Fixed-Point Model For Structured Labeling

Quannan Li[1] QUANNAN.LI@GMAIL.COM
Jingdong Wang[2] JINGDW@MICROSOFT.COM
David Wipf[2] DAVIDWIPF@GMAIL.COM
Zhuowen Tu[1,2] ZHUOWEN.TU@GMAIL.COM
[1]Lab of Neuro Imaging and Department of Computer Science, UCLA
[2]Microsoft Research Asia

Abstract

In this paper, we propose a simple but effective solution to the structured labeling problem: a fixed-point model. Recently, layered models with sequential classifiers/regressors have gained an increasing amount of interests for structural prediction. Here, we design an algorithm with a new perspective on layered models; we aim to find a fixed-point function with the structured labels being both the output and the input. Our approach alleviates the burden in learning multiple/different classifiers in different layers. We devise a training strategy for our method and provide justifications for the fixed-point function to be a contraction mapping. The learned function captures rich contextual information and is easy to train and test. On several widely used benchmark datasets, the proposed method observes significant improvement in both performance and efficiency over many state-of-the-art algorithms.

1. Introduction

Here we study the problem of predicting a labeling for a structured input, which is denoted as a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node $v_i \in \mathcal{V}$ corresponds to a data entry with its features denoted as $\mathbf{x}_i$; the objective of the structured labeling task is to jointly assign labels $\mathbf{y} = (y_i : i = 1..|\mathcal{V}|)$ ($y_i \in \mathcal{L}$, $\mathcal{L}$ is the label space) to all nodes $\mathcal{V} = (v_i : i = 1..|\mathcal{V}|)$ as a joint output. This problem is fundamental since structured inputs and outputs are common in a wide range of applications. For example, in computer vision/image processing, a structured input is an image of all pixels, and the structured outputs are the corresponding labels of these pixels. There are correlations among the structured outputs, denoted by the edges between the nodes, and the correlation may occur between neighboring nodes, or the nodes relatively distant apart. These correlations make the structured prediction problem a difficult task.

A simple scheme is to treat the structured outputs as independent entries and apply the standard classification/regression algorithms. Such a scheme is straightforward, but it loses the important interdependency information, which is crucial in modeling and understanding the structured data. The other extreme of the solution is to treat each instance of $\mathbf{y} = (y_1, .., y_{|\mathcal{V}|})$ as a single label and transform the problem into a multiclass classification problem. This implementation is infeasible because the space of the output labels is exponentially large at the size of $|\mathcal{L}|^{|\mathcal{V}|}$.

Markov random fields (MRF) (Geman & Geman, 1984) and conditional random fields (CRF) (Lafferty et al., 2001) have been widely used to model the correlations of the structured labels. However, due to the heavy computational burdens in their training and testing (inference) stages, MRF and CRF are often limited to capturing a few neighborhood interactions, and thus, limiting their modeling capabilities. Structural SVM methods (Tsochantaridis et al., 2005) and maximum margin Markov networks (M^{3}N) (Taskar et al., 2003) model the correlation in a similar way as the CRF, but they try to specifically maximize the prediction margin. These approaches are also limited in the range of contexts due to the high computational demand. When long range contexts are used, approximations should be used to trade-off the accuracy and the running time (Finley & Joachims, 2008).

Recently, layered models (Tu & Bai, 2010; Heitz et al.,

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

2008; Daumé et al., 2009), in the spirit of stacking (Wolpert, 1992), are proposed to take the outputs of classifiers of the current layer as added features to classifiers of the next layer. Since these approaches perform direct label prediction as functions instead of performing inferences as in MRF or CRF, the layered models (Tu & Bai, 2010; Heitz et al., 2008) are able to model complex and long range contexts.

In this paper, we look into the structured labeling problem from a different angle and develop a simple yet effective approach, a fixed-point model. We introduce a contextual prediction function $f : (\mathbf{x}, \mathcal{L}^{|\mathcal{V}|-1}) \rightarrow \mathcal{L}$ with the output being the labeling of an individual node and the input being both its features and the labeling of the rest of the nodes (or its neighbors). The overall fixed-point function $\mathbf{f} : (\mathbf{x}_1, \cdots, \mathbf{x}_{|\mathcal{V}|}, \mathcal{L}^{|\mathcal{V}|}) \rightarrow \mathcal{L}^{|\mathcal{V}|}$ is a vector form of the contextual prediction function of the nodes, and is trained with the property of a contraction mapping so that an iterative solution is applicable in the prediction process. We also analyze conditions for ensuring that our training strategy leads to a contraction mapping, provably so in certain cases. Not only does the learned fixed-point function preserve the modeling capability of the layered models (Tu & Bai, 2010; Heitz et al., 2008), but also it is simper and much easier to scale since it only consists of a single layer function.

2. Related Work

The conditional random fields (CRF) model (Lafferty et al., 2001) is a state-of-the-art work for solving the structured prediction problem. In the max-margin Markov networks (Taskar et al., 2003; Tsochantaridis et al., 2005), the authors propose to maximize the margin for structured output, in a spirit similar to the multi-class SVM method (Weston & Watkins, 1998). Due to the computational demand in both the training and the testing stages, usually only a small number of interactions among the neighboring outputs are included in both the CRF and the M^{3}N. The hidden Markov models (Rabiner, 1989) share a similar property in modeling the graph connections.

The layered contextual models (Tu & Bai, 2010; Heitz et al., 2008) train a sequences of classifiers using the output of the previous layers as additional features to the next layer; on tasks where the long range contexts play a significant role, e.g., the OCR task, they greatly outperform the CRF and M^{3}N (as shown in the experiments). The proposed fixed-point model has a similar modeling capability to model long range contexts as the layered models. During the training process, while the layered models train a series of classifiers, the fixed-point model trains a single classification/regression function which assumes a stable status for the ground-truth labeling. Layered models have to compute the classification scores for each training sample as the input to the next layer, thus limiting their capability to scale up. On the contrary, the fixed-point model is much faster to train than the layered models, and thus is much more scalable. In addition, the convergence behavior of the layered models has not been clearly stated so far, whereas the proposed fixed-point model provides a contraction mapping interpretation to the convergence.

The pseudo-likelihood algorithm in (Besag, 1975) models the conditional probability of an entry based on its neighborhoods; in (Sontag et al., 2010), a pseudo-max framework is introduced to approximate the exponential number of constraints by a polynomial number of constraints; in structured output-associative regression (SOAR) (Bo & Sminchisescu, 2009), the output components, other than the one being considered, are used as auxiliary features to train a vector of regression functions for the task of image reconstructions and human pose estimation. Compared with SOAR, the main purpose of the proposed fixed-point model is to train a fixed-point function that assumes the stable status of the structured labels; in SOAR, a generalized linear regression function is trained for reconstruction, whereas we study the structured labeling problem by exploring rich contextual correlations; the lack of analysis in the learned function also leaves the convergence in SOAR untouched; on the contrary, our algorithm is not limited to generalized linear regressions and, existing methods such as logistic regression and random forest (Breiman, 2001) can be used too.

There are also other related algorithms. In (Collins, 2002), an averaged perceptron algorithm is proposed: the training samples are processed iteratively; once a mistake occurs, the weight vector is updated according to the prediction error and the final weight vector is a weighted version of all the vectors that have appeared. In (Nguyen & Guo, 2007), an ensemble method is proposed to transform the predictions of different models into a chain with state transition matrices and then dynamic programming is used to get the voted prediction result. The underlying mechanism of the fixed-point model is different from that of these algorithms, and we will compare the performance in Section 4.

It is worth mentioning that the proposed fixed-point model is not a method merely designed to balance the performance and learning-time; it provides a new way of thinking about the structured learning problem by investigating a shallow model (instead of cascaded approaches with deep layers) and also having the capability to incorporate rich structural/contextual information with effective and efficient inference.

3. The Fixed-Point Model

3.1. Model Description

In this paper, we are interested in the structured labeling task for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The edges $\mathcal{E}$ decides the graph topology, and hence the neighborhoods of each node. For instance, in sequence labeling, where the nodes $\{v_1, v_2, \cdots, v_n\}$ in $\mathcal{V}$ form a chain, we can specify the neighborhood $\mathcal{N}_i$ of v_i to be the m nodes preceding and after it, i.e., $\mathcal{N}_i = \{v_{i-m/2}, v_{i-m/2+1}, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{i+m/2}\}$. We use m to denote the number of neighbors a node can have in the neighborhood specification.

We assume that our problem is a binary-classification problem where $\mathcal{L} = \{-1, +1\}$. To this end, we train a contextual prediction function which outputs the labeling of the node. Note that, as a lexical category label, y_i cannot be used in the equation/function directly. Instead, we can represent y with a labeling confidence q. For the binary class case, if $y_i = 1$, $q_i = 1$ and if $y_i = -1$, $q_i = 0$. At the prediction process, the label y_i is unknown, and thus q can be relaxed to a real value ranging in $[0, 1]$. We use $\mathbf{q}_{\mathcal{N}_i}$ to denote the labeling of the neighborhood of v_i and use $\mathbf{q}$ to denote the labeling of all the nodes in $\mathcal{G}$. This can easily be extended to multiclass problems by encoding the labeling with a matrix.

For each node v_i, the contextual prediction function f takes in both v_i's feature $\mathbf{x}_i$ and the labeling $\mathbf{q}_{\mathcal{N}_i}$ of its neighborhood. The contextual prediction function f can be formulated as

$$q_i = f(\mathbf{x}_i, \mathbf{q}_{\mathcal{N}_i}; \boldsymbol{\theta}), \tag{1}$$

where f is a regression function within range $[0, 1]$, and $\boldsymbol{\theta}$ is the parameter of the function. From Equation 1, the labeling $\mathbf{q}$ of all the nodes can be written in a vector form,

$$\mathbf{q} = \mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta}), \tag{2}$$

where $\mathbf{q} = [q_1, q_2, \cdots, q_n]^T$, $\mathbf{f}(\cdot) = [f(\mathbf{x}_1, \mathbf{q}_{\mathcal{N}_1}; \boldsymbol{\theta}), f(\mathbf{x}_2, \mathbf{q}_{\mathcal{N}_2}; \boldsymbol{\theta}), \cdots, f(\mathbf{x}_n, \mathbf{q}_{\mathcal{N}_n}; \boldsymbol{\theta})]^T$.

As from Equation 2, the labeling $\mathbf{q}$ appears as both the output as well as part of the input. Given the labeling $\mathbf{q}$ and the features $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ of the training data, we learn the parameter $\boldsymbol{\theta}$.

To get the labeling of a structured input $\mathcal{G}$, one can solve for the non-linear equation set $\mathbf{q} = \mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta})$, which is generally a difficult task. In this paper, we focus on a type of functions $\mathbf{f}$ that assumes the property of a contraction mapping, i.e., having a stable status (an attractive fixed-point) for each structured input. When using the ground-truth labeling in the training process, the ground-truth labeling is assumed to be the stable status and the existence of the stable status leads to the fixed-point iteration in the prediction process: $\mathbf{q}^t = \mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}^{t-1}; \boldsymbol{\theta})$ and $\mathbf{q}^t \to \mathbf{q}$ as $t \to \infty$. We name the functions $\mathbf{f}$ with such a property *the fixed-point functions (models)*. In the following subsections, we provide a sufficient condition for the fixed-point model, propose the learning strategy, and describe the training and testing processes.

3.2. Contraction Condition for the Fixed-Point Model

In this section, we give a sufficient condition for $\mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta})$ to be the fixed-point model and illustrate it using a logistic regression model first.

Our derivation is based on the Banach Fixed-Point theorem (Banach, 1922): for a complete metric space (X, dist) and a mapping $\mathcal{F} : X \to X$, if there exists a non-negative real number $\rho < 1$ such that,

$$\mathrm{dist}(\mathcal{F}(a), \mathcal{F}(b)) \le \rho \times \mathrm{dist}(a, b), \forall a, b \in X, \tag{3}$$

then $\mathcal{F}$ is a contraction mapping and it has a unique fixed-point.

Since for a node v_i, its feature $\mathbf{x}_i$ is given and fixed, we thus for notational simplicity we can simply write $q_i = f_i(\mathbf{q}_{\mathcal{N}_i}; \boldsymbol{\theta})$ and $\mathbf{q} = \mathbf{f}(\mathbf{q}; \boldsymbol{\theta})$.

Assuming f_i is a continuously differentiable real-valued function, then according to the mean value theorem for a scalar function of several variables, $\forall\ \mathbf{q}, \overline{\mathbf{q}}$

$$f_i(\mathbf{q}_{\mathcal{N}_i}; \boldsymbol{\theta}) - f_i(\overline{\mathbf{q}}_{\mathcal{N}_i}; \boldsymbol{\theta}) = \left\langle \nabla_{f_i(\tilde{\mathbf{q}}_{\mathcal{N}_i}; \boldsymbol{\theta}), (\mathbf{q}_{\mathcal{N}_i} - \overline{\mathbf{q}}_{\mathcal{N}_i})} \right\rangle, \tag{4}$$

where $\nabla f_i(\tilde{\mathbf{q}}_{\mathcal{N}_i}; \boldsymbol{\theta})$ is the gradient of f_i at $\tilde{\mathbf{q}}_{\mathcal{N}_i} = (1 - c_i)\mathbf{q}_{\mathcal{N}_i} + c_i\overline{\mathbf{q}}_{\mathcal{N}_i}$, for some $0 < c_i < 1$. For $\mathbf{f}$,

$$\mathbf{f}(\mathbf{q}; \boldsymbol{\theta}) - \mathbf{f}(\overline{\mathbf{q}}; \boldsymbol{\theta}) \quad = \quad J_{\mathbf{f}}(\mathbf{q} - \overline{\mathbf{q}}), \tag{5}$$

where $J_{\mathbf{f}}$ is a matrix of coordinate-wise derivatives, and its i-th row corresponds to $\nabla f_i(\tilde{\mathbf{q}}_{\mathcal{N}_i}; \boldsymbol{\theta})$: for the (i, k)-th element of $J_{\mathbf{f}}$, if the node v_k is a neighbor of node v_i, $J_{\mathbf{f}_{i,k}} = \frac{\partial f_i}{\partial q_k}|_{\tilde{q}_k}$ ($\tilde{q}_k = (1 - c_i)q_k + c_i\overline{q}_k$); if v_k is not in the neighborhood of v_i, $J_{\mathbf{f}_{i,k}} = 0$. Clearly

$$\begin{aligned} \frac{\|\mathbf{f}(\mathbf{q}; \boldsymbol{\theta}) - \mathbf{f}(\overline{\mathbf{q}}; \boldsymbol{\theta})\|}{\|\mathbf{q} - \overline{\mathbf{q}}\|} &= \frac{\|J_{\mathbf{f}}(\mathbf{q} - \overline{\mathbf{q}})\|}{\|\mathbf{q} - \overline{\mathbf{q}}\|} \\ &\le \max_{\mathbf{q} - \overline{\mathbf{q}} \neq \mathbf{0}} \frac{\|J_{\mathbf{f}}(\mathbf{q} - \overline{\mathbf{q}})\|}{\|\mathbf{q} - \overline{\mathbf{q}}\|} = \|J_{\mathbf{f}}\|, \end{aligned} \tag{6}$$

where $\|J_{\mathbf{f}}\|$ denotes the matrix norm on $J_{\mathbf{f}}$ induced from some vector norm $\|\cdot\|$. For example, $\|J_{\mathbf{f}}\|_1 = \max_{1\le k\le n}\sum_{i=1}^{n}\left|\frac{\partial f_i}{\partial q_k}|_{\tilde{q}_k}\right|$, the induced ℓ_1 norm of $J_{\mathbf{f}}$. We then have the following:

Lemma 1 *If* $\|J_{\mathbf{f}}\| < 1 \ \forall\ \mathbf{q}, \overline{\mathbf{q}}$, *then* $\mathbf{f}$ *is a contraction mapping.*

3.2.1. Contraction for logistic regression

Now we assume a linear logistic regression model. $\boldsymbol{\theta}$ can be decomposed into $\boldsymbol{\alpha}$ ($\boldsymbol{\alpha} \in \mathcal{R}^{d\times 1}$, d is the dimension of $\mathbf{x}_i$) and $\boldsymbol{\beta}$ ($\boldsymbol{\beta} \in \mathcal{R}^{m\times 1}$), corresponding to the appearance feature $\mathbf{x}_i$ and the contextual feature $\mathbf{q}_{\mathcal{N}_i}$ respectively. $q_i = f_i(\mathbf{q}_{\mathcal{N}_i};\boldsymbol{\theta})$ can be formulated as

$$q_i = \frac{\exp(\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle + \langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\rangle)}{1+\exp(\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle + \langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\rangle)}. \qquad (7)$$

In the following, we use $I(k,j)$ to denote the index of the node that has v_k as its j-th neighbor and define an auxiliary function $h_i(\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-(\sum_{j=1}^{m}\beta_j-\sum_{j=1}^{m}|\beta_j|)/2)}{(1+\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-(\sum_{j=1}^{m}\beta_j+\sum_{j=1}^{m}|\beta_j|)/2))^2}$. In Lemma 2, we give a sufficient condition for $\|J_{\mathbf{f}}\|_1 < 1$ for logistic regression.

Lemma 2 *For the logistic regression model, if* $\max_{1\le k\le n}\sum_{j=1}^{m}|\beta_j|\,h_{I(k,j)}(\boldsymbol{\alpha},\boldsymbol{\beta}) < 1$, $\|J_{\mathbf{f}}\|_1 < 1$ *and the fixed-point function* $\mathbf{f}$ *is a contraction mapping.*

Proof If v_k is the j-th neighbor of v_i, then the partial derivative $\frac{\partial f_i}{\partial q_k} = \frac{\beta_j\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-\left\langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\right\rangle)}{(1+\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-\left\langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\right\rangle))^2}$. As q_k is in the range $[0,1]$, for the term $\langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\rangle$, its minimum is $(\sum_{j=1}^{m}\beta_j - \sum_{j=1}^{m}|\beta_j|)/2$, the sum of the negative entries of $\boldsymbol{\beta}$, and its maximum is $(\sum_{j=1}^{m}\beta_j + \sum_{j=1}^{m}|\beta_j|)/2$, the sum of the positive entries of $\boldsymbol{\beta}$. So $\frac{\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-\left\langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\right\rangle)}{(1+\exp(-\langle\boldsymbol{\alpha},\mathbf{x}_i\rangle-\left\langle\boldsymbol{\beta},\mathbf{q}_{\mathcal{N}_i}\right\rangle))^2} \le h_i(\boldsymbol{\alpha},\boldsymbol{\beta})$ and $\left|J_{\mathbf{f}_{i,k}}\right| = \left|\frac{\partial f_i}{\partial g_k}|_{\tilde{q}_k}\right| \le |\beta_j|\,h_i(\boldsymbol{\alpha},\boldsymbol{\beta})$.

Ignoring the boundary effect of the structured input, the k-th absolute column sum of $J_{\mathbf{f}}$ sums over the m nodes that have v_k as their neighbor giving

$$\sum_{i=1}^{n}\left|J_{\mathbf{f}_{i,k}}\right| = \sum_{i:v_k\in\mathcal{N}_i}\left|\frac{\partial f_i}{\partial q_k}|_{\tilde{q}_k}\right| \le \sum_{j=1}^{m}|\beta_j|\,h_{I(k,j)}(\boldsymbol{\alpha},\boldsymbol{\beta}). \qquad (8)$$

Thus, $\|J_{\mathbf{f}}\|_1 \le \max_{1\le k\le n}\sum_{j=1}^{m}|\beta_j|\,h_{I(k,j)}(\boldsymbol{\alpha},\boldsymbol{\beta})$. If $\max_{1\le k\le n}\sum_{j=1}^{m}|\beta_j|\,h_{I(k,j)}(\boldsymbol{\alpha},\boldsymbol{\beta}) < 1$, $\mathbf{f}$ is then a contraction mapping. ∎

The value of $\mathbf{x}$ plays a role in the constraint requirement in Equation 8. So long as the value of $\mathbf{x}$ satisfies the constraint requirement in Equation 8, the fixed-points can be guaranteed. Another possible sufficient condition is $\|\boldsymbol{\beta}\|_1 < 1$ as $\left|J_{\mathbf{f}_{i,k}}\right| = |\beta_j|\,q_i(1-q_i) < |\beta_j|$. This condition is much simpler but more difficult to satisfy because it ignores dependency on $\mathbf{x}$ entirely.

3.2.2. Contraction in general cases

The condition described above can be used as constraints for the fixed-point function in the training process when f is restricted to a logistic regression function. We now briefly discuss a scheme for learning a contraction function in a more general setting, which can be used to implicitly enforce the contraction condition for other functions.

It is well-known that adding some amount of input noise (also called input jitter) during the training process can improve the robustness of neural network classifiers (Reed et al., 1992). Moreover, this is generally true with recursive models such as the algorithm being proposed herein, in part by favoring the contraction condition. For example, when training the function $\mathbf{q} = \mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta})$, we may produce some replica $Q = \{\mathbf{q} + \boldsymbol{\delta}_r, r = 1..R\}$ by introducing small random perturbations $\boldsymbol{\delta}_r$. This small amount of randomness is added to the input $\mathbf{q}$ of the function f while keeping the targeted output the same as the ground-truth $\mathbf{q}$, leading to an augmented training set. Given modest assumptions on the space of classifiers and training algorithms, it is possible to show that when a sufficient number of such replica are included with suitable distribution, then a contraction mapping will be obtained as part of the learning process with high probability. Intuitively, this occurs because these replica will effectively reduce the relative importance of $\mathbf{q}$ as an input feature to $\mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta})$. In the case of logistic regression, this is tantamount to reducing the magnitude of the coefficients $\boldsymbol{\beta}$; however, with other classifiers the effect may be less transparent. While it is difficult to know the optimal distribution of such replica a priori, we have found empirically that a contraction mapping is consistently obtained without sensitivity to this distribution. For evaluation purposes, the gradient of any classifier can be computed in principle, numerically or analytically, to examine if the contraction condition is satisfied in a particular region. Additionally, if a classifier is ever observed to violate the contraction condition, we can always retrain after increasing the number and/or magnitude of the replica. Given some assumptions about the classifier and the replica, we next briefly discuss efficient methods for checking (at least locally around the training data) whether or not a contracting function has been obtained.

From Lemma 1, we want to guarantee that $\|J_{\mathbf{f}}\| < 1$, for some norm $\|\cdot\|$. If we choose the induced ℓ_∞ norm, this corresponds to the requirement that all rows of $J_{\mathbf{f}}$ have ℓ_1 norm less than one. This can be guaranteed if the gradient of each f_i with respect to $\mathbf{q}$ is less than one for all $\mathbf{q}$. Let $\psi_i(\mathbf{q})$ denote this gradient. Using a Taylor series expansion we can approximate each function f_i as

$$f_i(\mathbf{q}+\boldsymbol{\delta}_r) = f_i(\mathbf{q}) + \boldsymbol{\delta}_r \cdot \psi_i(\mathbf{q}) + O(\partial^n f_i, n \geq 2), \quad (9)$$

where for simplicity we use $f_i(\mathbf{q})$ to denote the i-th element of $\mathbf{f}(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{q}; \boldsymbol{\theta})$. It is not uncommon to assume a relatively smooth function f_i with small higher-order derivatives $O(\partial^n f_i), n \geq 2$. Moreover, we may also assume in some situations that the $\boldsymbol{\delta}_r$ are small, in which case higher-order terms can be largely ignored. Let $f_i(\mathbf{q}+\boldsymbol{\delta}_r) - f_i(\mathbf{q}) = e_r$ for all $r = 1..R$. Given the first-order Taylor-series approximation, we then have

$$\boldsymbol{\delta}_r \cdot \psi_i(\mathbf{q}) \approx e_r, \quad r = 1..R. \quad (10)$$

The above constraints represent a linear system that can be viewed as random samples of the unknown $\psi_i(\mathbf{q})$. These samples, which can be efficiently collected and monitored during the training process, can then be used to help determine whether or not the contraction condition is satisfied (at least in the locality of the training data). Depending on the neighborhood structure of the graph, we know that $\psi_i(\mathbf{q})$ will typically be sparse, with nonzero-valued locations inferred from the edges. In cases where this degree of sparsity is sufficiently high relative to the number of replica, we can simply solve for $\psi_i(\mathbf{q})$ directly via the above linear system. However, when this is not possible, we may still potentially estimate whether $\|\psi_i(\mathbf{q})\|_1 < 1$.

For example, assume for simplicity that the replica $\boldsymbol{\delta}_r$ are iid Gaussian distributed with zero mean and known covariance $\sigma^2 I$ (other distributions can be accommodated as well). It then follows that each e_r represents an iid sample from a zero-mean Gaussian with variance $\sigma^2\|\psi_i(\mathbf{q})\|_2^2$. Given R such samples, it is a simple matter to design any number of standard statistical tests to infer the likelihood that $\|\psi_i(\mathbf{q})\|_2^2 < C$ for any constant C. So we need only determine some C sufficiently small such that we ensure the contraction condition holds, namely $\|\psi_i(\mathbf{q})\|_1 < 1$ with high probability. Now assume that the number of significant elements in $\psi_i(\mathbf{q})$ is less than or equal to some value τ (in addition to zero-valued elements enforced by the graph, there are typically many other elements with marginal influence, although these locations may not be known). Using the well-known relationships among p-norms, if $\tau\|\psi_i(\mathbf{q})\|_2^2 < 1$, then $\|\psi_i(\mathbf{q})\|_1 \leq \sqrt{\tau}\|\psi_i(\mathbf{q})\|_2 < 1$. So $C = 1/\tau$ is an appropriate choice.

This methodology can be loosely used to show that our learned function satisfies the contraction condition of $\|\psi_i(\mathbf{q})\|_1 < 1$ at the $\mathbf{q}$ from the training data and in neighboring regions such that an affine approximation to the true $\mathbf{f}$ is sufficient. We could also potentially incorporate an additional penalty term to encourage each e_r and therefore $\psi_i(\mathbf{q})$ to be small. In practice, as shown in the experiments, our method can learn a good contraction mapping with few or even no perturbations during training. Moreover, it can quickly converge to good solutions even though we always initialize from $\mathbf{q} = \mathbf{0}$ in testing, demonstrating a nice convergence property of the fixed-point model.

Algorithm 1 The training process of the fixed-point model

```
Input: Training structures $\{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_N\}$ and their labelings $\{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_N\}$;
Output: The trained contextual prediction function $f$;
    1: For each $\mathbf{q}$, produce some replica $Q = \{\mathbf{q}+\boldsymbol{\delta}_r\}$ by adding random perturbations;
    2: For each node $v_i$, create the contextual feature $\overline{\mathbf{q}}_{\mathcal{N}_i}$ from the perturbed labeling $\overline{\mathbf{q}} \in Q$;
    3: Based on the feature $\mathbf{x}_i$ and $\overline{\mathbf{q}}_{\mathcal{N}_i}$, train a $f : q_i = f(\mathbf{x}_i, \overline{\mathbf{q}}_{\mathcal{N}_i}; \boldsymbol{\theta})$.
```

Algorithm 2 The testing process of the fixed-point model

```
Input: The testing structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$; the trained contextual prediction function $f$; the number of iterations $\mathcal{T}$; a threshold $\varepsilon$;
Output: The labeling $\mathbf{q}$ of $\mathcal{G}$;
Initialize: $t = 1$; for each $v_i \in \mathcal{V}$, $q_i^0 = 0$;
repeat
    1: For each node $v_i$, compute the labeling $q_i^t$: $q_i^t = f(\mathbf{x}_i, \mathbf{q}_{\mathcal{N}_i}^{t-1}; \boldsymbol{\theta})$;
    2: $t = t + 1$;
until $t \geq \mathcal{T}$ or $\|\mathbf{q}^t - \mathbf{q}^{t-1}\| \leq \varepsilon$.
$\mathbf{q} = [q_1^t, q_2^t, \cdots, q_n^t]^T$.
```

3.3. The Training and Prediction Processes

The training and prediction processes are depicted in Algorithm 1 and Algorithm 2. The training process is to learn a contextual prediction function f by favoring fixed-point solutions (or nearly so) using some perturbations. Once learned, the contraction mapping is applied iteratively to the new structured inputs. For a novel structured input $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the labeling q_i of a node $v_i \in \mathcal{V}$ is initialized with a value. Note that q_i is not sensitive to the choice of initialization, and it is simply initialized with 0 in our experiments.

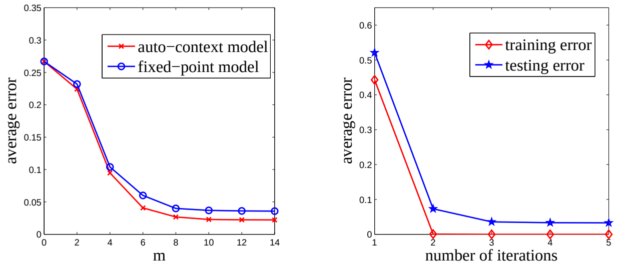
4. Experiments

We now apply the proposed fixed-point model to the tasks of Optical Character Recognition (OCR), Part-of-Speech tagging (POS) and Hypertext (web pages) classification. The data in OCR and POS have chain structure and the average error per sequence in (Nguyen & Guo, 2007) is used for performance evaluation. In hypertext classification, the linking structure of the hypertext is highly non-regular and we use the average labeling error over all the testing web pages.

4.1. Optical Character Recognition (OCR)

Optical character recognition involves the identification of letters in scanned texts. In this paper, the benchmark dataset (Taskar et al., 2003) is used.

In OCR, a word corresponds to a structured input $\mathcal{V}$ and the i-th character corresponds to v_i. We use the $m/2$ characters preceding and the $m/2$ characters after v_i as its neighbors and thus m indicates the complexity of the interdependence. For each character, its pixel values are concatenated to form $\mathbf{x}_i$. In training, the lexical label y_i is encoded to a $|\mathcal{L}|$-dimensional contextual feature vector, which has value 1 only at the entry corresponding to the value of y_i. In all, a $|\mathcal{L}| \times m$-dimensional contextual feature is created. In testing, the entry corresponding to the label with the maximum score is assigned 1 at each iteration. No perturbed replicas are used in OCR and we use kernel logistic regression (KLR) (Zhu & Hastie, 2001) as the contextual prediction function; at the testing process, 5 iterations are used, i.e., $\mathcal{T} = 5$.



(a) (b)

Figure 1. (a): comparison of the testing errors on the OCR dataset by varying m; (b): the training and testing errors on the OCR dataset as $\mathcal{T}$ varies.

In Fig. 1 (a), we compare the fixed-point model with the auto-context model. Both of the two methods use KLR with RBF kernel as the classifier. We note that the original result of the auto-context model reported in (Tu & Bai, 2010) is only 19.5% because the Harr-like features used in that paper are not so effective on the OCR dataset. We compare the two models by varying m from 0 to 14. As from Fig. 1 (a), the errors decrease monotonously as m increases. The fixed-point model performs slightly worse than the auto-context model, but it is simpler and more efficient: it takes about 7.55 minutes to train the model, while the auto-context model takes around 50 minutes because the auto-context model needs to train $\mathcal{T}$ classifiers sequentially and apply the classifiers to each of the training sequences.

Table 1. Average errors on the OCR dataset in percentage. The results of SVM^{struct2} and CRF^2 are from (Keerthi & Sundararajan, 2007)

SVM^{struct}	**SVM**^{struct2}	**M**^{3}**N**	**SVM**
21.16	19.24	25.08	28.54
Perceptron	**KLR**	**SEARN**	**CRF**
26.4	26.7	27.02	32.30
CRF^2	**HMM**	**SLE**	**KCGM**
19.97	23.7	20.58	5.8
Auto-context	**Fixed-Point**		
2.22	3.6		

In Table 1, we compare the fixed-point model with several state-of-the-art algorithms: SVM (Crammer & Singer, 2001), SVM^{struct} (Tsochantaridis et al., 2005), M^{3}N (Taskar et al., 2003), Perceptron (Collins, 2002), KLR (Zhu & Hastie, 2001), SEARN (Daumé et al., 2009), CRF (Lafferty et al., 2001), HMM (Rabiner, 1989), structured learning ensemble (SLE) (Nguyen & Guo, 2007), kernel conditional graphic model (KCGM) (Cruz et al., 2007) and the auto-context model (Tu & Bai, 2010). For SVM^{struct}, M^{3}N and CRF, the results are from (Nguyen & Guo, 2007) with linear kernel. As from Table 1, with the exception of the auto-context model, the fixed-point model outperforms the state-of-the-art methods.

In (Keerthi & Sundararajan, 2007), CRF and structural SVM are implemented with a different set of features and the performance is better than that in (Nguyen & Guo, 2007), see CRF^2 and SVM^{struct2} in Table 1. Still, the errors are much higher than that of the fixed-point model. In (Taskar et al., 2003), M^{3}N reports the average error per character 12.8% with cubic kernel while the average error per character of the fixed-point model is 2.13%. One may argue that if CRF, SVM^{struct} and M^{3}N were to use the contexts like those in the fixed-point model, they would generate similar results. However, it is exactly their large computational burden in taking into account long range interactions that limits their modeling ability.

In Fig. 1 (b), we illustrate the convergence rate of the fixed-point model, revealing that both the training and testing errors are very small after the second

Table 2. Average errors on the POS dataset in percentage. The results of SVM^{struct2} and CRF^2 are from (Keerthi & Sundararajan, 2007).

Train Size	500	1000	2000	4000	8000
SVM-L1	8.74	6.74	5.67	4.81	4.28
SVM^{struct}	8.37	6.58	5.75	4.71	4.08
SVM^{struct2}	8.38	7.15	5.63	-	-
M^{3}N	10.19	7.26	6.34	5.26	4.19
Perceptron	10.16	7.79	6.38	5.39	4.49
SEARN	10.49	8.92	7.58	6.44	5.48
CRF	16.53	12.51	9.84	7.76	6.38
CRF^2	8.84	7.08	5.83	-	-
HMM	23.46	19.95	17.96	17.58	15.87
SLE	7.71	5.93	5.14	4.19	3.67
Auto-context	8.12	6.34	5.38	4.6	3.91
Fixed-point	8.24	6.40	5.48	4.66	4.02

iteration. This suggests that we are able to train a fixed-point function that satisfies the conditions for convergence. In addition, the fixed-point model converges very quickly at the testing stage with only $2 \sim 3$ iterations.

4.2. Part-of-Speech Tagging (POS)

For the Part-of-Speech Tagging task, we use the POS dataset (Treebank, 2002) and comply with the training/validation/testing splits in (Nguyen & Guo, 2007). For each word, $446,054$ lexical features are used. We use the L1 regularized support vector machine (SVM-L1) provided in the LIBLINEAR software package (Fan et al., 2008) as the classifier.

In our experiment, $m = 6$ is used as it gives the best results on the validation datasets. For each sequence, one perturbed replica is produced using Gaussian noise with $\boldsymbol{\delta} = 0.25$ and the contextual prediction function is trained with the perturbed replica and the original sequences.

In Table 2 and Table 3, we compare the average errors and the times to train the classifiers respectively. HMM is the most efficient in training, but its performance is poor. The fixed-point model is nearly as efficient as SVM-L1 since it needs only more feature dimensions and more training samples than SVM-L1; it is much simpler and more efficient than algorithms such as the auto-context model: on the data split of $8,000$ training sentences, it takes 2.214 hours for the auto-context model to train, while it takes only 0.192 hours for the fixed-point model to train. The average error of the fixed-point model is slightly worse than the auto-context model but the difference is rather small. With the exception of the auto-context model and SLE, the fixed-point model outperforms the other methods. SLE performs the best but is the most complex since it is an ensemble method using about 200 different models and its training time is not listed in (Nguyen & Guo, 2007).

Table 3. Training times on the split of POS Dataset with $8,000$ sentences (in hour)

Train Size	500	1000	2000	4000	8000
SVM-L1	0.0027	0.004	0.0053	0.009	0.0166
SVM^{struct}	0.11	0.21	0.38	1.7	2.2
M^{3}N	12.0	22.9	46.2	144.4	204
Perceptron	0.107	0.22	0.53	1.02	1.27
SEARN	0.035	0.043	0.053	0.096	0.13
CRF	0.53	2.33	5.4	13.3	32.7
HMM	6E-5	8E-5	1E-4	2E-4	3E-4
Auto-context	0.144	0.271	0.543	1.097	2.214
Fixed-point	0.009	0.019	0.037	0.08	0.192

Table 4. Comparison on the WebKB Dataset.

	CRF	SVM	Auto-context	Fixed-Point
error (%)	15	22.95	16.55	16.47
train time(s)	3005	0.05	1.85	0.3

4.3. Hypertext Classification

Hypertext classification aims to classify the web pages based on their contents and the linking structures. We use the WebKB dataset in (Craven et al., 1998) which contains web pages from 4 universities: Cornell, Texas, Washington and Wisconsin. Each page belongs to one of the 5 categories: course, faculty, student, project or other. The Bag of Words representation is used, and a codebook with $40,195$ codes is built using Rainbow (McCallum, 1996). We compare the fixed-point model with SVM, CRF, and the auto-context model. The statistics (a normalized histogram) of the labels of the in-linking and out-linking pages are used as the context feature. The fixed-point model and the auto-context algorithm both achieve small average errors when the third-order in-linking and out-linking statistics are used. For CRF, we adopt the UGM toolbox (Schmidt, 2011) and use loopy belief propagation for the inference. The first order, token-independent first order, and token-independent second order feature functions are used as these feature functions give the best performance in (Keerthi & Sundararajan, 2007).

The models are trained on three universities and tested on the remaining one. The average errors of the 4 universities are reported in Table 4. With one layer of fixed-point function, the proposed method achieves comparable result but is more efficient than the auto-context model. CRF performs the best but is much more computationally demanding.

5. Conclusions

In this paper, we have proposed a fixed-point model for the structured labeling problem. The fixed-point model takes the labeling of the structure as both the input and the output with the assumption that the ground-truth labeling being the stable status for the function. The fixed-point model preserves the ability to capture long range contexts as in more complex layered models. A simple learning strategy is adopted and contraction conditions are analyzed. On three structured labeling problems, the fixed-point model has achieved encouraging performance and efficiency.

Acknowledgement

This project is supported by NSF CAREER award IIS-0844566, NSF award IIS-1216528, and Microsoft Research Asia. We thank Jian Zhang and Xianghang Liu for the initial discussions; we thank Bo Wang and Liwei Wang for verifying the codes; also, we want to thank the anonymous reviewers for their valuable comments.

References

Banach, Stefan. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. In *Fund. Math*, pp. 133–181, 1922.

Besag, Julian. Statistical Analysis of Non-Lattice Data. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 24(3):179–195, 1975.

Bo, Liefeng and Sminchisescu, Cristian. Structured output-associative regression. In *CVPR*, pp. 2403–2410, 2009.

Breiman, Leo. Random forests. *Machine Learning*, 45(1): 5–32, 2001.

Collins, Michael. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, pp. 1–8, 2002.

Crammer, Koby and Singer, Yoram. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

Craven, Mark, DiPasquo, Dan, Freitag, Dayne, McCallum, Andrew, Mitchell, Tom M., Nigam, Kamal, and Slattery, Seán. Learning to extract symbolic knowledge from the world wide web. In *AAAI/IAAI*, pp. 509–516, 1998.

Cruz, F. Perez, Ghahramani, Z., and Pontil, M. Kernel conditional graphical models. *Predicting Structured Data*, pp. 265–282, 2007.

Daumé, Hal III, Langford, John, and Marcu, Daniel. Search-based structured prediction. *Machine Learning*, 75(3):297–325, 2009.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. Liblinear: A library for large linear classification, 2008.

Finley, Thomas and Joachims, Thorsten. Training structural svms when exact inference is intractable. In *ICML*, pp. 304–311, 2008.

Geman, Stuart and Geman, Donald. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6): 721–741, 1984.

Heitz, Geremy, Gould, Stephen, Saxena, Ashutosh, and Koller, Daphne. Cascaded classification models: Combining models for holistic scene understanding. In *NIPS*, pp. 641–648, 2008.

Keerthi, S. Sathiya and Sundararajan, S. Crf versus svm-struct for sequence labeling. In *Yahoo Research Technical Report*, 2007.

Lafferty, John D., McCallum, Andrew, and Pereira, Fernando C. N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pp. 282–289, 2001.

McCallum, Andrew Kachites. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

Nguyen, Nam and Guo, Yunsong. Comparisons of sequence labeling algorithms and extensions. In *ICML*, pp. 681–688, 2007.

Rabiner, Lawrence R. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pp. 257–286, 1989.

Reed, R., Oh, S., and Marks, R. J. Regularization using jittered training data. In *IJCNN*, pp. 509–516, 1992.

Schmidt, Mark. Ugm: Matlab code for undirected graphical models. 2011.

Sontag, David, Meshi, Ofer, Jaakkola, Tommi, and Globerson, Amir. More data means less inference: A pseudo-max approach to structured learning. In *NIPS*, pp. 2181–2189, 2010.

Taskar, Benjamin, Guestrin, Carlos, and Koller, Daphne. Max-margin markov networks. In *NIPS*, 2003.

Treebank, The Penn. Penn's linguistic data consortium. http://www.cis.upenn.edu/treebank. 2002.

Tsochantaridis, Ioannis, Joachims, Thorsten, Hofmann, Thomas, and Altun, Yasemin. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

Tu, Zhuowen and Bai, Xiang. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(10):1744–1757, 2010.

Weston, J. and Watkins, C. Multi-class support vector machines. Technical Report, 1998.

Wolpert, D. Stacked generalization. In *Neural Networks*, pp. 241–259, 1992.

Zhu, Ji and Hastie, Trevor. Kernel logistic regression and the import vector machine. In *NIPS*, pp. 1081–1088, 2001.