Max-margin Learning for Lower Linear Envelope Potentials in Binary Markov Random Fields

Stephen Gould STEPHEN.GOULD@ANU.EDU.AU
Research School of Computer Science, Australian National University, ACT 0200, Australia

Abstract

The standard approach to max-margin parameter learning for Markov random fields (MRFs) involves incrementally adding the most violated constraints during each iteration of the algorithm. This requires exact MAP inference, which is intractable for many classes of MRF. In this paper, we propose an exact MAP inference algorithm for binary MRFs containing a class of higher-order models, known as *lower linear envelope potentials*. Our algorithm is polynomial in the number of variables and number of linear envelope functions. With tractable inference in hand, we show how the parameters and corresponding feature vectors can be represented in a max-margin framework for efficiently learning lower linear envelope potentials.

1. Introduction

Considerable advances have been made in the past several years in applying the max-margin principle to the task of learning the parameters of a Markov random field (MRF) for structured prediction (Taskar et al., 2005; Tsochantaridis et al., 2004). The standard approach is to learn model parameters by constraining the max-margin objective to favour the ground-truth assignment over all other joint assignments to the variables. Since the set of all possible joint assignments can be prohibitively large (exponential in the number of the variables), constraints are introduced incrementally by finding the most violated ones at each iteration (with respect to the current parameter settings).

Despite these advances, learning the parameters of an MRF remains a notoriously challenging task due to the difficulty of finding the most violated constraints, which requires performing exact MAP inference. Except in a few special cases, such as tree-structured graphs or binary pairwise MRFs with submodular potentials, inference is intractable and the max-margin framework cannot be applied. When substituting approximate inference routines to generate constraints, the max-margin framework is not guaranteed to learn the optimal parameters and often performs poorly (Finley and Joachims, 2008).

Recently, models with structured higher-order terms have become of interest to the machine learning community with many applications in computer vision, particularly for encoding consistency constraints over large sets of pixels, e.g., (Lempitsky et al., 2009; Nowozin and Lampert, 2009; Rother et al., 2009). A rich class of higher-order models, known as *lower linear envelope potentials*, which includes the generalized Potts model and its variants (Kohli et al., 2007), was proposed by Kohli and Kumar (2010). While efficient approximate inference algorithms exist for these models, parameter learning remains an unsolved problem.

In this paper we focus on learning the lower linear envelope parameters for binary MRFs. We propose an exact MAP inference algorithm for these models that is polynomial in the number of variables and number of linear envelope functions. This opens the way for max-margin parameter learning. However, to encode the max-margin constraints we require a linear relationship between model parameters and the features that encode each problem instance.

Our key insight is that we can represent the lower linear envelope in two different ways: the first—as the minimum over a set of linear functions—is tractable for MAP inference (i.e., constraint generation), and the second—a sample-based representation with linear constraints—is tractable for max-margin learning. By switching between these representations we can learn model parameters efficiently.

We evaluate our approach on synthetic data as well as a variant of the real-world "GrabCut" image segmentation problem (Rother et al., 2004).

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

2. Related Work

Our work focuses on a class of higher-order potentials known as lower linear envelope potentials, which can be used to represent arbitrary concave functions over the number of variables (in a clique) taking a given assignment. Kohli and Kumar (2010) show how such potentials can be represented by introducing a multi-valued auxiliary variable to select each linear function in the envelope. In principle, the optimal assignment can be found by jointly minimizing the energy function over the original variables and this auxiliary variable. However, this is non-trivial, in general, and Kohli and Kumar (2010) only show how the resulting energy function can be approximately optimized.

Earlier research, on MRFs with restricted variants of the lower linear envelope potential, showed how exact inference can be performed in the binary case. Kohli et al. (2007) introduced the P^n-model for encoding consistency constraints. This was later extended to the robust P^n-model (a lower linear envelope potential with only two terms per label—one increasing and one constant) by Kohli et al. (2008) who also describe an efficient move-making inference algorithm based on graph-cuts (Boykov and Kolmogorov, 2004; Boykov et al., 1999). Multiple robust P^n-models can be added to form a non-decreasing concave envelope. Ladicky et al. (2009) used this model for improving the quality of multi-class image labeling.

In contrast to these works, we propose an algorithm for exactly optimizing binary MRFs with *arbitrary* lower linear envelope potentials. Our work, and the previous approaches, are related to a number of methods that transform higher-order or multi-label energy functions into quadratic pseudo-Boolean functions (e.g., (Ishikawa; 2009; Rother et al., 2009)). These functions have been studied extensively in the operations research literature (for a survey see Boros and Hammer (2002)). Under certain conditions, the resulting pseudo-Boolean function can be minimized exactly by finding the minimum-cut in a suitably constructed graph (Freedman and Drineas, 2005; Hammer, 1965). Our work makes use of this result.

Our max-margin learning framework is based on the approaches introduced by Tsochantaridis et al. (2004; 2005) and Taskar et al. (2005), which have been successfully applied within many application domains (see Joachims et al. (2009) for a recent survey and the "1-slack" reformulation). Szummer et al. (2008) showed how this framework could be adapted to learn pairwise MRF parameters using graph-cuts for inference. Unlike their approach, our method applies to models with higher-order terms.

3. Lower Linear Envelope MRFs

We begin by providing a brief overview of higher-order Markov random fields (MRFs). We then introduce the lower linear envelope potential and show how to perform exact inference in models that contain these potentials. In the next section we will discuss learning the parameters of these models.

Higher-order MRFs. The *energy function* for a higher-order MRF over discrete random variables $\boldsymbol{y} = \{y_1, \ldots, y_n\}$ can be written as:

$$E(\boldsymbol{y}) = \underbrace{\sum_i \psi_i^U(y_i)}_{\text{unary}} + \underbrace{\sum_{ij} \psi_{ij}^P(y_i, y_j)}_{\text{pairwise}} + \underbrace{\sum_c \psi_c^H(\boldsymbol{y}_c)}_{\text{higher-order}} \quad (1)$$

where the *potential* functions ψ_i^U, ψ_{ij}^P and ψ_c^H encode preferences for unary, pairwise and k-ary variable assignments, respectively. The pairwise terms, ψ_{ij}^P, also called *edge potentials*, are usually only defined over a sparse subset of possible variable pairs (y_i, y_j). The latter terms, ψ_c^H, are defined over arbitrary subsets of variables (or *cliques*), $\boldsymbol{y}_c = \{y_i : i \in \mathcal{C}_c\}$ where $\mathcal{C}_c \subseteq \{1, \ldots, n\}$ is a subset of variable indices, and are known as *higher-order potentials*.

In this paper, we will be concerned with inference and learning of higher-order binary MRFs (i.e., $y_i \in \{0, 1\}$) with lower linear envelope potentials. A lower linear envelope potential[1] over a subset of binary variables $\boldsymbol{y}_c$ is a piecewise linear function defined as the minimum over a set of K linear functions

$$\psi_c^H(\boldsymbol{y}_c) \triangleq \min_{k=1,\ldots,K} \left\{ a_k \sum_{i \in \mathcal{C}} y_i + b_k \right\} \quad (2)$$

where $(a_k, b_k) \in \mathbb{R}^2$ are the linear function parameters. Figure 1 shows an example lower envelope for three linear functions. Kohli and Kumar (2010) showed that this representation can encode arbitrary concave functions of $x = \sum_{i \in \mathcal{C}} y_i$ given sufficiently many linear functions. The parameterization, however, is not unique.

Definition 3.1 (Active). *We say that the k-th linear function is* active *with respect to an assignment $\boldsymbol{y}_c$ if $\psi_c^H(\boldsymbol{y}_c) = a_k \sum_{i \in \mathcal{C}} y_i + b_k$.*

Clearly, if a linear function is never active it can be removed from the potential without changing the energy function.

[1] For brevity, in this paper we set the per-variable weight w_i that appears in Kohli et al. (2008) and Kohli and Kumar (2010) to one, but note that arbitrary non-negative per-variable weights can be easily added.

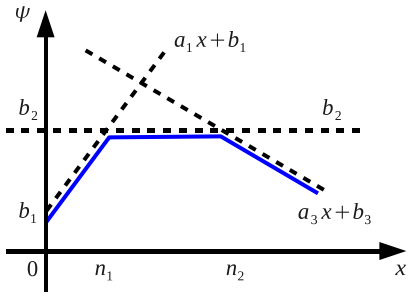


Figure 1. Example lower linear envelope $\psi_c^H(\boldsymbol{y}_c)$ (shown solid) with three terms (dashed) as a function of $x = \sum_{i\in\mathcal{C}} y_i$. When $x \leq n_1$ the first linear function is active, when $n_1 < x \leq n_2$ the second linear function is active, otherwise the third linear function is active.

Definition 3.2 (Redundant). *We say that the k-th linear function is* redundant *if it is not active for any assignment to* $\boldsymbol{y}_c$.

Although not strictly necessary, in the following, we assume that our potentials do not contain redundant linear functions. Furthermore, we assume that the parameters $\{(a_k, b_k)\}_{k=1}^K$ are sorted in decreasing order of a_k. Clearly, this implies that $a_k > a_{k+1}$ and $b_k < b_{k+1}$.

Exact Inference. The goal of inference is to find an energy-minimizing assignment $\boldsymbol{y}^\star \in \operatorname{argmin}_{\boldsymbol{y}} E(\boldsymbol{y})$. To do this, we follow the approach of a number of works that address the problem of inference in certain classes of higher-order MRFs by transforming the inference problem to that of minimizing a quadratic pseudo-Boolean function, i.e., pairwise MRF (e.g., (Boros and Hammer, 2002; Freedman and Drineas, 2005; Ishikawa, 2009)). For example, Kohli et al. (2008) showed that exact inference can be performed when the potential is a concave piecewise linear function of at most three terms (one increasing, one constant, and one decreasing). We now extend this result to arbitrary many terms.

Consider, again the lower linear envelope potential represented by Equation 2. Introducing $K-1$ auxiliary binary variables $\boldsymbol{z} = (z_1, \ldots, z_{K-1})$, we define the quadratic pseudo-Boolean function

$$E^c(\boldsymbol{y}_c, \boldsymbol{z}) = a_1 \sum_{i\in\mathcal{C}} y_i + b_1 + \sum_{k=1}^{K-1} z_k \left((a_{k+1} - a_k) \sum_{i\in\mathcal{C}} y_i + b_{k+1} - b_k \right) \quad (3)$$

The advantage of this formulation is that minimizing over $\boldsymbol{z}$, subject to some constraints, selects (one of) the active function(s) from $\psi_c^H(\boldsymbol{y})$ as we will now show.

Proposition 3.3. *Minimizing the function* $E^c(\boldsymbol{y}_c, \boldsymbol{z})$ *over* $\boldsymbol{z}$ *subject to* $z_{k+1} \leq z_k$ *for all* k *is equivalent to* $\min_{k=1,\ldots,K} \left\{ a_k \sum_{i\in\mathcal{C}} y_i + b_k \right\}$, *i.e.,* $\psi_c^H(\boldsymbol{y}_c) = \min_{\boldsymbol{z}: z_{k+1}\leq z_k} E^c(\boldsymbol{y}_c, \boldsymbol{z})$.

Proof. The constraints ensure that $\boldsymbol{z}$ takes the form of a vector of all ones followed by all zeros. There are K such vectors and for $k = \mathbf{1}^T\boldsymbol{z} + 1$ we have $E^c(\boldsymbol{y}_c, \boldsymbol{z}) = a_k \sum_{i\in\mathcal{C}} y_i + b_k$. Therefore, minimizing over $\boldsymbol{z}$ is the same as minimizing over $k \in \{1, \ldots, K\}$. □

The constraints on $\boldsymbol{z}$ can be enforced by adding $Mz_{k+1}(1 - z_k)$ for $k = 1, \ldots, K-2$ to the energy function with M sufficiently large.[2] Rewriting the quadratic pseudo-Boolean function of Equation 3 in *posiform* (Boros and Hammer, 2002) and adding the constraints on $\boldsymbol{z}$, we have

$$\begin{aligned}\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z}) &= b_1 - |\mathcal{C}|(a_1 - a_K) + \sum_{i\in\mathcal{C}} a_1 y_i \\ &+ \sum_{k=1}^{K-1} (b_{k+1} - b_k)\, z_k + \sum_{k=1}^{K-1} |\mathcal{C}|\, (a_k - a_{k+1})\, \bar{z}_k \\ &+ \sum_{k=1}^{K-1} \sum_{i\in\mathcal{C}} (a_k - a_{k+1})\, \bar{y}_i z_k + \sum_{k=1}^{K-2} M z_{k+1} \bar{z}_k \quad (4)\end{aligned}$$

where $\bar{z}_k = 1 - z_k$ and, likewise, $\bar{y}_i = 1 - y_i$, and all coefficients (apart from the constant term) are positive.

Importantly, $\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z})$ is a *submodular* energy function, which allows us to perform efficient inference.

Definition 3.4 (Submodularity). *A pseudo-Boolean function* $f : \{0,1\}^n \to \mathbb{R}$ *is called* submodular *if* $f(\boldsymbol{u}) + f(\boldsymbol{v}) \geq f(\boldsymbol{u} \vee \boldsymbol{v}) + f(\boldsymbol{u} \wedge \boldsymbol{v})$ *for all* $\boldsymbol{u}, \boldsymbol{v} \in \{0,1\}^n$.

Proposition 3.5. *The energy function* $\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z})$ *defined by Equation 4 is submodular.*

Proof. Follows from the fact that all the bi-linear terms in Equation 4 are of the form $\lambda\bar{u}v$ with $\lambda \geq 0$. See Boros and Hammer (2002). □

It is well known that submodular pairwise energy functions can be minimized exactly in time polynomial in the number of variables by finding the minimum-*st*-cut on a suitably constructed graph (Hammer, 1965; Kolmogorov and Zabih, 2004). We illustrate one possible construction for $\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z})$ in Figure 2.

Using this fact, we can show that an energy function containing arbitrary lower linear envelope potentials can be minimized in polynomial time.

[2] In practice we can set $M = \sum_k |a_k| n + |b_k|$ to ensure that each constraint is satisfied.

Theorem 3.6. *For binary variables $\boldsymbol{y} \in \{0,1\}^n$, let $E^0(\boldsymbol{y})$ be a submodular energy function, and let*

$$E(\boldsymbol{y}) = E^0(\boldsymbol{y}) + \sum_c \psi_c^H(\boldsymbol{y}_c),$$

where $\psi_c^H(\boldsymbol{y}_c)$ are arbitrary lower linear envelope higher-order potentials. Then $E(\boldsymbol{y})$ can be minimized in time polynomial in the number of variables n and total number of linear envelope functions.

Proof. By Proposition 3.3 we have $\operatorname{argmin}_{\boldsymbol{y}} E(\boldsymbol{y}) = \operatorname{argmin}_{\boldsymbol{y}}(E^0(\boldsymbol{y}) + \sum_c \min_{\boldsymbol{z}_c} \widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z}_c))$. By Proposition 3.5 we have that the $\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z}_c)$ are submodular. The sum of submodular energy functions is submodular. Each higher-order term adds $K-1$ auxiliary variables so the total number of variables in the augmented energy function is less than n plus the total number of linear functions. □

Relationship to Binary MRFs. As an aside, we note that $\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z}_c)$ is just a pairwise binary MRF. Evidently, we can express Equation 4 as

$$\widetilde{E}^c(\boldsymbol{y}_c, \boldsymbol{z}) = \text{const.} + \sum_{i \in \mathcal{C}} \psi_i^Y(y_i) + \sum_{k=1}^{K-1} \psi_k^Z(z_k) + \sum_{(i,k)} \psi_{ik}^P(y_i, z_k) + \sum_{(k,k+1)} \psi_{k,k+1}^C(z_k, z_{k+1}) \quad (5)$$

where, for example, $\psi_k^Z(z_k) = (b_{k+1} - b_k)$ if $z_k = 1$ and $|\mathcal{C}|\,(a_k - a_{k+1})$ otherwise. For brevity, we omit details of the remaining potential functions, which can be trivially constructed by considering the correspondence between the two forms.

4. Learning the Lower Linear Envelope

We now show how the max-margin framework can be used to learn parameters of our lower linear envelope potentials. For simplicity of exposition we consider a single higher-order term. The extension to multiple higher-order terms is straightforward.

We begin by reviewing a variant of the max-margin framework introduced by Tsochantaridis et al. (2004) and Taskar et al. (2005). We then show how an alternative representation of the lower linear envelope can be learned using the framework. Finally, we discuss some practical issues such as invariance to clique size.

Max-margin Learning. Given an energy function $E(\boldsymbol{y}; \boldsymbol{\theta}) = \boldsymbol{\theta}^T \phi(\boldsymbol{y})$ parameterized as linear combination of features $\phi(\boldsymbol{y}) \in \mathbb{R}^m$ and weights $\boldsymbol{\theta} \in \mathbb{R}^m$, and a set of T training examples $\{\boldsymbol{y}_t\}_{t=1}^T$ the max-margin framework is a principled approach to learning the weights of the model.

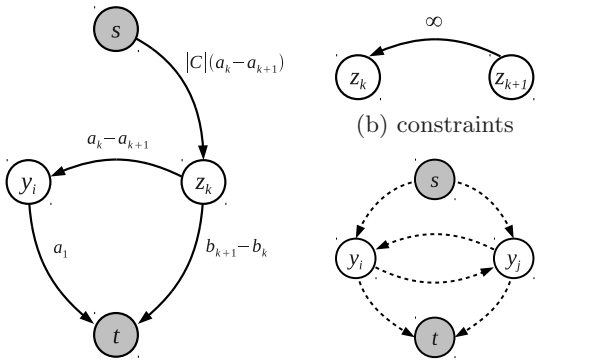


Figure 2. Construction of an *st*-graph for minimizing energy functions with arbitrary lower linear envelope potentials. With slight abuse of notation, we use the variables to denote nodes in our graph. For each lower linear envelope potential edges are added as follows: for each $i \in \mathcal{C}$, add an edge from y_i to t with weight a_1; for each $i \in \mathcal{C}$ and $k = 1, \ldots, K-1$, add an edge from z_k to y_i with weight $a_k - a_{k+1}$; for $k = 1, \ldots, K-1$, add an edge from s to z_k of with weight $|\mathcal{C}|(a_k - a_{k+1})$ and edge from z_k to t with weight $b_{k+1} - b_k$; and for $k = 1, \ldots, K-2$, add a constraint edge from z_{k+1} to z_k of infinite weight. Other edges may be required to represent unary and pairwise potentials (see (Kolmogorov and Zabih, 2004)).

In our formulation we will allow additional linear constraints to be imposed on the weights of the form $\boldsymbol{G}\boldsymbol{\theta} \geq \boldsymbol{h}$, where $\boldsymbol{G} \in \mathbb{R}^{d \times m}$ and $\boldsymbol{h} \in \mathbb{R}^d$. This is not typically necessary, but, as we will see below, is required when learning lower linear envelope potentials.

Now, let $\mathcal{Y}_t = \{0,1\}^n$ be the set of all possible assignments for the t-th training example. The (margin-rescaling) max-margin approach formulates learning as a quadratic programming optimization problem, $\textsc{MaxMarginQP}\big(\{\boldsymbol{y}_t, \mathcal{Y}_t\}_{t=1}^T, \boldsymbol{G}, \boldsymbol{h}\big)$:

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2}\|\boldsymbol{\theta}\|^2 + \tfrac{C}{T}\textstyle\sum_{t=1}^T \xi_t && (6)\\ &\text{subject to} \\ &\quad \boldsymbol{\theta}^T(\phi_t(\boldsymbol{y}) - \phi_t(\boldsymbol{y}_t)) + \xi_t \geq \Delta(\boldsymbol{y}, \boldsymbol{y}_t), && \forall t, \boldsymbol{y} \in \mathcal{Y}_t, \\ &\quad \xi_t \geq 0, && \forall t, \\ &\quad \boldsymbol{G}\boldsymbol{\theta} \geq \boldsymbol{h} \end{aligned}$$

where $C \geq 0$ is a regularization constant, and $\Delta(\boldsymbol{y}, \boldsymbol{y}_t)$ measures the loss between a ground-truth assignment $\boldsymbol{y}_t$ and any other assignment. In our work we use the Hamming loss, which measures the proportion of variables whose corresponding assignments disagree. More formally, the Hamming loss is defined as $\Delta(\boldsymbol{y}, \boldsymbol{y}') =$

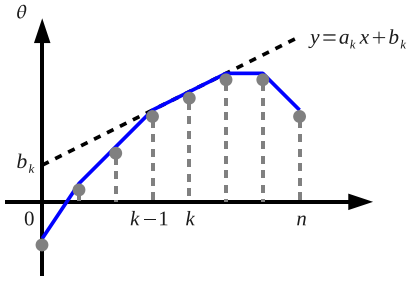


Figure 3. Example concave function of $x = \sum_{i=1}^{n} y_i$. The function can be represented as the minimum over a set of linear functions (lower linear envelope) or as a set of sampled points θ_k with curvature constraint.

$\frac{1}{n}\sum_{i=1}^{n} [\![y_i \neq y'_i]\!]$, where $[\![P]\!]$ is the indicator function taking value one when P is true and zero otherwise.

The number of constraints in the QP is exponential in the number of variables, and a standard approach to solving the max-margin QP is by adding constraints incrementally. Briefly, at each iteration the algorithm checks for the most violated constraint (for each training example), using *loss-augmented inference*, and, if found, adds it to the constraint set. The algorithm terminates when no more violated constraints are found (see Algorithm 1).

Transforming Between Representations. The max-margin formulation (see Equation 6) requires that the energy function be expressed as a linear combination of features and weights, however, our higher-order potential is represented as the minimum over a set of linear functions. One simple way to re-parameterize the energy function for learning is to sample the higher-order potential at each possible $x = \sum_{i=1}^{n} y_i$, i.e., at points $0, 1, \ldots, n$. Let $\boldsymbol{\theta} = (\theta_0, \ldots, \theta_n) \in \mathbb{R}^{n+1}$ be the sampled values. Then, we can retrieve the lower linear envelope representation as $a_k = \theta_k - \theta_{k-1}$ and $b_k = \theta_k - a_k k$ for $k = 1, \ldots, n$ as illustrated in Figure 3.[3] The corresponding feature vector $\phi(\boldsymbol{y}) \in \mathbb{R}^{n+1}$, under this representation, has the m-th element one if $\sum_{i \in \mathcal{C}} y_i = m$ and zero otherwise.

It remains to ensure that $\boldsymbol{\theta}$ represents a concave function. We do this by adding the second-order curvature constraint $\boldsymbol{D}^2\boldsymbol{\theta} \geq \mathbf{0}$ where $\boldsymbol{D}^2 \in \mathbb{R}^{(n-2)\times n}$ is the (negative) discrete second-derivative operator:

$$\boldsymbol{D}^2 = \begin{bmatrix} -1 & 2 & -1 & 0 & \cdots \\ & & \ddots & & \\ \cdots & 0 & -1 & 2 & -1 \end{bmatrix}. \tag{7}$$

[3]Note that if $a_k = a_{k-1}$ then the k-th linear function is redundant and can be omitted from the energy function.

Algorithm 1 Learning lower linear envelope MRFs.

1: **input** training set $\{\boldsymbol{y}_t\}_{t=1}^{T}$, regularization constant $C > 0$, and tolerance $\epsilon \geq 0$
2: **initialize** constraints set $\mathcal{A}_t = \{\}$ for all t
3: **repeat**
4: solve $\textsc{MaxMarginQP}\big(\{\boldsymbol{y}_t, \mathcal{A}_t\}_{t=1}^{T}, \boldsymbol{D}^2, \mathbf{0}\big)$ to get $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\xi}}$
5: convert from $\hat{\boldsymbol{\theta}}$ to (a_k, b_k) representation
6: **for** each training example, $t = 1, \ldots, T$ **do**
7: compute $\boldsymbol{y}_t^\star = \operatorname{argmin}_{\boldsymbol{y}} E(\boldsymbol{y}; \hat{\boldsymbol{\theta}}) - \Delta(\boldsymbol{y}, \boldsymbol{y}_t)$
8: **if** $\hat{\xi}_t + \epsilon < \Delta(\boldsymbol{y}_t^\star, \boldsymbol{y}_t) - E(\boldsymbol{y}_t^\star; \hat{\boldsymbol{\theta}}) + E(\boldsymbol{y}_t; \hat{\boldsymbol{\theta}})$ **then**
9: $\mathcal{A}_t \leftarrow \mathcal{A}_t \cup \{\boldsymbol{y}_t^\star\}$
10: **end if**
11: **end for**
12: **until** no more violated constraints
13: **return** parameters $\hat{\boldsymbol{\theta}}$

Our optimization follows the standard max-margin approach and is summarized in Algorithm 1.[4]

Theorem 4.1. *For $\epsilon = 0$, Algorithm 1 terminates with the optimal parameters $\boldsymbol{\theta}^\star$ for* $\textsc{MaxMarginQP}\big(\{\boldsymbol{y}_t, \mathcal{Y}_t\}_{t=1}^{T}, \boldsymbol{D}^2, \boldsymbol{0}\big)$.

Proof. By Theorem 3.6, our test for the most violated constraints (lines 7 and 8) can be performed exactly ($\Delta(\boldsymbol{y}, \boldsymbol{y}_t)$ decomposes as a sum of unary terms). If the test succeeds, then $\boldsymbol{y}_t^\star$ cannot already be in $\mathcal{A}_t$. It is now added (line 9). Since there are only finitely many constraints, this happens at most $2^n - 1$ times (per training example), and the algorithm must eventually terminate. On termination there are no more violated constraints, hence the parameters are optimal. □

Unfortunately, as our proof suggests, it may take exponential time for the algorithm to reach convergence with $\epsilon = 0$. Tsochantaridis et al. (2005) showed, however, that for $\epsilon > 0$ and no additional linear constraints (i.e., $\boldsymbol{G} = \mathbf{0}$, $\boldsymbol{h} = \mathbf{0}$) max-margin learning will terminate in a polynomial number of iterations. Their result can be extended to the case of additional linear constraints (details omitted due to space restrictions).

Clique-size Invariance. For many applications, the number of variables in the higher-order clique is extremely large. Furthermore, we may wish to

[4]To jointly learn the unary and pairwise weights, we augment the parameter vector $\boldsymbol{\theta}$ with a weight θ^{unary} for the unary terms and non-negative weight θ^{pair} for the pairwise terms, and add the corresponding features $\phi^{\text{unary}} = \sum_i \psi_i^U(y_i)$ and $\phi^{\text{pair}} = \sum_{ij} \psi_{ij}^P(y_i, y_j)$ to the feature vector $\phi(\boldsymbol{y})$. The non-negativity of θ^{pair} ensures that the energy function remains submodular.

learn and apply our models on problems with varying clique sizes. These issues can be addressed by undersampling the lower linear envelope to give a piecewise linear approximation to the envelope.

Assume that we are learning a lower linear envelope parameterized by samples $\boldsymbol{\theta} \in \mathbb{R}^{n+1}$. We can construct a feature vector that interpolates between samples of the linear envelope for any clique of size $m \geq n$ as follows. Let $p = \frac{1}{m}\sum_{i=1}^{m} y_i$. Then the k-th element of $\phi(\boldsymbol{y}) \in \mathbb{R}^{n+1}$ is

$$(\phi(\boldsymbol{y}))_k = \begin{cases} k - pn & \text{if } \frac{k-1}{n} \leq p < \frac{k}{n} \\ pn - k + 2 & \text{if } \frac{k-2}{n} \leq p < \frac{k-1}{n} \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

For example, if $n = 3$ and $p = \frac{1}{2} + \epsilon$ then $\phi(\boldsymbol{y}) = (0, \frac{1}{2} - 3\epsilon, \frac{1}{2} + 3\epsilon, 0)$. Note that $\mathbf{1}^T\phi(\boldsymbol{y}) = 1$.

At inference time, we can convert (a_k, b_k) for cliques of size n to $(\frac{n}{m}a_k, b_k)$ for cliques of size m. This acts to scale the linear envelope so that it depends on the proportion (not the absolute number) of variables that are assigned one and is therefore clique-size invariant.

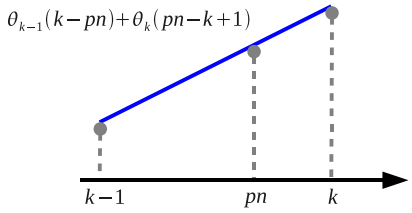


Figure 4. Illustration of interpolating between samples to construct features independent of clique size.

Alternative QP Formulations. Our quadratic program above is just one possible formulation that is based on a particular choice for representing the lower linear envelope and corresponding feature vectors. An alternative representation may encode the slope of the lower linear envelope directly, that is,

$$\widetilde{\theta}_i = \begin{cases} b_1 & \text{for } i = 0 \\ a_i = \theta_i - \theta_{i-1} & \text{for } i = 1, \ldots, K \end{cases} \tag{9}$$

The i-th component in the corresponding feature vector is then $(\widetilde{\phi}(\boldsymbol{y}))_i = \sum_{j \geq i} (\phi(\boldsymbol{y}))_j$. And instead of a second-order constraint $\boldsymbol{D}^2\boldsymbol{\theta} \geq \mathbf{0}$, we have a first-order constraint $\boldsymbol{D}\widetilde{\boldsymbol{\theta}} \geq \mathbf{0}$.

One of the advantages of this formulation is that it does not penalize constant envelopes (i.e., $\widetilde{\boldsymbol{\theta}} = \mathbf{0}$). Interestingly, under this formulation the optimal θ_0 is always zero, i.e., $b_1 = 0$, which is not surprising since from Equation 3 we see that b_1 only acts to offset the energy function.

We can take this process one step further and represent the higher-order potential as

$$\widetilde{\theta}_k = \begin{cases} b_1 & \text{for } k = 0 \\ a_1 & \text{for } k = 1 \\ a_k - a_{k-1} & \text{for } k = 2, \ldots, K \end{cases} \tag{10}$$

with appropriate feature vectors. Here we are encoding the coefficients of the pseudo-Boolean function used during inference directly and learning now resembles a latent-variable SVM formulation (Yu and Joachims, 2009) with constraints on the latent variables (namely, $z_{k+1} \leq z_k$).

5. Experimental Results

We conduct experiments on synthetic and real-world data, comparing baseline MRF models with ones that include higher-order terms learned by our method.

Synthetic Checkerboard. Our synthetic experiments involve an 8×8 checkerboard pattern of alternating white ($y_i = 1$) and black ($y_i = 0$) squares. Each square contains 256 variables, giving our MRF a total of $8 \times 8 \times 256 = 16,384$ variables. We generate a noisy version of the checkerboard as input to our model. Let $\boldsymbol{y}^\star$ be the ground-truth checkerboard, then our input is generated as $x_i = \eta_0[\![y_i^\star = 0]\!] - \eta_1[\![y_i^\star = 1]\!] + \delta_i$ where η_0 and η_1 are the signal-to-noise ratios for the black and white squares, respectively, and $\delta_i \sim \mathcal{U}(-1, 1)$ is additive i.i.d. uniform noise. Our unary terms are constructed for each pixel as $\psi_i^U(y_i) = \theta^{\text{unary}} x_i$. We add one lower linear envelope potential term for each square in the checkerboard, so each higher-order potential contains 256 variables and the terms are disjoint. Intuitively, we would like the potential to favour label consistency within the square. Our higher-order model does not contain any pairwise terms. We learn θ^{unary} and $\{(a_k, b_k)\}_{k=1}^K$ for $K = 10$ linear functions using Algorithm 1.

We report results on two different problem instances: The first has symmetric signal-to-noise ratios $\eta_0 = \eta_1 = 0.1$, and the second has five times less noise on the black squares ($\eta_0 = 0.5$) than on the white ($\eta_1 = 0.1$). Figure 5 shows the ground-truth checkerboard patterns and the noisy input. For both instances we set $C = 1000$ in Equation 6. Learning is run to convergence, taking 48 iterations for the first instance and 43 iterations for the second. Each training iteration took under 1s with inference taking about 120ms.

As a baseline, we compare our results against those from a pairwise MRF model. The unary terms are the same as above and the pairwise terms take the form $\psi_{ij}^P(y_i, y_j) = \theta^{\text{pair}}[\![y_i \neq y_j]\!]$, where i and j are

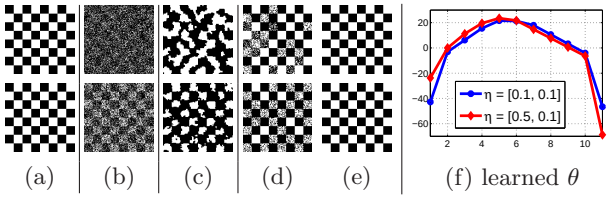


Figure 5. Results from our synthetic experiments. Panels (a)-(e) show the ground-truth, noisy input, predicted labels with unary and pairwise terms only, and using a model with higher-order terms for the third and final learning iterations, respectively. Two separate problem instances are shown (one per row). Panel (f) shows learned linear envelopes (parameters are normalized by the unary weight). Matlab source code for reproducing these results is available from the author's homepage.

neighbouring pixels. Here we set $\theta^{\text{unary}} = 1$ and choose θ^{pair} to give best the Hamming loss.

Figure 5 shows the inferred pattern for the pairwise MRF baseline, and for our higher-order model after the third and after the final training iterations ((c), (d), and (e), respectively). We see that after just three iterations our higher-order model is already performing well on both instances, and by the final iteration we can perfectly recover the checkerboard, unlike the pairwise model. The learned linear envelope parameters (relative to the unary weight) are shown in Figure 5(f). Note that for the second instance, our algorithm is able to learn an asymmetric potential.

Figure-Ground Segmentation. We also ran experiments on the real-world "GrabCut" problem (Rother et al., 2004), which aims to segment an object from an image given a user-annotated bounding box of the object (see Figure 6 for an example). Each pixel in the image is associated with a binary random variable indicating "background" or "foreground (i.e., object). Variables associated with pixels outside of the user-annotated bounding box are automatically assigned a label of zero (i.e., background). The assignment for the remaining variables is inferred.

We compare a model with learned higher-order terms against a baseline GrabCut model by performing leave-one-out cross-validation on a 50 image dataset from Lempitsky et al. (2009). Following Rother et al. (2004), our baseline model contains unary and pairwise terms. The unary terms are defined as the log-likelihood from foreground and background Gaussian mixture models (GMMs) over pixel colour and are image-specific. Briefly, the GMMs are initialized by learning a foreground and background model from pixels inside and outside the user-annotated bounding box, respectively. Next, the GMMs are used to relabel pixels as foreground or background, and their parameters re-estimated. This loop runs until convergence (or a maximum number of iterations is reached), and the final GMMs used to construct the unary terms.

The pairwise terms encode smoothness between each pixel and its eight neighbours, and are defined as

$$\psi_{ij}^{P}(y_i, y_j) = \frac{\lambda}{d_{ij}} [\![y_i \neq y_j]\!] \exp\left\{ -\frac{\|x_i - x_j\|^2}{2\beta} \right\} \quad (11)$$

where d_{ij} is the distance between pixels i and j, x_i and x_j are the RGB colour vectors for pixels i and j, β is the average squared-distance between adjacent RGB colour vectors in the image, and λ determines the strength of the pairwise smoothness term. It is the only free parameter in the baseline model and learned by cross-validation.

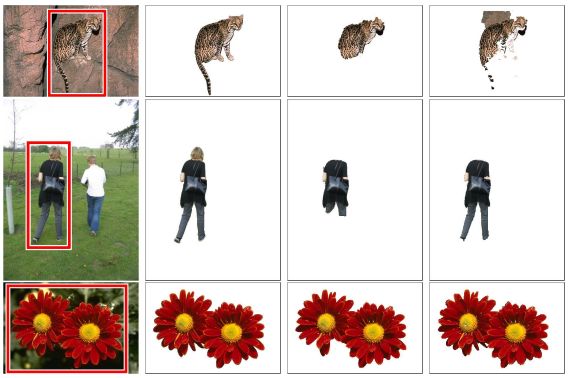
To construct the higher-order terms, we adopt a similar superpixel-based approach as Ladicky et al. (2009). First, we over-segment the image into a few hundred superpixels. The pixels within each superpixel then define a higher-order term, much like the checkerboard squares in our synthetic experiments. Here, however, the higher-order terms are over different sized cliques.

We learn the weights for the unary and pairwise potentials and the parameters for a lower linear envelope potential with $K = 10$ terms using Algorithm 1. We set $C = 1000$ and ran for a maximum of 100 iterations, however, for most folds, the algorithm converged before the maximum number of iterations was reached. The parameters determined at the last iteration were used for testing. Learning took approximately 3 hours per cross-validation fold with the majority of the time spent generating violated constraints for the 49 training images (each typically containing 640×480 pixels).

Some example results are shown in Figure 6. The first row shows that that our higher-order terms can capture some fine structure such as the cheetah's tail but it also segments part of the similarly-appearing rock. In the second example, we are able to correctly segment the person's legs. The third example shows that we are able to segment the petals at the lower part of the rightmost flower, which the baseline model does not. Quantitatively, our method achieves 91.5% accuracy compared to 90.0% for the strong baseline.

6. Discussion

This paper has shown how to perform efficient inference and learning for lower linear envelope binary MRFs, which are becoming popular for enforcing higher-order consistency constraints over large sets of random variables, particularly in computer vision.



(a) (b) (c) (d)

Figure 6. Example results from our GrabCut experiments. Shown are: (a) the image and bounding box, (b) ground-truth segmentation, (c) baseline model output, and (d) output from model with higher-order terms.

Our work suggests a number of directions for future research. Perhaps the most obvious is extending our approach to multi-label MRFs. We can already use our inference method inside move-making algorithms such as α-expansion or $\alpha\beta$-swap (Boykov et al., 1999), however, the question of efficient learning remains open since inference in this regime is only approximate.

More interesting is the implicit relationship between structured higher-order models and latent-variable SVMs (Yu and Joachims, 2009) as suggested by the introduction of auxiliary variables for inference and our alternative QP formulations. Exploring this relationship further may provide insights into both models.

Acknowledgments. We thank Bob Williamson and the anonymous reviewers for their helpful feedback. This work was supported by the Australian Research Council.

References

E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123:155–225, 2002.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 26:1124–1137, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Fast approx. energy minimization via graph cuts. In *Proc. of the International Conference on Computer Vision (ICCV)*, 1999.

T. Finley and T. Joachims. Training structural SVMs when exact inference is intractable. In *Proc. of the International Conference on Machine Learning (ICML)*, 2008.

D. Freedman and P. Drineas. Energy minimization via graph cuts. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

P. L. Hammer. Some network flow problems solved with psuedo-boolean programming. *Operations Research*, 13: 388–399, 1965.

H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 25:1333–1336.

H. Ishikawa. Higher-order clique reduction in binary graph cut. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77:27–59, 2009.

P. Kohli and M. P. Kumar. Energy minimization for linear envelope MRFs. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

P. Kohli, M. P. Kumar, and P. H. S. Torr. P3 & beyond: Solving energies with higher order cliques. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

P. Kohli, L. Ladicky, and P. H. S. Torr. Graph cuts for minimizing higher order potentials. Technical report, Microsoft Research, 2008.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 26:65–81, 2004.

L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Associative hierarchical CRFs for object class image segmentation. In *Proc. of the International Conference on Computer Vision (ICCV)*, 2009.

V. Lempitsky, P. Kohli, C. Rother, and T. Sharp. Image segmentation with a bounding box prior. In *Proc. of the International Conference on Computer Vision (ICCV)*, 2009.

S. Nowozin and C. H. Lampert. Global connectivity potentials for random field models. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. In *Proc. of the Intl. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2004.

C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph-cuts. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2008.

B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *Proc. of the International Conference on Machine Learning (ICML)*, 2005.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector learning for interdependent and structured output spaces. In *Proc. of the International Conference on Machine Learning (ICML)*, 2004.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453–1484, 2005.

C.-N. Yu and T. Joachims. Learning structural SVMs with latent variables. In *Proc. of the International Conference on Machine Learning (ICML)*, 2009.