$ might be the set of finite chains, appropriate for sequence modeling, or the rectangular 2-dimensional grids, appropriate for some image processing tasks. The set of vertices of a graph $\mathfrak{g} \in \mathfrak{G}$ is denoted by $V(\mathfrak{g})$, and size of the graph is the number of vertices, denoted $|\mathfrak{g}| = |V(\mathfrak{g})|$. A *clique* is a subset of the vertices which is fully connected, with any pair of vertices joined by an edge; we denote the set of cliques in the graph by $\mathcal{C}(\mathfrak{g})$. The number of vertices in a clique is denoted by $|c|$. Similarly, we denote by $\mathcal{C}(\mathfrak{G}) = \{(\mathfrak{g}, c) \,|\, \mathfrak{g} \in \mathfrak{G},\ c \in \mathcal{C}(\mathfrak{g})\}$ the collection of cliques across varying graphs. In other words, a member of $\mathcal{C}(\mathfrak{G})$ consists of a graph and a distinguished clique of that graph. We will work with kernels that compare components of different graphs. For example, we could consider a kernel $K : \mathcal{C}(\mathfrak{G}) \times \mathcal{C}(\mathfrak{G}) \to 0$ given by $K((\mathfrak{g}, c), (\mathfrak{g}', c')) = \delta(|c|, |c'|)$.

We next consider labelings of a graph. Let $\mathcal{Y}$ be a finite set of labels; infinite $\mathcal{Y}$ is also possible in a regression framework, but we restrict to finite $\mathcal{Y}$ for simplicity. The set of $\mathcal{Y}$-labelings of a graph $\mathfrak{g}$ is denoted $\mathcal{Y}(\mathfrak{g}) = \left\{\boldsymbol{y} \,|\, \boldsymbol{y} \in \mathcal{Y}^{|\mathfrak{g}|}\right\}$, and the collection of all $\mathcal{Y}$-labeled graphs is $\mathcal{Y}(\mathfrak{G}) = \{(\mathfrak{g}, \boldsymbol{x}) \,|\, \mathfrak{g} \in \mathfrak{G},\ \boldsymbol{y} \in \mathcal{Y}(\mathfrak{g})\}$. Similarly, let $\mathcal{X}$ be an input feature space; for example, $\mathcal{X} = \mathbb{R}^n$. The set $\mathcal{X}(\mathfrak{g}) = \left\{\boldsymbol{x} \,|\, \boldsymbol{x} \in \mathcal{X}^{|\mathfrak{g}|}\right\}$ denotes the set of assignments of a feature vector to each vertex of the graph $\mathfrak{g}$; $\mathcal{X}(\mathfrak{G}) = \{(\mathfrak{g}, \boldsymbol{x}) \,|\, \mathfrak{g} \in \mathfrak{G},\ \boldsymbol{x} \in \mathcal{X}(\mathfrak{g})\}$ is the collection of all such annotated graphs. Finally, let $\mathcal{Y}_{\mathcal{C}}(\mathfrak{g}) = \left\{(c, \boldsymbol{y}_c) \,|\, c \in \mathcal{C}(\mathfrak{g}),\ \boldsymbol{y}_c \in \mathcal{Y}^{|c|}\right\}$ be the set of $\mathcal{Y}$-labeled cliques in a graph. As above, we similarly define $\mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{g}) = \{(\boldsymbol{x}, c, \boldsymbol{y}_c) \,|\, \boldsymbol{x} \in \mathcal{X}(\mathfrak{g}),\ (c, \boldsymbol{y}_c) \in \mathcal{Y}_{\mathcal{C}}(\mathfrak{g})\}$ and $\mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{G}) = \{(\mathfrak{g}, \boldsymbol{x}, c, \boldsymbol{y}_c) \,|\, (\boldsymbol{x}, c, \boldsymbol{y}_c) \in \mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{g})\}$.

2.2 Representer Theorem

The prediction task for conditional graphical models is to learn a function $h : \mathcal{X}(\mathfrak{G}) \to \mathcal{Y}(\mathfrak{G})$ where $h(\mathfrak{g}, \boldsymbol{x}) \in \mathcal{Y}(\mathfrak{g})$ is a labeling of $\mathfrak{g}$, with the goal of minimizing a suitably defined loss function. The classifier $h = h_n$ is chosen based on a labeled sample $\left\{(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\right\}_{i=1}^{n}$, with each $(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ being a labeled graph, the graph possibly changing from example to example.

To limit the complexity of the hypothesis, we will assume that it is determined completely by a function $f : \mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{G}) \to \mathbb{R}$. Let $f(\mathfrak{g}, \boldsymbol{x})$ denote the collection of values $\{f(\mathfrak{g}, \boldsymbol{x}, c, \boldsymbol{y}_c)\}$, with $c \in \mathcal{C}(\mathfrak{g})$ varying over the cliques of $\mathfrak{g}$ and $\boldsymbol{y}_c \in \mathcal{Y}^{|c|}$ varying over all possible labelings of that clique. We assume that a loss function $\phi(\boldsymbol{y}, f(\mathfrak{g}, \boldsymbol{x}))$ is given. As an important example, and the loss function used in this paper, consider the *negative log loss*

$$\phi(\boldsymbol{y}, f(\mathfrak{g}, \boldsymbol{x})) \;=\; -\sum_{c \in \mathcal{C}(\mathfrak{g})} f_c(\boldsymbol{x}, \boldsymbol{y}_c) + \log \sum_{\boldsymbol{y}' \in \mathcal{Y}(\mathfrak{g})} \exp\left(\sum_{c \in \mathcal{C}(\mathfrak{g})} f_c(\boldsymbol{x}, \boldsymbol{y}'_c)\right) \tag{1}$$

where $f_c(\boldsymbol{x}, \boldsymbol{y}_c)$ is shorthand for $f(\mathfrak{g}, \boldsymbol{x}, c, \boldsymbol{y}_c)$. The negative log *marginal* loss could also be considered for minimizing the per-node error. The negative log loss function corresponds to a conditional random field given by

$$p(\boldsymbol{y} \,|\, \mathfrak{g}, \boldsymbol{x}) = Z^{-1}(\mathfrak{g}, \boldsymbol{x}, f)\, \exp\left(\sum_{c} f_c(\boldsymbol{x}, \boldsymbol{y}_c)\right) \tag{2}$$

We now extend the standard "representer theorem" of kernel machines (Kimeldorf and Wahba, 1971) to conditional graphical models. While this is a simple extension, we're not aware of an analogous formulation in the statistics or machine learning literature.

Let K be a Mercer kernel on $\mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{G})$; thus

$$K((\mathfrak{g}, \boldsymbol{x}, c, \boldsymbol{y}_c), (\mathfrak{g}', \boldsymbol{x}', c', \boldsymbol{y}'_{c'})) \in \mathbb{R} \tag{3}$$

for each $(\boldsymbol{x}, c, \boldsymbol{y}_c) \in \mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{g})$ and $(\boldsymbol{x}', c', \boldsymbol{y}'_{c'}) \in \mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{g}')$. Intuitively, this assigns a measure of similarity between a labeled clique in one graph and a labeled clique in a (possibly) different graph. We denote by $\mathcal{H}_K$ the associated reproducing kernel Hilbert space, and by $\|\cdot\|_K$ the associated norm on $L^2(\mathcal{X}\mathcal{Y}_{\mathcal{C}}(\mathfrak{G}))$.

Consider a regularized loss function of the form

$$R_\phi f = \sum_{i=1}^{n} \phi\left(\boldsymbol{y}^{(i)}, f(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)})\right) + \Omega\left(\|f\|_K\right) \tag{4}$$

It is important to note that the loss depends on all possible assignments $\boldsymbol{y}_c$ of labels to each clique, not just those observed in the labeled data $\boldsymbol{y}^{(i)}$.

Suppressing the dependence on the graph $\mathfrak{g}$ in the notation, let $K_c(\boldsymbol{x}, \boldsymbol{y}_c; , \cdot) = K((\mathfrak{g}, \boldsymbol{x}, c, \boldsymbol{y}_c), \cdot)$. Following the argument for the standard representer theorem, we can decompose $\mathcal{H}$ as $\mathcal{H} = \mathcal{H}_n \oplus \mathcal{H}_n^\perp$ where

$$\mathcal{H}_n = \operatorname{span}\left(\left\{K_c(\boldsymbol{x}^{(i)}, \boldsymbol{y}_c; \cdot)\right\}_{i, \boldsymbol{y}_c}\right) \tag{5}$$

Thus, we write

$$f \quad = \quad f^\perp + \sum_{i=1}^{n} \sum_{c \in \mathfrak{g}^{(i)}} \sum_{\boldsymbol{y}_c} \alpha_c^{(i)}(\boldsymbol{y}_c)\, K_c(\boldsymbol{x}^{(i)}, \boldsymbol{y}_c; \cdot) \tag{6}$$

where $f^\perp \in \mathcal{H}_n^\perp$. By the reproducing property,

$$f(\mathfrak{g}^{(j)}, \boldsymbol{x}^{(j)}, c', \boldsymbol{y}'_{c'}) \quad = \quad \sum_{i, c, \boldsymbol{y}_c} \alpha_c^{(i)}(\boldsymbol{y}_c)\, K\left((\mathfrak{g}^{(i)}, \boldsymbol{x}_i, c, \boldsymbol{y}_c), (\mathfrak{g}^{(j)}, \boldsymbol{x}^{(j)}, c', \boldsymbol{y}'_{c'})\right) \tag{7}$$

since, by definition, $f^\perp$ is orthogonal to each $K_c(\boldsymbol{x}^{(i)}, \boldsymbol{y}_c; \cdot)$. Finally, if Ω is strictly increasing, the penalty term $\Omega(\|f\|_K)$ can only be decreased by letting $f^\perp = 0$. This proves the following result.

Proposition (Representer theorem for CRFs). *Let K be a Mercer kernel on $\mathcal{X}\mathcal{Y}_\mathcal{C}(\mathfrak{G})$ with associated RKHS norm $\|\cdot\|_K$, and let $\Omega : \mathbb{R}_+ \to \mathbb{R}_+$ be strictly increasing. Then the minimizer $f^\star$ of*

$$R_\phi f = \sum_{i=1}^{n} \phi\left(\boldsymbol{y}^{(i)}, f(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)})\right) + \Omega\left(\|f\|_K\right) \tag{8}$$

if it exists, has the form

$$f^\star(\cdot) \;=\; \sum_{i=1}^{n} \sum_{c \in \mathcal{C}(\mathfrak{g}^{(i)})} \sum_{\boldsymbol{y}_c \in \mathcal{Y}^{|c|}} \alpha_c^{(i)}(\boldsymbol{y}_c)\, K_c(\boldsymbol{x}^{(i)}, \boldsymbol{y}_c; \cdot) \tag{9}$$

The key property distinguishing this result from the standard representer theorem is that the "dual parameters" $\alpha_c^{(i)}(\boldsymbol{y}_c)$ now depend on *all* assignments of labels.

2.3 Two special cases

Let $\overline{K}$ be a Mercer kernel on $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} \times \mathcal{Y}$. Thus, the kernel is defined in terms of the matrix entries $\overline{K}(\boldsymbol{z}, \boldsymbol{z}')$ where $\boldsymbol{z} = (x, y_1, y_2)$. Using $\overline{K}$ we can define a kernel on edges in $\mathcal{X}\mathcal{Y}_\mathcal{C}(\mathfrak{G})$ by $K\left((\mathfrak{g}, \boldsymbol{x}, (v_1, v_2), (y_1, y_2)), (\mathfrak{g}', \boldsymbol{x}', (v'_1, v'_2), (y'_1, y'_2))\right) = \overline{K}((\boldsymbol{x}_{v_1}, y_1, y_2), (\boldsymbol{x}'_{v'_1}, y'_1, y'_2))$. For the regularized risk minimization problem

$$\min_{f \in \mathcal{H}_K} R_\phi(\boldsymbol{x}, f, \lambda) = \min_{f \in \mathcal{H}_K} \sum_{i=1}^{n} \phi(\boldsymbol{y}^{(i)}, f(\boldsymbol{x}^{(i)})) + \lambda\, \|f\|_K \tag{10}$$

where $f \in \mathcal{H}_K$, the CRF representer theorem implies that the solution $f^\star$ has the form

$$f^\star_{(v_1,v_2)}(\boldsymbol{x}, y_1, y_2) \quad = \quad \sum_{i=1}^{n} \sum_{y,y'} \sum_{(v,v')} \alpha^{(i)}_{(v,v')}(y, y')\, \overline{K}((\boldsymbol{x}_{v_1}, y_1, y_2), (\boldsymbol{x}^{(i)}_v, y, y')) \tag{11}$$

In the special case of kernel $\overline{K}(\boldsymbol{z}, \boldsymbol{z}') = \overline{K}(x, x')\, \delta(y_1, y_1')$ it follows that

$$f^\star_{(v_1,v_2)}(\boldsymbol{x}, y_1, y_2) = \sum_{i=1}^{n} \sum_{v \in V(\mathfrak{g}^{(i)})} \alpha^{(i)}_v(y_1)\, \overline{K}(\boldsymbol{x}_{v_1}, \boldsymbol{x}^{(i)}_v) \tag{12}$$

Under the probabilistic model (2), this is simply kernel logistic regression. In the special case of $\overline{K}(\boldsymbol{z}, \boldsymbol{z}') = \overline{K}(x, x')\, \delta(y_1, y_1') + \delta(y_1, y_1')\, \delta(y_2, y_2')$ we get that

$$f^\star_{(v_1,v_2)}(\boldsymbol{x}, y_1, y_2) = \sum_{i,v} \alpha^{(i)}_v(y_1)\, \overline{K}(\boldsymbol{x}^{(i)}_v, \boldsymbol{x}_{v_1}) + \alpha(y_1, y_2) \tag{13}$$

and we recover a simple type of semiparametric CRF.

3 Clique Selection

The representer theorem shows that the minimizing function f is supported by labeled cliques over the training examples; however, this may result in an extremely large number of parameters. We therefore pursue a strategy of incrementally selecting cliques in order to greedily reduce the regularized risk. The resulting procedure is parallel to forward stepwise logistic regression, and to related methods for kernel logistic regression (Zhu and Hastie, 2001).

Our algorithm will maintain an *active set* $\mathcal{A} = \left\{(\mathfrak{g}^{(i)}, c, \boldsymbol{y}_c)\right\} \subset \mathcal{Y}_{\mathcal{C}}(\mathfrak{G})$ of labeled cliques, where the labelings are not restricted to those appearing in the training data. Each such candidate clique can be represented by a basis function $h(\cdot) = K((\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)}, c, \boldsymbol{y}_c), \cdot) \in \mathcal{H}_K$, and is assigned a parameter $\alpha_h = \alpha^{(i)}_c(\boldsymbol{y}_c)$. We work with the regularized risk

$$R_\phi(f) = \sum_i \phi\left(\boldsymbol{y}^{(i)}, f(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)})\right) + \frac{\lambda}{2}\, \|f\|^2_K \tag{14}$$

where ϕ is the log-loss of equation (1). To evaluate a candidate h, one strategy is to compute the *gain* $\sup_\alpha R_\phi(f) - R_\phi(f + \alpha h)$, and to choose the candidate h having the largest gain. This presents an apparent difficulty, since the optimal parameter α cannot be computed in closed form, and must be evaluated numerically. For sequence models this would involve forward-backward calculations for each candidate h, the cost of which is prohibitive.

As an alternative, we adopt the functional gradient descent approach, which evaluates a small change to the current function. For a given candidate h, consider adding h to the current model with small weight ε; thus $f \mapsto f + \varepsilon h$. Then $R_\phi(f + \varepsilon h) = R_\phi(f) + \varepsilon dR_\phi(f, h) + O(\varepsilon^2)$, where the functional derivative of R_ϕ at f in the direction h is computed as

$$dR_\phi(f, h) \quad = \quad E_f[h] - \widetilde{E}[h] + \lambda\, \langle f, h \rangle_K \tag{15}$$

where $\widetilde{E}[h] = \sum_i h(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ is the empirical expectation and $E_f[h] = \sum_i \sum_{\boldsymbol{y}} p(\boldsymbol{y} \,|\, \boldsymbol{x}^{(i)}, f) h(\boldsymbol{x}^{(i)}, \boldsymbol{y})$ is the model expectation conditioned on $\boldsymbol{x}$. Following our earlier notation,

$$h(\boldsymbol{x}^{(i)}, \boldsymbol{y}) = \sum_{c \in \mathcal{C}(\mathfrak{g}^{(i)})} h(\mathfrak{g}^{(i)}, \boldsymbol{x}^{(i)}, c, \boldsymbol{y}_c) \tag{16}$$

Initialize with $f = 0$, and iterate:

1. For each candidate $h \in \mathcal{H}_K$, supported by a single labeled clique, calculate the functional derivative $dR_\phi(f, h)$.
2. Select the candidate $h = \arg\max_h |dR_\phi(f, h)|$ having the largest gradient direction. Set $f \mapsto f + \alpha_h h$.
3. Estimate parameters α_f for each active f by minimizing $R_\phi(f)$.

Figure 1: Greedy Clique Selection. Labeled cliques encode basis functions h which are greedily added to the model, using a form of functional gradient descent.

is the sum over all cliques. The idea is that in directions h where the functional gradient $dR_\phi(f, h)$ is large, the model is mismatched with the labeled data; this direction should be added to the model to make a correction. This results in the greedy clique selection algorithm summarized in Figure 1.

An alternative to the greedy functional gradient descent algorithm above is to estimate parameters α_h for each candidate. When each candidate clique is a vertex, the gain can be efficiently approximated using a mean field approximation. Under this approximation, a candidate is evaluated according to the approximate gain

$$R_\phi(f) - R_\phi(f + \alpha h) \approx \sum_i \sum_v Z(f, \boldsymbol{x}^{(i)})^{-1} p(\boldsymbol{y}_v^{(i)} \,|\, \boldsymbol{x}^{(i)}, f) \exp(\alpha h(\boldsymbol{x}^{(i)}, \boldsymbol{y}_v^{(i)})) + \lambda \langle f, h \rangle \qquad (17)$$

which is a logistic approximation. We adopt this approach in our experimental results for proteins when only vertex cliques are used. In the experiments reported below for sequences, the marginal probabilities $p(\boldsymbol{y}_t = 1 \,|\, \boldsymbol{x})$ and expected counts for the state transitions are required; these are computed using the forward-backward algorithm, with log domain arithmetic to avoid underflow. A quasi-Newton method (BFGS, cubic-polynomial line search) is used to estimate the parameters in step 3.

4 Kernels for Semi-Supervised Learning

One of the emerging themes in semi-supervised learning is that graph kernels can provide a useful framework for combining labeled and unlabeled data. Here an undirected graph is defined over labeled and unlabeled data instances, and generally the assumption is that labels vary smoothly over the graph. The graph is represented by the weight matrix W. Let D be the diagonal degree matrix $D_{ii} = \sum_j w_{ij}$. The focus of the graph kernels is the eigensystem of the combinatorial Laplacian matrix $L = D - W$, or the normalized Laplacian $\tilde{L} = D^{-1/2} L D^{-1/2}$. The eigenvectors of the smaller eigenvalues vary more smoothly over the graph than those of the larger eigenvalues. In a physical analogy, imagining springs on the edges of the graph, the eigenvectors correspond to the vibrational modes of the graph and the eigenvalues are the corresponding frequencies.

To obtain smooth functions (soft labelings) over the graph, one can construct a kernel from the graph Laplacian but substituting eigenvalues λ by $r(\lambda)$, where r is a non-negative and (typically) decreasing function. This regularizes high frequency components and encourages smooth functions on the graph. This view of graph kernels, proposed by Smola and Kondor (2003), unifies within a single framework several graph kernel methods, including manifold learning (Belkin and Niyogi,

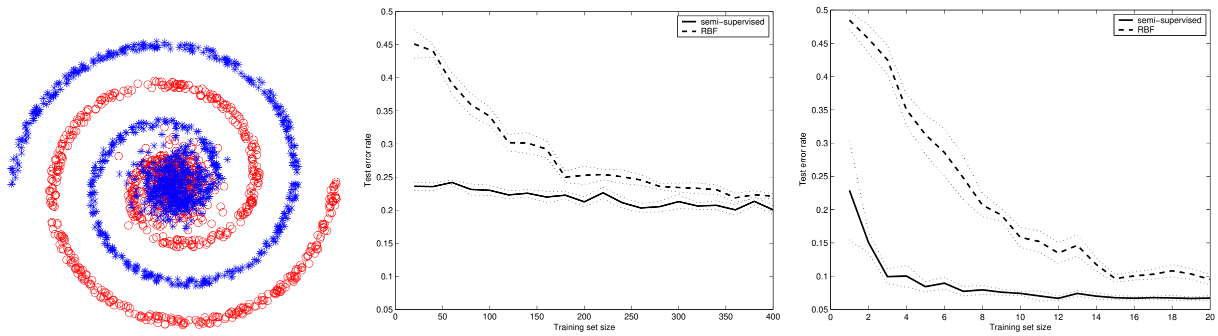

Figure 2: Left: The galaxy data is comprised of two interlocking spirals together with a "dense core" of samples from both classes. Center: Kernel logistic regression comparing two kernels, RBF and a graph kernel using the unlabeled data. Right: Kernel conditional random fields, which take into account the sequential structure of the data.

2002), cluster kernels (Chapelle et al., 2002), diffusion kernels (Kondor and Lafferty, 2002), random walks (Szummer and Jaakkola, 2001), normalized cuts (Yu and Shi, 2001), and Gaussian fields (Zhu et al., 2003). In this paper, we define semi-supervised graph kernels over the vertices. It is straightforward to plug such graph kernels into kernel CRFs, to enable semi-supervised learning for sequences and more complex graphical structures.

5 Synthetic Data Experiments

To work with a data set that will distinguish a semi-supervised graph kernel from a standard kernel, and a sequence model from a non-sequence model, we prepared a synthetic data set ("galaxy") that is a variant of spirals, see Figure 2 (left). Note the mixed dense core of data from both classes.

We sample 100 sequences of length 20 according to an HMM with two states, where each state emits instances uniformly from one of the classes. There is a 90% chance of staying in the same state, and the initial state is uniformly chosen. The idea is that under a sequence model we should be able to use the context to determine which of the classes an example from the core is truly from. However, under a non-sequence model without the context, the core region will be indistinguishable, and the dataset as a whole will have about 20% Bayes error rate. Note the choice of semi-supervised vs. standard kernels and sequence vs. non-sequence models are orthogonal; the four combinations are all tested on.

We construct the semi-supervised graph kernel by first building an unweighted 10-nearest neighbor graph. We compute the associated graph Laplacian L, and then form the kernel $K = 10\left(L + 10^{-6} I\right)^{-1}$. This corresponds to a function $r(\lambda) = \frac{1}{\lambda + 10^{-6}}$ on L's eigenvalues. The standard kernel is the radial basis function (RBF) kernel with an optimal bandwidth $\sigma = 0.35$.

Now we apply both kernels to a non-sequence model, kernel logistic regression; see Figure 2 (center). The sequence structure is ignored. Ten random trials were performed with each training set size, which ranges from 20 to 400 points. The error intervals are one standard error. As expected, when the labeled set size is small, the RBF kernel results in significantly larger test error than the graph kernel. Furthermore, both kernels saturate at the 20% Bayes error rate.

Next we apply both kernels to a KCRF sequence model. Experimental results are shown in Figure 2 (right). Note the x-axis is the number of training sequences: Since each sequence has 20 instances, the range is the same as Figure 2 (center). The kernel CRF is capable of getting

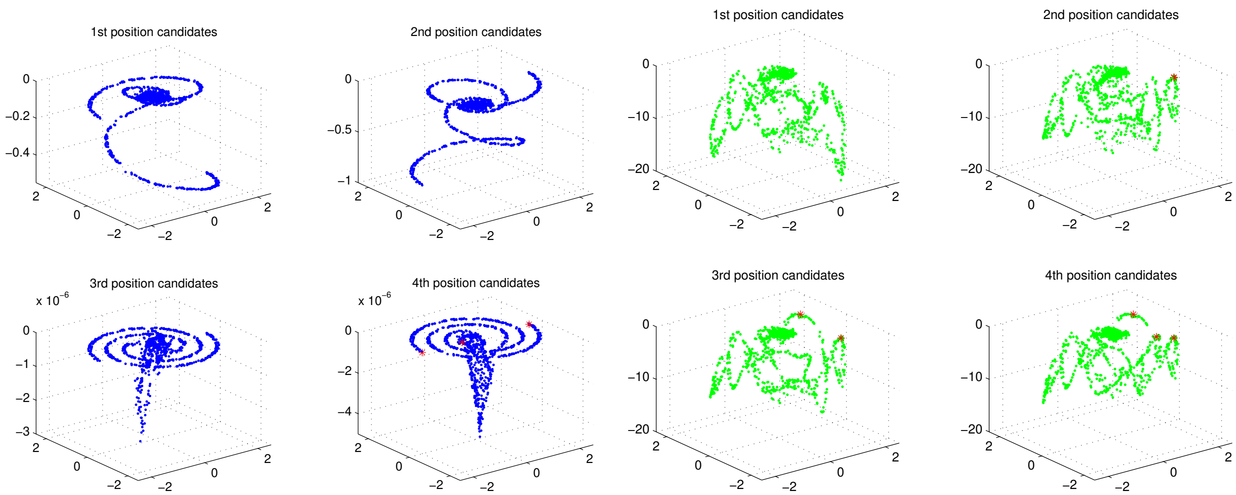

Figure 3: Mean field estimate of the change in loss function with the graph kernel (left) and the RBF kernel (right) for the first four iterations of clique selection on the galaxy dataset. For the graph kernel the endpoints of the spirals are chosen as the first two cliques.

below the 20% Bayes error rate of the non-sequence model, with both kernels and sufficient labeled data. However the graph kernel is able to learn the structure much faster than the RBF kernel. Evidently the high error rate for low label data sizes prevents the RBF model from effectively using the context.

Finally we examine clique selection in KCRFs. For this experiment we use 50 training sequences. We use the mean field approximation and only select vertex cliques. At each iteration the selection is based on the estimated change in risk for each candidate vertex (training position). We plot the estimated change in risk for the first four iterations of clique selection, with the graph kernel and RBF kernel respectively in Figure 3. Smaller values (lower on z-axis) indicate good candidates with potentially large reduction in risk if selected. For the graph kernel, the first two selected vertices are sufficient to reduce the risk essentially to the minimum (note in the third iteration the z-axis scale is already 10^{-6}). Such reduction does not happen with the RBF kernel.

6 Protein Secondary Structure Prediction

We used the RS126 dataset, on which many current secondary structure prediction methods have been developed and tested (Cuff and Barton, 1999). It is a non-homologous dataset, since among the 126 protein chains, no two proteins share more than 25% sequence identity over a length of more than 80 residues (Cuff and Barton, 1999). The dataset can be downloaded from `http://barton.ebi.ac.uk/`.

We adopt the DSSP definition of protein secondary structure (Kabsch and Sander, 1983), which is based on hydrogen bonding patterns and geometric constraints. The 8 DSSP labels are reduced to a 3 state model as follows: H & G map to helix (H), E & B map to sheets (E), and all other states map to coil (C) (Cuff and Barton, 1999).

For protein secondary structure prediction, state-of-the-art performance is achieved by window-base methods, using the position-specific scoring matrices (PSSM) as input features, i.e., PSI-BLAST profiles (Jones, 1999), together with Support Vector Machines (SVMs) as the underlying

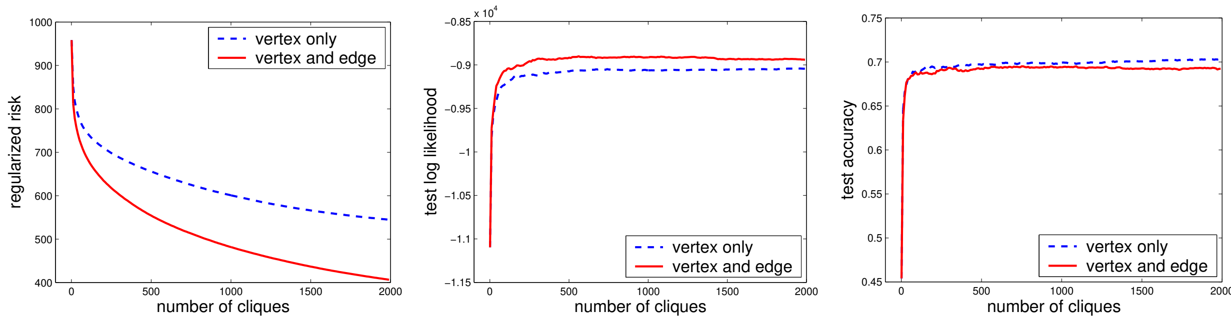

Figure 4: Clique selection for KCRFs on a particular trial. Left: regularized risk; Center: test set log likelihood; Right: test set accuracy. The two curves show the use of vertex cliques only (dashed) and both vertex and edge cliques (solid).

learning algorithm (Kim and Park, 2003). In our experiments, we apply a linear transformation L to the PSSM matrix elements according to $L(x) = 0$ for $x \leq -5$, $L(x) = \frac{1}{2} + \frac{x}{10}$ for $-5 \leq x \leq 5$, and $L(x) = 1$ for $x \geq 5$. This is the same transform used by Kim and Park (2003) in the recent CASP (Critical Assessment of Structure Predictions) competition, which achieved one of the best results. The window size is set to 13 by cross-validation. Therefore the number of features per position is 13×21 (the number of amino acids plus gap).

6.1 Clique selection.

We use an RBF kernel with bandwidth $\sigma = 0.1$ chosen by cross-validation. Figure 4 (left) shows the kernel CRF risk reduction as clique selection proceeds, when only vertex clique candidates are allowed (note there are always position independent edge parameters in the KCRF models, to prevent the models from degrading into kernel logistic regression), and when both vertex and edge cliques are allowed. The kernel between vertex cliques is $\overline{K}(\boldsymbol{z}, \boldsymbol{z}') = \overline{K}(x, x')\,\delta(y_1, y_1')$, and between edge cliques it is $\overline{K}(\boldsymbol{z}, \boldsymbol{z}') = \overline{K}(x, x')\,\delta(y_1, y_1')\,\delta(y_2, y_2')$. The total number of clique candidates is about 4800 (vertex only) and 20000 (vertex and edge). The rapid reduction in risk indicates sparse training of kernel CRFs is successful. Also when more flexibility is allowed by including edge cliques, the risk reduction is much faster. The more flexible model also has higher test set log likelihood (center); however, this does not translate into test set accuracy (right).

6.2 Per-residue accuracy.

To evaluate prediction performance, we use the overall per-residue accuracy (i.e., Q_3). We experiment with training set size of 5 and 10 sequences respectively. For each size we perform 10 trials where the training sequences are randomly sampled, and the remaining proteins are used as the test set. For kernel CRF we select 300 cliques, again from either vertex candidates alone or vertex and edge candidates. We compare them with the SVM-light package (Joachims, 1998) for SVM classifier. All methods use the same RBF kernel. See Table 1. KCRFs and SVMs have comparable performance.

6.3 Transition accuracy.

Further information can be obtained by studying transition boundaries, for example, the transition from "coil" to "sheet." From the point of view of structural biology, these transition boundaries may

	5 protein set		10 protein set	
Method	Accuracy	std	Accuracy	std
KCRF (v)	0.6625	0.0224	0.6933	0.0276
KCRF (v+e)	0.6562	0.0202	0.6933	0.0272
SVM	0.6509	0.0307	0.6875	0.0235

Table 1: Per-residue accuracy of different methods for secondary structure prediction, with the RBF kernel. KCRF (v) uses vertex cliques only; KCRF (v+e) uses vertex and edge cliques.

	5 protein set		10 protein set	
Method	Accuracy	std	Accuracy	std
KCRF (v)	0.1097	0.0271	0.1462	0.0235
KCRF (v+e)	0.1114	0.0250	0.1522	0.0214
SVM	0.0667	0.0313	0.1066	0.0311

Table 2: Transition accuracy with different methods.

	5 protein set		10 protein set	
Method	Accuracy	std	Accuracy	std
KCRF (v)	0.6722	0.0194	0.6854	0.0190
KCRF (v+e)	0.6674	0.0201	0.6819	0.0194
Trans. SVM	0.6480	0.0276	0.6813	0.0210

Table 3: Per-residue accuracy with semi-supervised methods.

provide important information about how proteins fold in three dimensions. From a computational and modeling perspective, they may indicate positions where most secondary structure prediction systems fail. We measure transition accuracy where a transition is defined as a pair of adjacent positions $(i, i+1)$ whose true labels differ. A transition is classified correctly only if both labels are correct. This is a very hard problem; see Table 2. There seems to be a small improvement with KCRFs. Figure 5 shows a plot of the true protein sequence labels and KCRFs' predicted labels.

6.4 Semi-supervised learning.

We start with an unweighted 10 nearest neighbor graph over positions in both training and test sequences, with the metric being Euclidean distance in the feature space. Then the eigensystem of the normalized Laplacian is computed. The semi-supervised graph kernel is obtained with the function $r(\lambda_i) = \frac{1}{\lambda+0.01}$ on the first $i \leq 200$ eigenvalues. The rest of the eigenvalues are set to zero. We use the graph kernel together with the RBF kernel in KCRF. As a clique candidate is associated with a kernel, we now select the two best candidates per iteration, one with the graph kernel and the other with the RBF kernel. We still run for 300 iterations for all trials. We also report the results using transductive SVMs (TSVMs) (Joachims, 1999) with the RBF kernel. See Table 3. The approximate average running time of each trial, including both training and testing, is 7 minutes for SVMs, 30 minutes for KCRFs, and 16 hours for transductive SVMs. For KCRFs

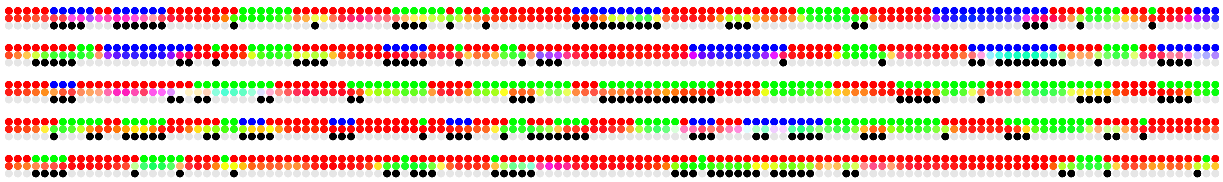

Figure 5: KCRF classification on 5 protein fragments (PDB ID top to bottom: 1S01, 1RHD, 1RBP, 1R092, and 1PYP). The first row of each protein shows the true labels as C(red), E(green), and H(blue). The second row shows the KCRF marginals, where the R,G,B color components reflect the posterior probabilities. The third row shows with black dots the decoding errors made by the model.

the majority of the time is spent on clique selection.

The semi-supervised graph kernel does not significantly improve performance. Diagnosing the cause, we find upon examining the graph together with all of the test labels, that the labels are not smooth with respect to the graph: on average only 54.5% of a node's neighbors have the same label as that node. Obviously it is useless to ask for smooth functions on the graph. The problem appears to lie in the Euclidean distance-based similarity graph. Detecting faulty graphs and constructing better graphs, without the use of large numbers of labeled examples, remain important areas for future research.

7 Conclusion

Kernel conditional random fields have been introduced as a framework for approaching graph-structured classification problems. A representer theorem was derived which shows how KCRFs can be motivated by first principles and regularization theory. The resulting techniques combine the strengths of hidden Markov models, or more general Bayesian networks, kernel machines, and standard discriminative linear classifiers including logistic regression and SVMs. The formalism of graphical processes that has been presented is general, and should apply naturally to a wide range of problems.

Our experimental results on synthetic data, while carefully controlled to be simple, clearly indicate how sequence modeling, graph kernels for semi-supervised learning, and clique selection for sparse representations work together within this framework. The success of these methods in real problems will depend on the choice of suitable kernels that capture the structure of the data.

For protein secondary structure prediction, our results are only suggestive. Secondary structure prediction is a problem that has been extensively studied for more than 20 years; yet the task remains difficult, with prediction accuracies remaining low. The major bottleneck lies in beta-sheet prediction, where there are long range interactions between regions of the protein chain that are not necessarily consecutive in the primary sequence. Our experimental results indicate that KCRFs and semi-supervised kernels have the potential to lead to progress on this problem, where the state of the art has been based on heuristic "sliding window" methods. However, our results also suggest that the improvement due to semi-supervised learning is hindered by the lack of a good similarity measure with which to construct the graph. The construction of an effective graph is a challenge that may best be tackled by biologists and machine learning researchers working together.

References

M. Belkin and P. Niyogi. Semi-supervised learning on manifolds. Technical Report TR-2002-12, University of Chicago, 2002.

O. Chapelle, J. Weston, , and B. Schoelkopf. Cluster kernels for semi-supervised learning. In *Advances in Neural Information Proceeding Systems*, volume 15, 2002.

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

J. Cuff and G. Barton. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins*, 34:508–519, 1999.

T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *European Conference on Machine Learning (ECML)*, pages 137–142, Berlin, 1998. Springer.

T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning (ICML'99)*, San Francisco, CA, 1999. Morgan Kaufmann.

D. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *J Mol Biol.*, 292:195–202, 1999.

W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

H. Kim and H. Park. Protein secondary structure prediction based on an improved support vector machines approach. *Protein Eng.*, 16:553–60, 2003.

G. Kimeldorf and G. Wahba. Some results on Tchebychean spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proc. 19th International Conf. on Machine Learning*, 2002.

S. Kumar and M. Hebert. Discriminative fields for modeling spatial dependencies in natural images. In *Advances in Neural Information Processing Systems 16*, 2003.

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

A. McCallum. Efficiently inducing features of conditional random fields. In *Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI03)*, 2003.

D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 235–242. ACM Press, 2003. ISBN 1-58113-646-3.

F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL*. Association for Computational Linguistics, 2003.

A. Smola and R. Kondor. Kernels and regularization on graphs. In *Conference on Learning Theory, COLT/KW*, 2003.

M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *NIPS*, 2001.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems 16*, 2003.

S. X. Yu and J. Shi. Grouping with bias. In *NIPS*, 2001.

J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. In *Advances in Neural Information Processing Systems 14*, 2001.

X. Zhu, Z. Gharahmani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.