Pattern Recognition 43 (2010) 3683–3692

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Large margin cost-sensitive learning of conditional random fields

Minyoung Kim

Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

ARTICLE INFO

Article history:
Received 2 December 2009
Received in revised form
7 April 2010
Accepted 6 May 2010

Keywords:
Conditional random fields
Cost-sensitive learning

ABSTRACT

We tackle the structured output classification problem using the Conditional Random Fields (CRFs). Unlike the standard 0/1 loss case, we consider a cost-sensitive learning setting where we are given a non-0/1 misclassification cost matrix at the individual output level. Although the task of cost-sensitive classification has many interesting practical applications that retain domain-specific scales in the output space (e.g., hierarchical or ordinal scale), most CRF learning algorithms are unable to effectively deal with the cost-sensitive scenarios as they merely assume a nominal scale (hence 0/1 loss) in the output space. In this paper, we incorporate the cost-sensitive loss into the large margin learning framework. By large margin learning, the proposed algorithm inherits most benefits from the SVM-like margin-based classifiers, such as the provable generalization error bounds. Moreover, the soft-max approximation employed in our approach yields a convex optimization similar to the standard CRF learning with only slight modification in the potential functions. We also provide the theoretical cost-sensitive generalization error bound. We demonstrate the improved prediction performance of the proposed method over the existing approaches in a diverse set of sequence/image structured prediction problems that often arise in pattern recognition and computer vision domains.

1. Introduction

The problem of predicting from observation the labels that are statistically correlated within some combinatorial structures like chains or lattices is of great importance. The task is often named the *structured output prediction* in machine learning and pattern recognition fields, and appears in a broad spectrum of application domains including annotating natural language sentences (e.g., parsing, chunking, named entity recognition), labeling biological sequences (e.g., protein secondary structure prediction), and classifying regions of images (e.g., image segmentation with object recognition), to name a few.

The conditional random field (CRF), the log-linear model representing the conditional distribution of the output labels given the observation, is one of the most successful approaches to the structured output prediction problem, and have received significant attention in machine learning and related areas [1–6]. The CRFs, by effectively capturing the dependency structure in the output variables, are shown to achieve superb prediction performance in a variety of scenarios.

In this paper, we are interested in applying CRFs to a cost-sensitive learning setting, where unlike the standard 0/1 loss case, the misclassification costs are not equal, but different depending on the predicted class and the true class. The cost-sensitive loss is often represented as a cost matrix C, a $(S \times S)$ matrix whose (i,j)-entry C_{ij} contains the cost of misclassifying an output variable of true class j to class i, where S is the class cardinality of each output variable. The standard 0/1 loss corresponds to the all-1 matrix except 0s in the diagonal.

Incorporating a non-0/1 cost matrix often leads to many interesting practical applications in the structured output prediction problem. A typical example is the object recognition/segmentation in image, where we introduce an output variable for each pixel (or site) to represent the object class of the corresponding pixel. The output space, the set of all object classes, often retains a particular hierarchy or taxonomy. For instance, consider five object categories, `rose`, `tulip`, `lily`, `mouse`, and `hamster`, where the first three belong to a super-category `flower`, and the rest two `rodent`. In the 0/1 loss case, all the five object categories are treated individually and independently, which may often lead to less intuitive interpretation. For instance, predicting `mouse` to `hamster` incurs the same cost as predicting it to `rose`. Obviously, it makes more sense to penalize the latter situation more than the former. This can be addressed by defining a cost matrix that reflects the hierarchical scale of the output space, namely, misclassification *within* a super-category (`flower` or `rodent`) should cost less than misclassification *across* super-categories.

Another important motivation for the cost-sensitive structured output prediction arises when one deals with *real-valued* outputs. Probabilistic models like CRFs bear inherent difficulty in handling real-valued structured output variables directly as they require density integration over output variables, which, unlike discrete

E-mail addresses: mikim@andrew.cmu.edu, mikim21@gmail.com.

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.05.013

cases, is often highly intractable and sometimes even unbounded. Although some efficient inference and learning algorithms have been proposed by [7] to address the density integrability issue for the real-valued output CRFs, the model is restricted to sequence structures with a rather strong Gaussianity assumption. An alternative, practically more appealing solution is to discretize the output space into ordinal-scale classes. In this case, however, the metric information in the original output should be preserved in the discretized output in a sensible manner. For instance, consider that each output y is real-valued on a line [0,1]. Suppose that we discretize y into 10 classes ($k=1,\ldots,10$) where y is imputed to k if $(k-1)/10 \le y \le k/10$. It is desirable to have the discretized output maintain the ordinal scale (e.g., predicting the class 5 as 4 or 6 should cost less than classifying it as 1 or 10). This can be easily encoded in the cost matrix C by letting C_{ij} be a monotonically increasing function of $|i-j|$ (i.e., the off-diagonal entries of C have higher values than those inside).

Throughout the paper we assume that the non-0/1 cost matrix C is given a priori, possibly with the aid of domain experts, although defining a cost matrix often requires subjective judgement. More generally, in the structured output case, it is important to properly represent the preference interrelation among the adjoining sites (output variables) within the underlying graph structure.

Our main goal is to learn the parameters of the CRF model so as to improve the performance of predicting structured outputs in a cost-sensitive scenario. Even though there exist many sophisticated learning algorithms for CRFs that aim at either optimizing penalized conditional likelihoods [2,8] or maximizing margins in 0/1 loss sense [9,10], all these approaches tacitly assume the 0/1 loss, treating all types of misclassification errors equally costly, and hence, may not be optimal for the cost-sensitive settings.

Cost-sensitive learning in static (i.e., non-structured output) cases has received significant attention in data mining and pattern recognition areas [11–15]. These approaches are essentially based on accurate estimation of class-conditioned distribution, and were applied to many interesting applications including targeted marketing and fraud/intrusion detection. Despite their effectiveness and broad success, migrating these concepts to the CRF learning in the structured domain is non-trivial and less popular.[1] One may naively tackle the cost-sensitive structured output prediction problem using static methods by regarding individual output variable prediction as an independent static classification task. However, this is suboptimal, unable to fully take advantage of the structural dependency information encoded in the data. The use of structured models like CRFs is crucial for capturing structural dependencies, and failing to use such information can potentially cause significant performance degradation.

We propose a new learning algorithm for CRFs that incorporates the cost-sensitive loss into the large margin learning framework. The main idea is to enforce the margin, defined as the difference between model scores toward the correct class and the best predicted class to be greater than the corresponding loss. It thus has the effect of separating decision boundaries whose margin sizes depend on the quality of the prediction; we tolerate a small margin if the prediction incurs a small loss, while we force the model to retain a large margin if the misclassification loss is large. This is an intuitively appealing argument in that the learned model can have decision boundaries that respect the cost-sensitive loss.

Our approach can be seen as a parametric extension of SVMs to a cost-sensitive, non-static structured output domain. This gives rise to a model that inherits benefits of SVM-like margin-based approaches, most importantly, the provable generalization error bounds [17–20]. To handle the exponential number of margin constraints originated from the structured outputs, we use a soft-max approximation, essentially imposing stronger constraints than the original optimization problem. This reduces the learning problem to a convex optimization, solvable by efficient gradient search, which is algorithmically similar to the standard CRF learning with only slight modification of the potential functions.

The paper is organized as follows: In the next two sections we formally set up the problem, introduce notations, and briefly review CRFs. In Section 4, we propose our max-margin cost-sensitive learning for CRFs. After reviewing related work in Section 5, the experimental evaluation is provided in Section 6, which demonstrates the benefits of our approach over the existing methods.

2. Problem setup and notation

Unlike standard classification problems where there is only a single output variable y to be predicted, in the structured output prediction problems we deal with *multiple* output variables (denoted by boldfaced $\mathbf{y}$ for distinction). $\mathbf{y}$ is composed of individual output variables y_r (i.e., $\mathbf{y}=\{y_r\}_r$) where r is the variable index. Each y_r is assumed to take S different class labels.

The output variables in $\mathbf{y}$ are correlated within some combinatorial structure. Among various types of structures, in this paper we are particularly interested in two special structures: 1*D sequences* and 2*D lattices* (c.f., Fig. 1). They are of great importance due to their popularity and algorithmic simplicity, while extension to more complex arbitrary graph structures is relatively straightforward. These structures frequently arise in many interesting problems: natural language sentences, protein sequences, and general time-series have 1D chain structures, while 2D lattices subsume images and geographical maps, to name a few.

The observation, denoted by $\mathbf{x}$, is often structured similarly as the output $\mathbf{y}$, and serves as input covariates used for predicting $\mathbf{y}$. For instance, for the named entity recognition, a popular structured prediction task in the field of natural language processing, the input $\mathbf{x}=\mathbf{x}_1\ldots\mathbf{x}_T$ is a sentence of T words or phrases, and we predict the sequence of tags $\mathbf{y}=y_1\ldots y_T$ where y_r ($r=1,\ldots,T$) indicates whether the phrase $\mathbf{x}_r$ is either a person, a location, or an organization. In the speech recognition, the observation $\mathbf{x}=\mathbf{x}_1\ldots\mathbf{x}_T$ is a sequence of T utterances from which we predict the corresponding transcripts $\mathbf{y}=y_1\ldots y_T$. In the task of simultaneous image segmentation and object recognition, the

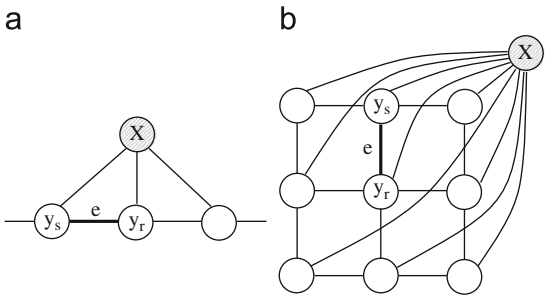

Fig. 1. Graphical representation of CRFs. (a) Chain (sequence); (b) lattice (image).

[1] Only recently, cost-sensitive learning for CRFs has been tackled in a sensible manner by [16]. This approach is based on the maximum entropy principle, where we contrast it with our approach in Section 5. Also in Section 6, we show that our approach exhibits superior prediction performance to theirs in a variety of situations.

observation $\mathbf{x} = \{\mathbf{x}_r\}_r$ is the input image (2D lattice) where $\mathbf{x}_r$ is the pixel (or site) intensities at position $r=(r_x,r_y)$. The output has a similar structure $\mathbf{y} = \{y_r\}_r$ where y_r takes the class label of the object that covers the (r_x,r_y)-region. Note that in all cases, the sizes of $\mathbf{y}$ and $\mathbf{x}$ (e.g., the sequence length T or the image sizes) can vary from instances to instances.

Recall that each individual output variable y_r can take S class categories. Without loss of generality, we let these class labels be $\{1,\ldots,S\}$. That is, $y_r \in \{1,\ldots,S\}$. Throughout the paper, we assume a supervised setting, implying that we are given a training set of n dyadic data points $\mathcal{D} = \{(\mathbf{x}^i,\mathbf{y}^i)\}_{i=1}^n$, which are presumably i.i.d. samples from the underlying but unknown distribution $P_*(\mathbf{x},\mathbf{y})$. The goal of the structured output prediction is to learn a classifier $\mathbf{y}=f(\mathbf{x})$ from data such that for a new test input $\mathbf{x}$, the predicted output $\mathbf{y}=f(\mathbf{x})$ is close to the true[2] output $\mathbf{y}^*$ as much as possible.

As a quantitative measure of the closeness between the predicted $\mathbf{y}$ and the true $\mathbf{y}^*$, one needs to define a loss (or mismatch) function $l(\mathbf{y}^*,\mathbf{y})$. A common choice adopted by most of the structured prediction algorithms is the so-called *Hamming loss*, $l_{0/1}(\mathbf{y}^*,\mathbf{y}) = \sum_r I(y_r^* \neq y_r)$ where I is the indicator function that returns 1 (0) if the argument is true (false). The Hamming loss is a natural extension of the 0/1 loss to the structured output domain. Using the Hamming loss, all types of mismatches are treated equally. On the other hand, in the cost-sensitive scenario, it is more reasonable to consider a mismatch loss that reflects the cost-sensitive loss at the individual output level, typically encoded by a $(S \times S)$ cost matrix C (i.e., $C(y_r^*,y_r)$). Then one can naturally define $l_C(\mathbf{y}^*,\mathbf{y}) = \sum_r C(y_r^*,y_r)$ as a cost-sensitive loss for structured outputs.

In what follows, we review the conditional random fields (CRFs), a probabilistic model $P(\mathbf{y}|\mathbf{x})$ that aims at fitting a conditional distribution of output given input observation, where the learning of CRFs is traditionally framed in the Hamming loss $l_{0/1}$. Our main contribution is to provide a novel learning method that accounts for a general cost-sensitive loss l_C via large margin paradigm, which will be described in Section 4.

3. Conditional random fields (CRFs)

The CRF is a log-linear model that represents the conditional distribution $P(\mathbf{y}|\mathbf{x})$ as the Gibbs form clamped on the observation $\mathbf{x}$:

$$P(\mathbf{y}|\mathbf{x},\boldsymbol{\theta}) = \frac{1}{Z(\mathbf{x};\boldsymbol{\theta})} e^{s(\mathbf{x},\mathbf{y};\boldsymbol{\theta})}. \tag{1}$$

Here $Z(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{y}\in\mathcal{Y}} e^{s(\mathbf{x},\mathbf{y};\boldsymbol{\theta})}$ is the normalizing partition function ($\mathcal{Y}$ is a set of all possible output configurations), and $\boldsymbol{\theta}$ is the parameters[3] of the *score function* (or the negative energy) that can be written as

$$s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \cdot \boldsymbol{\Psi}(\mathbf{x},\mathbf{y}), \tag{2}$$

where $\boldsymbol{\Psi}(\mathbf{x},\mathbf{y})$ is the joint feature vector.[4]

The score function plays a crucial role in the CRF. First of all, it determines the decision boundary of the model, and serves as a predictor (classifier):

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x},\boldsymbol{\theta}) = \operatorname*{argmax}_{\mathbf{y}}\ s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}). \tag{3}$$

In addition, the graph (dependency) structure of the output variables $\mathbf{y}$ can be encoded in the score function through the feature vector $\boldsymbol{\Psi}(\mathbf{x},\mathbf{y})$, which in turn factorizes the conditional distribution $P(\mathbf{y}|\mathbf{x})$ according to the dependency structure. More specifically, letting $\mathcal{C}$ be the set of all cliques in the output graph, the score function can be decomposed as $s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}) = \sum_{c\in\mathcal{C}} \boldsymbol{\theta}_c^\top \cdot \boldsymbol{\Psi}_c(\mathbf{x},\mathbf{y}_c)$ where $\mathbf{y}_c$ indicates the output variables confined to the clique $c \in \mathcal{C}$ (similarly for $\boldsymbol{\theta}_c$ and $\boldsymbol{\Psi}_c$). Then we have a factorized distribution $P(\mathbf{y}|\mathbf{x},\boldsymbol{\theta}) \propto \prod_{c\in\mathcal{C}} \exp(\boldsymbol{\theta}_c^\top \cdot \boldsymbol{\Psi}_c(\mathbf{x},\mathbf{y}_c))$, which effectively respects all the Markov dependency constraints represented by the output graph.

The choice of the output graph $G = (V,E)$ and the cliques critically affects model's representational capacity and the inference complexity. In this paper, we are interested in two popular graph structures: 1D chain for modeling sequences and 2D lattice for images (Fig. 1). For the notational convenience, we further assume that we have either *node* cliques ($r \in V$) or *edge* cliques ($e = (r,s) \in E$). We denote the node features by $\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)$ and the edge features by $\boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)$. By letting $\boldsymbol{\theta} = \{\mathbf{v},\mathbf{u}\}$ be the parameters[5] for node and edge features, respectively, the score function can then be expressed as

$$s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}) = \sum_{r\in V} \mathbf{v}^\top \cdot \boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r) + \sum_{e=(r,s)\in E} \mathbf{u}^\top \cdot \boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s). \tag{4}$$

This form has also been adopted for the CRFs with the sequence output [1] and the lattice output [3]. We call the product of parameters and the feature vectors on a clique the *(clique) potential* function. For instance, $\mathbf{v}^\top \cdot \boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)$ and $\mathbf{u}^\top \cdot \boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)$ are the node potential and the edge potential, respectively. Hence the score function is the sum of the potentials over all cliques in the graph.

We then consider learning the parameters $\boldsymbol{\theta} = \{\mathbf{v},\mathbf{u}\}$ of the CRF for n i.i.d. training data points $\{(\mathbf{x}^i,\mathbf{y}^i)\}_{i=1}^n$. Following the standard maximum a posteriori (MAP) estimation, one minimizes the regularized negative log-likelihood objective:

$$\begin{aligned} L_{MAP}(\boldsymbol{\theta}) &= -\sum_{i=1}^n \log P(\mathbf{y}^i|\mathbf{x}^i,\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2 \\ &= \sum_{i=1}^n (\log Z(\mathbf{x}^i;\boldsymbol{\theta}) - s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2, \end{aligned} \tag{5}$$

where λ is a constant balancing the negative log-likelihood term against the regularization term.

The cost $L_{MAP}(\boldsymbol{\theta})$ is a convex function and can be optimized via gradient search. The gradient of the score term trivially reduces to the node/edge features, while the gradient of the log-partition term can be derived as the expected features with respect to the current model $P(\mathbf{y}|\mathbf{x},\boldsymbol{\theta})$. That is,

$$\begin{aligned} \frac{\partial \log Z}{\partial \mathbf{v}} &= \sum_{r\in V} \mathbb{E}_{P(y_r|\mathbf{x})}[\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)], \\ \frac{\partial \log Z}{\partial \mathbf{u}} &= \sum_{e=(r,s)\in E} \mathbb{E}_{P(y_r,y_s|\mathbf{x})}[\boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)]. \end{aligned} \tag{6}$$

Hence, the minimization of (5) using gradient search essentially involves at every iteration: (i) evaluating the log-partition function, $\log Z$, for (5), and (ii) computing the output posteriors concentrated on cliques, $P(y_r|\mathbf{x})$ and $P(y_r,y_s|\mathbf{x})$, for (6), the task also known as the *inference*.

The task of inference essentially amounts to marginalization over output variables, and can be done by dynamic programming. The belief propagation (BP), the sum-product, and the junction

[2] Typically this implies the Bayes optimal prediction $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y}} P_*(\mathbf{y}|\mathbf{x})$.

[3] For notational simplicity, we often drop the dependency on $\boldsymbol{\theta}$ in notations. E.g., we use both $P(\mathbf{y}|\mathbf{x},\boldsymbol{\theta})$ and $P(\mathbf{y}|\mathbf{x})$ interchangeably.

[4] As CRFs belong to the exponential families, $\boldsymbol{\Psi}(\mathbf{x},\mathbf{y})$ is often referred to as the *sufficient statistics*.

[5] Following a conventional modeling practice, we assume that the parameters are shared across the cliques.

tree algorithm are well-known inference algorithms for arbitrary graph structures [21], all of which are algorithmically similar in that they pass the clique potential functions along the adjoining cliques. These algorithms are known to have time complexity exponential in the tree-width[6] of the graph. This implies that the exact inference is feasible for the sequence cases (tree-width = 2).

For the lattice structures, on the other hand, one has to resort to certain approximation methods as they have much larger tree-widths, making exact inference computationally intractable. Among various approximate inference algorithms, the loopy belief propagation (LBP) has been shown to work well empirically [22], and successfully applied to the CRFs in the image segmentation problems [8]. The LBP algorithm is similar in nature to the BP, passing messages (i.e., potential functions) along the adjacent cliques, where it repeats until the messages converge (which is not always guaranteed).

For the gradient descent optimizers, one can employ the quasi-Newton type limited-memory BFGS algorithm which has been shown to yield good performance for CRFs [2]. Recently, the stochastic gradient descent (SGD) methods were introduced to achieve faster convergence rates via gain adaptation using the second order information [8]. The SGD, due to its online treatment of data, is particularly effective for handling a large amount of training data.

Once the model is learned, at the testing stage, the output predicted by the model given the test input $\mathbf{x}$ is obtained by solving (3). The optimization of (3), often referred to as the *decoding*, can be solved by the dynamic programming similar to the inference algorithms on CRFs. Unlike the inference which essentially performs *sum-product*, the decoding involves *max-product* that can be achieved simply by replacing *sum* with *max* in the inference algorithms. For the 1D sequence cases, the max-product can be solved exactly by the so-called Viterbi decoding algorithm, while for 2D structures, one needs approximation like the LBP algorithm.

4. Large margin cost-sensitive learning of CRFs

The MAP parameter learning for CRFs implicitly assumes the Hamming 0/1 loss $l_{0/1}$, and it is not so straightforward to extend it to general non-0/1 loss cases. In this section we introduce a principled way to incorporate a cost-sensitive loss l_C into the CRF learning through the SVM-like margin maximization framework. This is motivated from previous success of the margin-based learning which is known to exhibit higher prediction performance than the likelihood-based learning [10,23,24,9].

Recall that CRF's score function $s(\mathbf{x},\mathbf{y};\boldsymbol{\theta})$ determines the prediction function, i.e., $f(\mathbf{x})=\operatorname{argmax}_{\mathbf{y}} s(\mathbf{x},\mathbf{y};\boldsymbol{\theta})$. Given the training sample $(\mathbf{x}_i,\mathbf{y}_i)$, it is thus desirable to assign the largest score to $(\mathbf{x}_i,\mathbf{y}_i)$ than all other possible output configurations, i.e., $s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta}) \geq s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})$ for all $\mathbf{y}\neq\mathbf{y}^i$, as it results in correct classification. This is indeed the essential learning strategy adopted by many standard *single-output* classification approaches; for instance, the margin-based SVM enforces the score difference $s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})$, the definition of the *margin*, to be non-negative, and even greater than 1.

In the structured output case, one can additionally leverage the property of the structural similarity. Suppose we have two output configurations $\mathbf{y}^a$ and $\mathbf{y}^b$ where $\mathbf{y}^a$ is closer to the true $\mathbf{y}^i$ than $\mathbf{y}^b$, implying that $\mathbf{y}^a$ agrees with $\mathbf{y}^i$ on more output variables than $\mathbf{y}^b$. Unlike the static (non-structured) cases, it makes less sense to enforce the margin (i.e., the score difference between $\mathbf{y}^i$ and $\mathbf{y}$) to be greater than 1 *uniformly* for all $\mathbf{y}$. Rather, it is more reasonable to tolerate the margin for $\mathbf{y}^a$ becomes smaller than the margin for $\mathbf{y}^b$, where the amount of tolerance is proportional to the similarity to the true $\mathbf{y}^i$.

This notion can be further extended to the cost-sensitive scenario characterized by the cost matrix C. The structural similarity is not merely the number of mismatching output variables, but rather the cost-weighted one, namely the cost-sensitive loss $l_C(\mathbf{y}^i,\mathbf{y})$. More specifically, for any output instance $\mathbf{y}$ ($\neq\mathbf{y}^i$), we force the margin $s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})$ to be greater than $l_C(\mathbf{y}^i,\mathbf{y})$. This has a crucial impact that the model score assigned to $\mathbf{y}$ (i.e., $s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})$) becomes consistent with its proximity to the true $\mathbf{y}^i$ in l_C-sense, hence effectively reflecting the underlying cost structure. By introducing the slack variables ξ's, our learning problem can formulated as

$$\begin{aligned}\min_{\xi,\boldsymbol{\theta}} \quad & \sum_{i=1}^{n}\xi_i+\frac{\lambda}{2}\|\boldsymbol{\theta}\|^2\\ \text{s.t.} \quad & s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})\geq l_C(\mathbf{y}^i,\mathbf{y})-\xi_i, \quad \forall \mathbf{y}\neq\mathbf{y}^i,\ i=1,\ldots,n,\\ & \xi_i\geq 0, \quad i=1,\ldots,n.\end{aligned} \tag{7}$$

Here, $\|\boldsymbol{\theta}\|^2$ is the regularization term which imposes smoothness of the model. $\lambda(\geq 0)$ is a constant which balances the loss term against the model smoothness. Note that (7) is similar to (6) of [9], however, unlike their definition of a margin in 0/1-loss sense, we consider a margin in cost-sensitive sense by taking into account the notion of preference encoded in the individual labels.

To solve the optimization problem, we will rewrite (7) as an equivalent but more succinct form. In the first constraints of (7), suppose that we include the case $\mathbf{y}=\mathbf{y}^i$. This simply reduces to the second constraints $\xi_i\geq 0$, provided that the cost-sensitive loss is 0 when the predicted $\mathbf{y}$ is exactly correct, i.e., $l_C(\mathbf{y}^i,\mathbf{y}^i)=0$, an assumption that can be imposed without loss of generality. Hence, we can drop the second constraints while relaxing $\mathbf{y}$ to take *any* output configuration. That is, (7) is equivalent to:

$$\begin{aligned}\min_{\xi,\boldsymbol{\theta}} \quad & \sum_{i=1}^{n}\xi_i+\frac{\lambda}{2}\|\boldsymbol{\theta}\|^2\\ \text{s.t.} \quad & s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})\geq l_C(\mathbf{y}^i,\mathbf{y})-\xi_i, \quad \forall \mathbf{y}\in\mathcal{Y},\ i=1,\ldots,n,\end{aligned} \tag{8}$$

where $\mathcal{Y}$ is a set of all possible output configurations.

Although (8) is an instance of quadratic programming (like SVM), the difficulty of the optimization lies in the exponential number ($|\mathcal{Y}|=S^{|V|}$) of constraints. To handle this issue, first note that the exponentially many inequality constraints in (8) can be equivalently rephrased as

$$s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-\max_{\mathbf{y}\in\mathcal{Y}}\{s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})+l_C(\mathbf{y}^i,\mathbf{y})\}\geq-\xi_i, \quad i=1,\ldots,n. \tag{9}$$

Although we now have a single constraint using the max function, (9) is in general non-differentiable, making the optimization difficult. Thus we consider the so-called soft-max approximation which amounts to replacing the max function by the *log-sum-exp* form, leading to

$$s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})-\log\sum_{\mathbf{y}\in\mathcal{Y}}\exp(s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})+l_C(\mathbf{y}^i,\mathbf{y}))\geq-\xi_i, \quad i=1,\ldots,n. \tag{10}$$

Note that (10) forms more restricted constraints than the original ones in (9) since the soft-max is an upper bound of the max function (i.e., $\log\sum_i e^{a_i}\geq\max_i a_i$).

Using (10), our large margin cost-sensitive CRF learning can be formulated as the following unconstrained optimization problem:

$$\min_{\boldsymbol{\theta}}\mathcal{L}:=\sum_{i=1}^{n}\left[\log\sum_{\mathbf{y}\in\mathcal{Y}}\exp(s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})+l_C(\mathbf{y}^i,\mathbf{y}))-s(\mathbf{x}^i,\mathbf{y}^i;\boldsymbol{\theta})\right]+\frac{\lambda}{2}\|\boldsymbol{\theta}\|^2. \tag{11}$$

[6] The size of the largest clique in the triangulated graph. Refer to [21] for further details.

The objective $\mathcal{L}$ is a convex function as the *log-sum-exp* of a linear function is convex.

Interestingly, (11) is inherently similar to the MAP learning objective of (5) with only the modification of the score function, that is, $s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta}) \rightarrow s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta}) + l_C(\mathbf{y}^i,\mathbf{y})$. This change of the score function amounts to having a *cost-augmented* node potential, namely

$$\mathbf{v}^\top \cdot \boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r) \longrightarrow \mathbf{v}^\top \cdot \boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r) + C(y_r^i,y_r), \tag{12}$$

while the edge potential remains unchanged. Furthermore, we can define the cost-augmented CRF conditional distribution as

$$P_{l_C}(\mathbf{y}|\mathbf{x}^i) \coloneqq \frac{\exp(s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta}) + l_C(\mathbf{y}^i,\mathbf{y}))}{\sum_{\mathbf{y}\in\mathcal{Y}} \exp(s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta}) + l_C(\mathbf{y}^i,\mathbf{y}))}. \tag{13}$$

Then evaluating the objective $\mathcal{L}$, especially the most complex *log-sum-exp* term, essentially reduces to computing the log-partition function of $P_{l_C}(\mathbf{y}|\mathbf{x}^i)$ (i.e., the denominator of (13)). Moreover, the gradient of the objective $(\partial\mathcal{L}/\partial\boldsymbol{\theta})$ can be readily available using the following derivation:

$$\begin{aligned}&\frac{\partial}{\partial\boldsymbol{\theta}} \log \sum_{\mathbf{y}\in\mathcal{Y}} \exp(s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta}) + l_C(\mathbf{y}^i,\mathbf{y})) \\ &\quad = \mathbb{E}_{P_{l_C}(\mathbf{y}|\mathbf{x}^i)}\left[\frac{\partial s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} + \frac{\partial l_C(\mathbf{y}^i,\mathbf{y})}{\partial\boldsymbol{\theta}}\right] = \mathbb{E}_{P_{l_C}(\mathbf{y}|\mathbf{x}^i)}\left[\frac{\partial s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right],\end{aligned} \tag{14}$$

where the score gradient $(\partial s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})/\partial\boldsymbol{\theta})$ is simply the sum of node or edge features from (4). In what follows, the inference for the modified model (13) is straightforward, merely changing the node potential as (12).

4.1. Cost-sensitive generalization error bound

In this section we study the theoretical aspect of the proposed large margin cost-sensitive learning. We provide the upper bound on the cost-sensitive generalization error, namely $\mathbb{E}_{\mathbf{x},\mathbf{y}}\mathcal{L}_C(\boldsymbol{\theta},\mathbf{x},\mathbf{y})$, where we denote by $\mathcal{L}_C(\boldsymbol{\theta},\mathbf{x},\mathbf{y})$ the per-label cost-sensitive loss given the data pair $(\mathbf{x},\mathbf{y})$, defined as

$$\mathcal{L}_C(\boldsymbol{\theta},\mathbf{x},\mathbf{y}) \coloneqq \frac{1}{|V|} l_C(\mathbf{y},\operatorname{argmax}_{\mathbf{y}'} s(\mathbf{x},\mathbf{y}';\boldsymbol{\theta})). \tag{15}$$

Note that we divide the loss by $|V|$ to have the per-label loss.

Our derivation for the generalization error bound is similar to that of Taskar et al.[7] which showed the generalization error bound for the margin-based *0/1-loss* CRF learning via the covering number theorem (Theorem 4 of [20]). In our cost-sensitive setting, we relate the generalization error to the margin-based cost-sensitive loss on the empirical (training) data $\mathcal{D} = \{(\mathbf{x}^i,\mathbf{y}^i)\}_{i=1}^n$, which can be written as (for some $\gamma \geq 0$):

$$\mathcal{L}_C^\gamma(\boldsymbol{\theta},\mathbf{x}^i,\mathbf{y}^i) \coloneqq \sup_{r:|r(\mathbf{x}^i,\mathbf{y}) - s(\mathbf{x}^i,\mathbf{y};\boldsymbol{\theta})| \leq \gamma\cdot l_C(\mathbf{y}^i,\mathbf{y}),\forall\mathbf{y}} \frac{1}{|V|} l_C\left(\mathbf{y}^i,\operatorname{argmax}_{\mathbf{y}} r(\mathbf{x}^i,\mathbf{y})\right). \tag{16}$$

Here, $r(\mathbf{x},\mathbf{y})$ is a structured predictor: Similar to the CRF score function $s(\mathbf{x},\mathbf{y};\boldsymbol{\theta})$, it assigns a score to the output $\mathbf{y}$ given $\mathbf{x}$, and the decision is made by $\operatorname{argmax}_{\mathbf{y}} r(\mathbf{x},\mathbf{y})$. However, it is not restricted to a particular parametric form, but rather an arbitrary function. Hence, $\mathcal{L}_C^\gamma(\boldsymbol{\theta},\mathbf{x}^i,\mathbf{y}^i)$ measures the worst-case performance of the predictors obtained by perturbing the underlying model $\boldsymbol{\theta}$ within the γ- margin. Also, it is not difficult to see that our cost-sensitive learning in (7) essentially aims at minimizing the empirical γ- margin cost-sensitive loss as well as the complexity term.

Theorem 1. *We let $C_{\max} = \max_{i,j} C(i,j)$ and $C_{\min} = \min_{i,j:C(i,j)>0} C(i,j)$ be the maximum and the minimum (non-zero) elements in the cost matrix, respectively. Without loss of generality we assume the cost matrix is properly scaled so that $C_{\min} = 1$. If the clique features are 2-norm bounded, i.e., $\max_{(r,s),y_r,y_s} \|\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r) + \boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)\|_2 \leq R$, then for any $\boldsymbol{\theta}$ and $\delta > 0$, there exists a constant K such that for any $\gamma > 0$ and any n-sample data $\mathcal{D}$,*

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\mathcal{L}_C(\boldsymbol{\theta},\mathbf{x},\mathbf{y}) \leq \mathbb{E}_{\mathcal{D}}\mathcal{L}_C^\gamma(\boldsymbol{\theta},\mathbf{x},\mathbf{y}) + C_{\max}\sqrt{\frac{K}{n}\left[\frac{R^2\|\boldsymbol{\theta}\|^2 q^2}{\gamma^2}\cdot\ln(n|V|qS) + \ln\frac{1}{\delta}\right]}, \tag{17}$$

with probability at least $1-\delta$, where q is the maximum degree in the output graph.

Due to the lack of space we only provide the proof sketch here. The proof is similar to that of Theorem 6.1 in [9], which makes use of the covering number error bound for linear classifiers [20]. To deal with the structured outputs, they form a multi-error-level function class, where the key idea is to partition the output space into $|V|$ different error levels (in the binary 0/1-loss case), namely $\{\mathbf{y}: l_{0/1}(\mathbf{y}^i,\mathbf{y}) = d\}$ for $d = 1,\ldots,|V|$. In what follows one can define an error metric and the γ- margin loss function, for which the covering number bound for linear classifiers can be directly applied (see [9] for details). To migrate this framework to our cost-sensitive setting, it is sufficient to extend the multi-error-level function class to account for the cost-sensitive loss. Noticing that $l_C(\mathbf{y}^i,\mathbf{y})$ is bounded by $|V|\cdot C_{\max}$, and the minimal increment in the loss is $C_{\min} = 1$, there are $|V|\cdot C_{\max}$ different error levels. Then it is fairly straightforward to derive (17) by following the remaining procedures of [9].

5. Related work

Learning the parameters of CRFs has been traditionally formulated as maximizing the penalized conditional likelihoods, a convex optimization problem that has been effectively solved by the quasi-Newton limited-memory BFGS [2] and the stochastic gradient methods [8]. Recently, the large margin learning of CRFs has emerged as an extension of SVMs to a structured output domain, which often exhibits higher prediction performance than the standard likelihood-based learning [9,10]. To circumvent the challenging issue of dealing with the exponential number of constraints, these approaches typically focus on a dual problem where the optimization is solved for a new set of dual variables, each of which is associated with each of the constraints. As such, one benefit is to establish a more explicit relationship between variables and constraints so as to have intuition for the variables in controlling the constraint feasibility (e.g., line search over a single dual variable may find a critical point when the corresponding constraint enters a feasible state from infeasible one). However, the exponential number of dual variables causes another difficulty in the optimization. For instance, [10] employed a perceptron-type algorithm which is less systematic than gradient search, and may take long time until convergence.

In [9], they formed a similar dual optimization which is solved using the so-called *mean parametrization* of the dual variables. It exploits the dependency structure encoded in the graph to reduce the number of variables to be optimized for. However, the number of coefficients in the quadratic terms becomes quadratic in the number of edges in the graph, which can be a significant bottleneck in the dual optimization. Hence the algorithm often

[7] See Theorem 6.1 and the proof in the longer version of [9], which can be found at http://www.seas.upenn.edu/~taskar/pubs/mmmn.pdf.

resorts to certain greedy algorithms such as the sequential minimal optimization (SMO).

The large margin techniques to structured output prediction problem have also been introduced in the speech recognition community. In [24], they considered the hidden Markov models (HMMs) with continuous-density likelihood models like Gaussians (or mixtures of Gaussians). They suggested the soft-max approximation similar to ours to formulate a convex learning problem. Our approach can be seen as its extension to undirected CRF models. And, more importantly, the above-mentioned approaches basically assume the 0/1 Hamming loss, hence they are unable to effectively handle the cost-sensitive scenario as our method.

Incorporating the cost-sensitive loss to a CRF learning was recently proposed by [16] in a different perspective from ours. This approach is based on the maximum entropy (ME) principle for the log-linear models. The ME principle states that the model should retain maximal uncertainty or entropy while satisfying the constraints that the model-expected potentials equal the observed potentials. To take into account a cost-sensitive loss, they place weights on the constraints, proportional to the losses associated with the misclassification types specified by the cost matrix.

Although this cost-weighted ME learning of CRFs provides an effective way to incorporate a cost-sensitive loss to the structured output prediction problem, the product form of the cost and the potential terms in the objective may cause additional problems, e.g., numerical issues which mainly originate from the cost terms serving as *exponents* of the exponentialized potentials. It also raises the question of how one can properly scale and balance the cost and the features.

On the other hand, our approach, by combining the cost and the potential terms under *summation* (as shown in (13)), avoids such cumbersome issues. More importantly, our large margin approach can enjoy potential benefit of achieving higher prediction performance than the ME approach, previously demonstrated by SVM-like margin-based approaches [17–20]. In the next section we empirically show that our approach indeed exhibits lower test errors than existing methods including [16] on several cost-sensitive structured output prediction problems.

6. Evaluation

In this section, we evaluate the performance of the proposed approach in a set of cost-sensitive structured output prediction problems for real applications including human motion estimation from silhouette videos, oceanography biome characteristics map prediction, and simultaneous image segmentation with object recognition. The CRF learning algorithms we contrast are summarized below:

- *MAP-CRF*: Standard MAP learning of CRF.
- *LM-0/1-CRF*: Large margin 0/1-loss CRF learning [9].
- *ME-CRF*: Maximum entropy cost-sensitive CRF learning of [16].
- *LM-Cost-CRF*: Large margin cost-sensitive CRF learning (proposed approach).

For gradient search in the above algorithms, we adopt the latest stochastic gradient descent algorithm [8], which often exhibits faster convergence rates than standard batch-mode quasi-Newton methods particularly for CRFs. The data sets we used in the evaluation have either 1D-sequence or 2D-lattice output structures. In the former cases we employ the forward-backward (exact) inference algorithm, while for the latter we use the loopy belief propagation (LBP) algorithm that is known to perform more reliably than other approximate inference algorithms [8].

Following the conventional modeling practice, the node/edge features for CRF models are defined as the product of measurement features and the output class indicators. More specifically, denoting the measurement feature vector at node r as $\boldsymbol{\phi}(\mathbf{x}_r)$, the node feature $\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)$ becomes

$$\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)=[\boldsymbol{\phi}(\mathbf{x}_r)^\top\cdot I(y_r=1),\ldots,\boldsymbol{\phi}(\mathbf{x}_r)^\top\cdot I(y_r=S)]^\top. \tag{18}$$

Hence the k-th block ($k=1,\ldots,S$) of $\boldsymbol{\Psi}_r^{(V)}(\mathbf{x},y_r)$ is $\boldsymbol{\phi}(\mathbf{x}_r)$ if $y_r=k$, and the $\mathbf{0}$-vector otherwise. The edge feature vector at $e=(r,s)$ is defined as the absolute difference between measurement features at adjoining nodes r and s. Similar to the way of forming the node features, the edge feature vector $\boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)$ consists of S^2 blocks, where the (k,l)-th block ($k,l=1,\ldots,S$) of $\boldsymbol{\Psi}_e^{(E)}(\mathbf{x},y_r,y_s)$ is set as: $|\boldsymbol{\phi}(\mathbf{x}_r)-\boldsymbol{\phi}(\mathbf{x}_s)|\cdot I(y_r=k\&y_s=l)$. In the following, we describe the data sets and the features in greater details.

6.1. Human walking motions

We evaluate the performance of the proposed method on the task of 3D body pose estimation from video sequences. The CMU motion capture data set (http://mocap.cs.cmu.edu/) provides the ground-truth body poses (3D joint angles), which makes it possible to compare competing methods quantitatively. Here we consider a walking motion. We collect 10 walking motions from one subject to perform leave-1-out validation. Sequences are about 150-frame long, containing 1 or 2 walking cycles. The measurement is a 10-dim Alt-Moment [25] feature vector extracted from a silhouette image. The images are taken by a single camera at a fixed view.

Among the original 59 joint angles at 31 articulation points, we are particularly interested in predicting two angles: (i) Angle #03 = the angle around the femur at the left leg, and (ii) Angle #38 = the angle around the humerus at the left arm/elbow. These angles have critical impacts on the human walking style, and exhibit the largest variability during the walking motion. For each of two angles, we discretize the real-valued angle into $S=10$ ordinal-scale categories ($y_r\in\{1,2,\ldots,10\}$ for each time frame r) which are treated as the output classes in the sequence-structured output prediction problem.

As the discretized output is in the ordinal scale that reflects the original real scale, it naturally favors the cost-sensitive learning framework, and we set the cost matrix to be the absolute difference[8] in the discretized outputs, namely $C_{ij}=|i-j|$ for $i,j=1,\ldots,10$. In this way we can mimic the original scale on real line: if i and j are closer to each other, it incurs less cost, and vice versa.

Since we have the ordinal scale outputs, another possible approach to be considered is the *ordinal regression* that has long been studied in the machine learning community, often named *ranking* or *preference learning*. However, most ordinal regression algorithms are static and unable to handle structured outputs in a principled manner. For empirical comparison with ordinal regression methods, we adopt one of the most recent approaches, called the support vector ordinal regression (SVOR) of [26] which optimizes multiple thresholds to define parallel discriminant hyperplanes for the ordinal scales. In SVOR, we treat each output variable comprising the structured output individually and independently. The features for the SVOR are the same to the node features used in CRFs.

[8] More generally, the cost can be defined as $C_{ij}=\rho(|i-j|)$ for some monotonically increasing function $\rho(\cdot)$.

Table 1
Test errors in human walking motion data set.

Methods	(a) Angle #03		(b) Angle #38	
	0/1 Loss	Cost-sensitive Loss	0/1 Loss	Cost-sensitive Loss
MAP-CRF	0.7468 ±0.0405	1.6490 ±0.1691	0.7400 ±0.0535	1.2014 ±0.2918
LM-0/1-CRF	0.6754 ±0.0532	1.1533 ±0.1872	0.6646 ±0.0562	1.0430 ±0.1459
ME-CRF	0.6227 ±0.0537	0.8792 ±0.1109	0.5733 ±0.0450	0.7349 ±0.1174
LM-Cost-CRF	**0.5048 ±0.0545**	**0.6331 ±0.1251**	**0.5122 ±0.0396**	**0.5782 ±0.0748**
Static SVOR	0.6566 ±0.0453	1.0125 ±0.1864	0.6513 ±0.0515	0.9918 ±0.1371

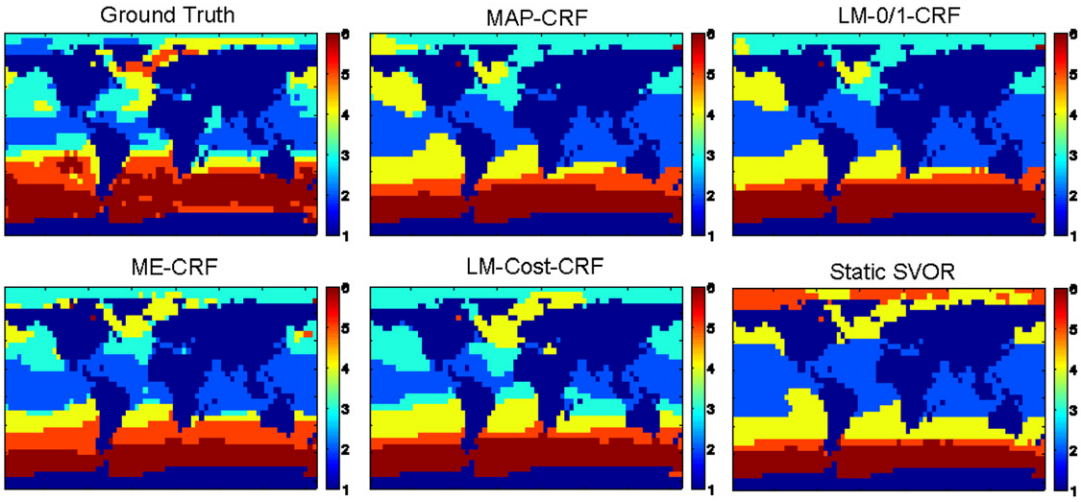

Fig. 2. Output prediction for the global oceanography map (2D lattice) on the density characteristics. The output labels ({1,…,6}) are depicted as colors: warmer (red) color represents a higher value while cooler (blue) lower. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The prediction errors are shown in Table 1, where we present both the 0/1 loss and the cost-sensitive loss. Our LM-Cost-CRF yields the lowest test errors for both of two angles, outperforming the maximum entropy based cost-sensitive CRF learning (ME-CRF) as well as the static ordinal regression method (Static SVOR). This success suggests a promising direction in tackling the structured output *regression* problems. We have shown that the difficult density integrability issues in dealing with real-valued outputs can be effectively circumvented through the proposed cost-sensitive CRF learning.

6.2. Oceanography biome characteristics prediction

We next evaluate the structured output cost-sensitive learning methods on the oceanographic biome data set obtained from the World Ocean Atlas 2005 [27]. The data set contains five oceanographic biome characteristics (features): temperature (°C), salinity, 10–200 m density difference, phosphate (μmol/L), and dissolved oxygen (ml/L). These five features are measured at 5°-by-5° granularity, which yields two-dimensional maps of size $((latitude=36) \times (longitude=72))$, one for each feature. Fig. 2 ("Ground Truth") shows an example map for the density characteristics. We consider five different 2D structured output prediction problems by taking one feature as output while treating the rest as input.

Each feature is normalized to take a real number between [0,1]. When selected as output, the feature is discretized into $S=6$ ordinal scale categories. For the input, we augment the four held-out features with the global position information (longitude and latitude) and the information indicating whether a site is land or ocean, thus forming seven-dimensional input feature vector. We collect four seasonal data, where we take maps of winter, spring, and autumn as a training set, and the map of summer as a test set. As the discretized output takes the ordinal scale, the cost-sensitive loss is again chosen as the absolute difference in classes, $C_{ij}=|i-j|$.

The test prediction errors are shown in Table 2, while the predicted map for the density output is shown in Fig. 2. As shown, our LM-Cost-CRF results in the best performance consistently for all five prediction problems. It is also interesting to note that for these 2D lattice structured output cases, the performance of the static approach (SVOR) is even worse than the standard 0/1-loss CRF learning (MAP-CRF). It may imply that capturing the dependency structure of complex underlying statistical processes like the two-dimensional interaction on the lattice can often be more crucial than merely considering the cost-sensitive loss. However, within the same CRF modeling capacity, the cost-sensitive learning is highly beneficial.

6.3. Simultaneous image segmentation and object recognition in hierarchy

We next test our cost-sensitive learning approach on the task of simultaneous image segmentation and object recognition, an

Table 2
Test errors in oceanographic features prediction.

Methods	(a) Density 0/1 Loss (cost loss)	(b) Temp. 0/1 Loss (cost loss)	(c) Oxygen 0/1 Loss (cost loss)	(d) Phos. 0/1 Loss (cost loss)	(e) Salinity 0/1 Loss (cost loss)
MAP-CRF	0.4697 (0.5466)	0.3927 (0.4365)	0.3465 (0.3734)	0.4415 (0.5441)	0.4609 (0.6961)
LM-0/1-CRF	0.4565 (0.5428)	0.3890 (0.4259)	0.3252 (0.3408)	0.4353 (0.5410)	0.4578 (0.6760)
ME-CRF	0.4171 (0.4565)	0.3577 (0.3784)	0.3008 (0.3189)	0.4303 (0.5053)	0.4365 (0.6098)
LM-cost-CRF	**0.3621 (0.3877)**	**0.3002 (0.3121)**	**0.2402 (0.2433)**	**0.3946 (0.4559)**	**0.3946 (0.4690)**
Static SVOR	0.5641 (0.7874)	0.5941 (0.6429)	0.3208 (0.3333)	0.4878 (0.6904)	0.5197 (0.6135)

The cost-sensitive errors are shown in parentheses.

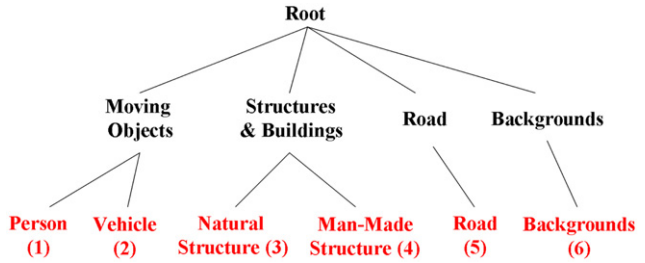

Fig. 3. Class hierarchy for the Cambridge video data set. The leaf nodes (in red) are the actual classes with class numbers in parentheses. The internal nodes (in black) are the meta nodes which can be seen as the super classes of the real classes. The misclassification loss is defined as the length of the shortest path in the tree. For instance, $C_{1,2} = 2$, $C_{2,3} = 4$, and $C_{4,5} = 4$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.).

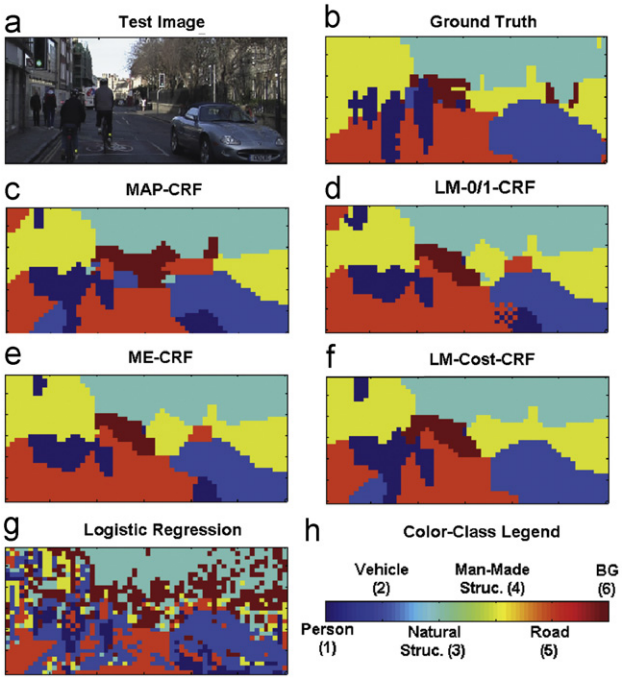

Fig. 4. Output prediction for simultaneous image segmentation and object recognition problem (2D lattice) on the Cambridge video data set. The output labels are depicted by colors according to the correspondence in (h). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.).

important problem in computer vision. We use the data set from the Cambridge driving labeled video database[9] (CamVid) [28,29], which contains video frames with per-pixel semantic segmentation and multiclass object labels marked by human. The videos were captured from the perspective of a driving automobile providing high-quality images recorded at 30 fps rates.

We take a video from the database that consists of 101 frames, where we select the first 60 frames as training set and the last 20 frames as the testing set. The size of the original image frames is (720 $\times$ 960), but we crop them to (300 $\times$ 600) images so that they contain only the objects of interest. Fig. 4(a) shows one of the test images. In the database there are overall 32 semantic object classes, while we only consider a subset comprised of six classes (class identity in parenthesis): (1) `Person`, (2) `Vehicle`, (3) `Natural Structure`, (4) `Man-Made Structure`, (5) `Road`, (6) `Backgrounds`. Although the database provides pixel-level labels, we consider a patch (site) of size 10-pixel by 10-pixel as an atomic block so as to be robust to spike-type noise in the images. Hence we have a 2D lattice structure of size (30 $\times$ 60). For each site we assign to it a single label that takes the majority within the site. Fig. 4(b) depicts the ground-truth labels corresponding to the test image in Fig. 4(a), where the six classes are encoded as colors as shown in Fig. 4(h).

Even though the six classes can be treated independently with the 0/1 misclassification loss, it is more reasonable to take into account the semantics of the classes and the relationship among them. One possible way to effectively represent the semantics is to build a class hierarchy, where classes can be grouped into higher-level meta classes in a tree structure. For this problem, we

[9] Available from http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/.

Table 3
Average test errors (per site) in the Cambridge video data set.

Methods	0/1 Loss	Cost-sensitive loss
MAP-CRF	0.2344	0.8544
LM-0/1-CRF	0.2100	0.7488
ME-CRF	0.1839	0.6600
LM-cost-CRF	**0.1567**	**0.5700**
Logistic regression	0.4761	1.7222

construct the class hierarchy shown in Fig. 3. We place the actual classes at the bottom level of the tree, and introduce four meta classes that subsume relevant sub-classes (e.g., the meta class `Moving Objects` includes the `Person` and the `Vehicle` classes as its instances). The meta classes are further grouped into the `Root` class.

In order to incorporate this hierarchical class structure into the prediction model, we utilize the cost-sensitive loss that conforms to the class hierarchy. We define the misclassification loss for predicting class i to j as the length (i.e., the number of edges) of the shortest path between nodes i and j in the tree. For instance, $C_{1,2} = 2$, $C_{2,3} = 4$, and $C_{4,5} = 4$. In this way the prediction errors related to classes within the same meta class take less costs than misclassification across different meta classes.

For the measurement features, we adopt the features used in [8]. At each site (i.e., node), we extract 14-dimensional multi-scale edge-orientation histogram (EOH) features that can capture the general statistical properties of the objects in images over spatially neighboring sites. See [30] for further details on the features. As before, the edge features are set to the absolute difference of the features at the adjoining nodes.

In addition to contrasting our approach with other CRF estimators, we also present the performance of a fairly basic logistic regression model as a baseline comparison. The logistic regression is a static classifier that treats each of the output variables independently, thus discarding the dependency among the adjoining sites (i.e., output variables). As we use the same node features for the logistic regression, one can view it as the CRF model with no edge features included.

The average (per-site normalized) test errors of the competing approaches are shown in Table 3. The cost-sensitive learning algorithms (ME-CRF and LM-Cost-CRF) significantly outperform the 0/1 loss based methods in terms of both the 0/1 loss and the hierarchical loss. Moreover, our LM-Cost-CRF still achieves higher prediction accuracy than the ME-CRF. Fig. 4 also depicts the predicted labels, where our approach appears to be the closest to the ground truth. The logistic regression in Fig. 4(g) has many salt-and-pepper type misclassification, which is mainly resulted from its individual and independent treatment of the output variables, unable to capture the (spatially) local smoothness property of natural images.

7. Conclusion

We introduced a novel large margin learning method for CRFs for structured output prediction in cost-sensitive settings. The proposed approach aims at optimizing the class-margin objective by incorporating the cost-sensitive loss into the large margin learning framework. We have shown, through a comprehensive set of evaluations, that the large margin cost-sensitive learning of CRFs can yield superior performance to the standard likelihood-based learning as well as the recent maximum entropy cost-sensitive approach. The proposed method is also computationally efficient as it only requires slight modification of the potential functions in the original CRF models. In this paper, although we have focused on 1D sequence and 2D lattice output structures, it should be straightforward to extend it to models with more complex interactions. Furthermore, not restricted to the parameter learning, we also plan to consider how the proposed approach can be extended to the learning of model structures in the cost-sensitive framework.

References

[1] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: International Conference on Machine Learning, Williamstown, MA, USA, 2001.
[2] F. Sha, F. Pereira, Shallow parsing with conditional random fields, in: Proc. of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada, 2003, pp. 134–141.
[3] S. Kumar, M. Hebert, Discriminative random fields, International Journal of Computer Vision 68 (2006) 179–201.
[4] R. McDonald, F. Pereira, Identifying gene and protein mentions in text using conditional random fields, BMC Bioinformatics 6 (Suppl. 1) (2005).
[5] X. He, R. Zemel, M.A. Carreira-Perpinan, Multiscale conditional random fields for image labelling, in: IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 2004.
[6] J. Zhang, S. Gong, Action categorization with modified hidden conditional random field, Pattern Recognition 43 (1) (2010) 197–203.
[7] M. Kim, V. Pavlovic, Conditional state space models for discriminative motion estimation, in: IEEE International Conference on Computer Vision, Rio de Janeiro, Brazil, 2007.
[8] S. Vishwanathan, N. Schraudolph, M. Schmidt, K. Murphy, Accelerated training of conditional random fields with stochastic meta-descent, in: International Conference on Machine Learning, Pittsburgh, PA, USA, 2006.
[9] B. Taskar, C. Guestrin, D. Koller, Max-margin Markov networks, in: Neural Information Processing Systems, Vancouver, BC, Canada, 2003.
[10] Y. Altun, I. Tsochantaridis, T, Hofmann, Hidden Markov support vector machines, in: International Conference on Machine Learning, Washington, DC, USA, 2003.
[11] C. Elkan, The foundations of cost-sensitive learning, in: Proceedings of International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, 2001, pp. 973–978.
[12] P. Domingos, MetaCost: a general method for making classifiers cost-sensitive, in: Proceedings of International Conference on Knowledge Discovery and Data Mining, ACM Press, 1999, pp. 155–164.
[13] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: IEEE International Conference on Data Mining, Melbourne, Florida, USA, 2003, pp. 435–442.
[14] Y. Sun, M.S. Kamel, A.K. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognition 40 (12) (2007) 3358–3378.
[15] F. Xia, Y. wu Yang, L. Zhou, F. Li, M. Cai, D.D. Zeng, A closed-form reduction of multi-class cost-sensitive learning to weighted multi-class learning, Pattern Recognition 42 (7) (2009) 1572–1581.
[16] P. Sen, L. Getoor, Cost-sensitive learning with conditional Markov networks, Data Mining and Knowledge Discovery 17 (2) (2008) 136–163.
[17] P.L. Bartlett, The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network, IEEE Transactions on Information Theory 44 (2) (1998) 525–536.
[18] J. Shawe-Taylor, P. Bartlett, R. Williamson, M. Anthony, A framework for structural risk minimisation, in: Proceedings of the Ninth Annual Conference on Computational Learning Theory, Desenzano sul Garda, Italy, 1996.
[19] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, 1995.
[20] T. Zhang, Covering number bounds of certain regularized linear function classes, Journal of Machine Learning Research 2 (2002) 527–550.
[21] M.I. Jordan, Graphical models, Statistical Science 19 (2004) 140–155.
[22] J. Yedidia, W. Freeman, Y. Weiss, Understanding belief propagation and its generalizations, in: Exploring Artificial Intelligence in the New Millennium, Science & Technology Books, 2003, pp. 239–236 (Chapter 8).
[23] K. Crammer, Y. Singer, On the algorithmic implementation of multiclass kernel based vector machines, Journal of Machine Learning Research 2 (5) (2001) 265–292.
[24] F. Sha, L.K. Saul, Large margin hidden Markov models for automatic speech recognition, Neural Information Processing Systems, Vancouver, BC, Canada, 2007.
[25] T.-P. Tian, R. Li, S. Sclaroff, Articulated pose estimation in a learned smooth space of feasible solutions, in: Proceedings of IEEE Workshop in Computer Vision and Pattern Recognition, 2005.
[26] W. Chu, S.S. Keerthi, New approaches to support vector ordinal regression, International Conference on Machine Learning, Bonn, Germany, 2005.
[27] R.A. Locarnini, A.V. Mishonov, J.I. Antonov, T.P. Boyer, H.E. Garcia, World ocean atlas 2005, in: S. Levitus (Ed.), NOAA Atlas NESDIS, US Government Printing Office, Washington, DC, pp. 61–64.
[28] G.J. Brostow, J. Fauqueur, R. Cipolla, Semantic object classes in video: a high-definition ground truth database, Pattern Recognition Letters 30 (2) (2008) 88–97.

[29] G.J. Brostow, J. Shotton, J. Fauqueur, R. Cipolla, Segmentation and recognition using structure from motion point clouds, in: European Conference on Computer Vision, 2008, pp. 44–57.

[30] S. Kumar, M. Hebert, Man-made structure detection in natural images using a causal multiscale random field, IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, USA, 2003.

About the Author—MINYOUNG KIM received the BS and MS degrees both in Computer Science and Engineering in Seoul National University, South Korea. He earned the Ph.D. degree in Computer Science from Rutgers University in 2008. He is currently a postdoctoral researcher at the Robotics Institute of Carnegie Mellon University. His primary research interest is machine learning and computer vision. His research focus includes graphical models, motion estimation/tracking, discriminative models/learning, kernel methods, and dimensionality reduction.