Learning Low-order Models for Enforcing High-order Statistics

Patrick Pletscher
ETH Zurich
Zurich, Switzerland

Pushmeet Kohli
Microsoft Research
Cambridge, UK

Abstract

Models such as pairwise conditional random fields (CRFs) are extremely popular in computer vision and various other machine learning disciplines. However, they have limited expressive power and often cannot represent the posterior distribution correctly. While learning the parameters of such models which have insufficient expressivity, researchers use loss functions to penalize certain misrepresentations of the solution space. Till now, researchers have used only simplistic loss functions such as the Hamming loss, to enable efficient inference. The paper shows how sophisticated and useful higher order loss functions can be incorporated in the learning process. These loss functions ensure that the MAP solution does not deviate much from the ground truth in terms of certain *higher order statistics*. We propose a learning algorithm which uses the recently proposed lower-envelop representation of higher order functions to transform them to pairwise functions, which allow efficient inference. We test the efficacy of our method on the problem of foreground-background image segmentation. Experimental results show that the incorporation of higher order loss functions in the learning formulation using our method leads to much better results compared to those obtained by using the traditional Hamming loss.

1 Introduction

Probabilistic models such as conditional random fields (CRFs) are extremely popular machine learning disciplines. Pairwise CRFs, in particular, have been used to formulate many image labeling problems in computer vision (Szeliski et al., 2008). However, their inability to handle higher order dependencies between random variables restricts their expressive power, and makes them unable to represent the data well (Sudderth & Jordan, 2008) i.e., the ground truth may not be the Maximum a Posterior (MAP) solution under the model.

Models containing higher order factors are able to encode complex dependencies between groups of variables, and can encourage solutions which match the statistics of the ground truth solution (Potetz, 2007; Roth & Black, 2005; Woodford et al., 2009). However, the high computational cost of performing MAP inference in such models has inhibited their use (Lan et al., 2006). Instead, there has been a widespread adoption of the simpler and less powerful pairwise-CRF models which allow efficient inference (Szeliski et al., 2008).

While learning the parameters of models with insufficient expressivity, researchers can penalize certain misrepresentations of the solution space using a 'loss function' which specifies the deviations from ground truth that the learning algorithm should avoid (Tsochantaridis et al., 2005; Taskar et al., 2003). Most previous works on these topics have used simple choices of the loss function, such as the Hamming loss or squared loss, which lead to tractable learning algorithms (Szummer et al., 2008). However, in real world applications, researchers might prefer more general loss functions which penalize deviations in some higher order statistics.

The ability to use such higher order loss functions is particularly important for many image labeling problems in medical imaging where predictions other than pixel labelling accuracy (Hamming loss) might be important. For instance, in some diagnostic scenarios, radiologists/physicians are interested in the area/volume of the segmentation of a tissue or tumor that is under investigation. In such cases, a loss function that heavily penalizes solutions whose volume/area is very different from that of the ground truth should be used.

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume 22 of JMLR: W&CP 22.

In this paper, we show how to learn the parameters of low-order models such as pairwise CRFs under higher-order loss functions. These loss functions can ensure that the MAP solution does not deviate much from the ground truth in terms of certain higher order statistics. We propose an efficient learning algorithm which uses the lower-envelop representation of higher order functions (Kohli & Kumar, 2010) to transform them to pairwise functions. We demonstrate the power of our method on the problem of foreground-background image segmentation. Experimental results show that our method is able to obtain parameters which lead to better results compared to the traditional approach.

2 Max-margin learning

This section reviews max-margin learning (Taskar et al., 2003; Tsochantaridis et al., 2005) and introduces our notation. For a given input $\boldsymbol{x} \in \mathcal{X}$ we consider models that predict a multivariate output $\boldsymbol{y} \in \mathcal{Y}$[1] by maximizing a linearly parametrized score function (a MAP predictor):

$$\boldsymbol{f}_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{y}\in\mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\rangle. \tag{1}$$

Here $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})$ denotes a mapping of the input and output variables to a joint input/output feature space. In computer vision, such a feature map is generally specified implicitly through a graphical model. Furthermore, $\boldsymbol{w}$ denote the parameters of the model. In our work we consider pairwise models $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with energies of the form

$$E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}) = -\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\rangle = \sum_{i\in\mathcal{V}} \psi_i(y_i, \boldsymbol{x}; \boldsymbol{w}^u) + \sum_{(i,j)\in\mathcal{E}} \psi_{ij}(y_i, y_j, \boldsymbol{x}; \boldsymbol{w}^p). \tag{2}$$

Here $\boldsymbol{w}$ is separated into parameters for the unary potentials ($\boldsymbol{w}^u$) and pairwise potentials ($\boldsymbol{w}^p$). The maximization problem in (1) can alternatively be written as an energy minimization

$$\boldsymbol{f}_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{y}\in\mathcal{Y}} E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}). \tag{3}$$

Having defined the form of the prediction function, we now consider learning the parameters $\boldsymbol{w}$ of such a model. Given the training data set $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^N, \boldsymbol{y}^N)\}$, max-margin learning[2] (or equivalently the structured SVM) formulates an upper bound on the empirical risk using a quadratic program (QP) with a combinatorial number of constraints. The exponential number of constraints can be dealt with by a cutting-plane approach (Tsochantaridis et al., 2005). The resulting QP for a regularizer weight λ reads as follows:

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \sum_{n=1}^{N} \xi^n \tag{4}$$

$$\text{s.t.} \quad \max_{\boldsymbol{y}\in\mathcal{Y}} \left[\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^n, \boldsymbol{y})\rangle + \Delta_{\boldsymbol{y}^n}(\boldsymbol{y})\right] - \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^n, \boldsymbol{y}^n)\rangle \geq \xi^n \quad \forall n \tag{5}$$

$$\xi^n \geq 0.$$

The slack-variable ξ^n measures the surrogate loss of the n-th example. $\Delta_{\boldsymbol{y}^n}(\boldsymbol{y})$ denotes an application-specific loss function, measuring the error incurred when predicting $\boldsymbol{y}$ instead of the ground truth output $\boldsymbol{y}^n$. We shall denote a generic ground truth label by $\boldsymbol{y}^*$. The loss of an example, as given by the constraint in (5) is convex and hence the overall optimization problem allows for efficient optimization over $\boldsymbol{w}$. The QP is typically solved by variants of the cutting-plane method shown in Algorithm 1. The algorithm operates in an alternating fashion by first generating the constraints for the current parameter estimates and thereafter solving the QP with the extended set of constraints.

Algorithm 1 Cutting-plane algorithm as in (Finley & Joachims, 2008).

Require: $(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^N, \boldsymbol{y}^N), \lambda, \epsilon, \Delta_{\boldsymbol{y}^*}(\cdot)$.

1: $S^n \leftarrow \emptyset$ for $n = 1, \ldots, N$.
2: **repeat**
3: **for** $n = 1, \ldots, N$ **do**
4: $H(\boldsymbol{y}) := \Delta_{\boldsymbol{y}^n}(\boldsymbol{y}) + \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^n, \boldsymbol{y}) - \boldsymbol{\phi}(\boldsymbol{x}^n, \boldsymbol{y}^n)\rangle$
5: compute $\hat{\boldsymbol{y}} = \operatorname{argmax}_{\boldsymbol{y}\in\mathcal{Y}} H(\boldsymbol{y})$
6: compute $\xi^n = \max\{0, \max_{\boldsymbol{y}\in S^n} H(\boldsymbol{y})\}$
7: **if** $H(\hat{\boldsymbol{y}}) > \xi^n + \epsilon$ **then**
8: $S^n \leftarrow S^n \cup \{\hat{\boldsymbol{y}}\}$
9: $\boldsymbol{w} \leftarrow$ optimize primal over $\bigcup_n S^n$
10: **end if**
11: **end for**
12: **until** no S^n has changed during iteration

Line 9 of Algorithm 1 corresponds to solving a standard QP for the constraints in $\bigcup_n S^n$ (a linear number of constraints as in each iteration at most one additional constraint is added for each example). The *loss augmented* inference problem on line 5 poses the major computational bottleneck for many applications. Here, an energy minimization of the form (3) needs to be solved, with one important difference: *The negative loss term enters the energy.* Depending on the loss term this can render the inference problem intractable. The loss augmented inference problem is

[1] Generally the dimension of the output space depends on the input $\boldsymbol{x}$, which is neglected here.

[2] We consider the margin rescaled version.

investigated in detail in section 4. The next section discusses loss functions in general and introduces the label-count loss, which is promoted in our work.

3 Loss functions

Max-margin learning leaves the choice of the loss function $\Delta_{\boldsymbol{y}^*}(\boldsymbol{y})$ unspecified. The loss allows the researcher to adjust the parameter estimation to the evaluation which follows the learning step. In our work we differentiate between low-order losses, which factorize, and high-order losses, which do not factorize. Factorization is considered to be a key property of a loss to maintain computational tractability.

3.1 Low-order loss functions

For image labelling in computer vision a popular choice is the pixelwise error, or also Hamming error. It is defined as:

$$\Delta_{\boldsymbol{y}^*}^{hamming}(\boldsymbol{y}) = \sum_{i \in \mathcal{V}} y_i \neq y_i^*. \tag{6}$$

For image labelling problems, it tries to prevent solutions with high pixel labelling error from having low energy under the model compared to the ground truth. If there is a natural ordering on the labels, such as in image denoising, another common choice for the loss is the squared pixelwise error. For the binary problems studied in our work, it is equivalent to the Hamming loss.

3.2 High-order loss functions

In many machine learning applications, practitioners are concerned with errors other than the simple Hamming loss. This is especially the case in medical imaging tasks involving segmentation of particular tissues or tumors. In such problems, radiologists and physicians are sometimes more interested in measuring the exact volume or area of the tumor (or tissue) to analyze if it is increasing or decreasing in size. This preference can be handled during the learning process by using a label-count based loss function.

More formally, consider a two-label image segmentation problem where we have to assign the label '0' (representing 'tumour') or '1' (representing 'non-tumour') to every pixel/voxel in the image/volume. The area/volume based *label-count loss* function in this case is defined as:

$$\Delta_{\boldsymbol{y}^*}^{count}(\boldsymbol{y}) = \left| \sum_{i \in \mathcal{V}} y_i - \sum_{i \in \mathcal{V}} y_i^* \right|. \tag{7}$$

Such a loss function prevents image labellings (segmentations) with substantially different area/volume compared to the ground truth to be assigned a low energy under the model. As we will show, despite the high-order form of the label-count loss, learning with it in the max-margin framework is tractable.

It is easy to show that the label-count loss is a lower bound on the Hamming loss:

$$\Delta_{\boldsymbol{y}^*}^{count}(\boldsymbol{y}) \leq \Delta_{\boldsymbol{y}^*}^{hamming}(\boldsymbol{y}). \tag{8}$$

The work of Lempitsky & Zisserman (2010), Gould (2011) and Tarlow & Zemel (2011) are most closely related to our paper. In (Lempitsky & Zisserman, 2010) a learning approach for counting is introduced. The major difference to our work stems from the model that is learned. In their work a continuous regression function is trained, which predicts for each pixel a positive real independent of all its neighboring pixels. In our work a CRF is used, which includes dependencies among variables, only the loss term in learning is changed. (Gould, 2011) discusses max-margin parameter learning in graphical models that contain potentials with a linear lower envelope representation. However, the loss function used in their work is still restricted to be a simple Hamming loss. The idea of learning with higher-order losses is also studied in (Tarlow & Zemel, 2011). They discuss several higher-order loss functions, but only approximate algorithms are presented. To the best of our knowledge, our work introduces for the first time a subclass of high-order loss functions, for which max-margin learning remains tractable.

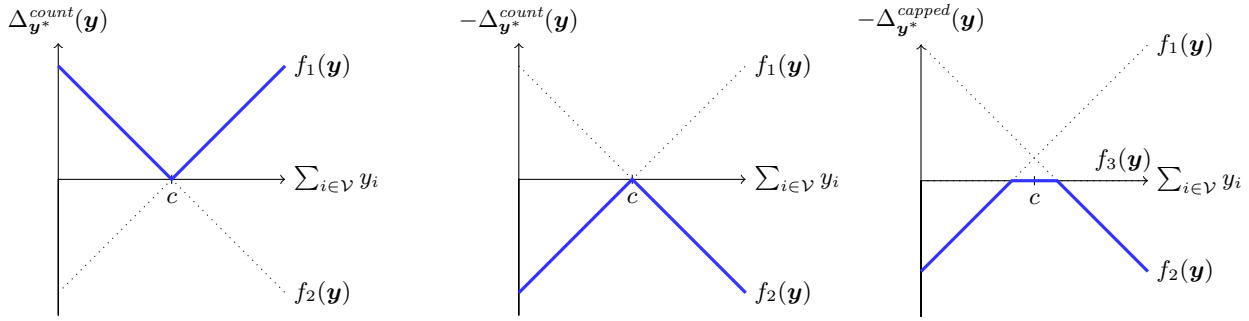
4 Loss augmented inference and lower-envelope representation

The loss-augmented energy minimization problem for a given input/output pair $(\boldsymbol{x}, \boldsymbol{y}^*)$ is given by

$$\min_{\boldsymbol{y}} E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}) - \Delta_{\boldsymbol{y}^*}(\boldsymbol{y}). \tag{9}$$

Even on its own, the problem of minimizing a general energy function of discrete variables is a NP-hard problem. However, certain classes of functions have been identified for which the problem can be solved exactly in polynomial time. These include pairwise functions that are defined over graphs that are tree-structured (Pearl, 1986) or perfect (Jebara, 2009).

Another important family of tractable functions are submodular functions which are discrete analogues of convex functions (Fujishige, 1991; Lovasz, 1983), a formal definition is given in the appendix. Submodular functions are particularly important because of their wide use in modeling labelling problems in computer vision such as 3D voxel segmentation (Snow et al.,



(a) Upper envelope for $\Delta^{count}_{\boldsymbol{y}^*}(\boldsymbol{y})$. (b) Lower envelope for $-\Delta^{count}_{\boldsymbol{y}^*}(\boldsymbol{y})$. (c) Capped loss.

Figure 1: Upper and lower envelope representations of the label-count loss and its negation. Here $c := \sum_{i\in\mathcal{V}} y^*_i$. Interestingly, as the loss enters the loss-augmented energy with a negative sign, the resulting energy minimization problem $\min_{\boldsymbol{y}} E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}) - \Delta^{count}_{\boldsymbol{y}^*}(\boldsymbol{y})$ becomes tractable. (c) shows an example of a loss which can be described as the lower envelope of three linear functions.

2000) and foreground-background image segmentation problems (Boykov & Jolly, 2001; Blake et al., 2004).

The presence of the loss term in the loss augmented energy minimization problem in (9) has the potential to make it harder to minimize. The Hamming loss, however, has the nice property that it decomposes into unary terms which can be integrated in the energy, and thus does not make the loss-augmented energy minimization problem harder (Szummer et al., 2008).

4.1 Compact representation of higher-order loss functions

While it is easy to incorporate the Hamming loss in the learning formulation, this is not true for higher order loss functions. In fact, a general n order loss function defined on k-state variables can require up to k^n parameters for just its definition. In recent years a lot of research has been done on developing compact representation of higher-order functions (Kohli et al., 2007; Rother et al., 2009; Kohli & Kumar, 2010). In particular, Kohli & Kumar (2010) proposed a representation based on *upper* and *lower* envelopes of linear functions which enables the use of many popular classes of higher order potentials employed in computer vision. More formally, they represent higher order functions as:

$$f^h(\boldsymbol{y}) = \otimes_{q\in\mathcal{Q}} f^q(\boldsymbol{y}) \tag{10}$$

where $\otimes = \{\max, \min\}$, and $\mathcal{Q}$ indexes a set of linear functions, defined as

$$f^q(\boldsymbol{y}) = \mu^q + \sum_{i\in\mathcal{V}} \sum_{a\in\mathcal{L}} \nu^q_{ia} \delta(y_i = a) \tag{11}$$

where the weights ν^q_{ia} and the constant term μ^q are the parameters of the *linear* function $f^q(\cdot)$, and the function $\delta(y_i = a)$ returns 1 if variable y_i takes label a and returns 0 for all other labels. While the $\otimes$ ='min' results in a lower envelope of the linear function, 'max' results in the upper envelope.

The upper envelope representation, in particular, is very powerful and is able to encode sophisticated silhouette constraints for 3D reconstruction (Kohli & Kumar, 2010; Kolev & Cremers, 2008). It can also be used to compactly represent general higher order energy terms which encourage solutions to have a particular distribution of labels. Woodford et al. (2009) had earlier shown that such terms were very useful in formulations of image labelling problems such as image denoising and texture, and led to much better results.

Our higher order loss term defined in equation (7) can be represented by taking the upper envelope of two linear functions $f^1(\cdot)$ and $f^2(\cdot)$ that are defined as:

$$f^1(\boldsymbol{y}) = \sum_{i\in\mathcal{V}} y_i - \sum_{i\in\mathcal{V}} y^*_i, \tag{12}$$

$$f^2(\boldsymbol{y}) = \sum_{i\in\mathcal{V}} y^*_i - \sum_{i\in\mathcal{V}} y_i. \tag{13}$$

This is illustrated in Fig. 1a.

4.2 Minimizing loss augmented energy functions

Although upper envelope functions are able to represent a large class of useful higher order functions, inference in models containing upper envelope potentials involves the solution of a hard min-max optimization problem (Kohli & Kumar, 2010).

We made the observation that the loss term in the loss-augmented energy minimization problem (9) has a negative coefficient, which allows us to represent the

label-count based loss (7) by the lower envelope of the functions defined in equation (12) and (13) (visualized in Fig. 1b).

Kohli and Kumar showed that the minimization of higher order functions that can be represented as lower envelopes of linear functions can be transformed to the minimization of a pairwise energy function with the addition of an auxiliary variable. In fact, in some cases, the resulting pairwise energy function can be shown to be submodular (Boros & Hammer, 2002; Kolmogorov & Zabih, 2004) and hence can be minimized by solving an minimum cost st-cut problem (Kohli et al., 2008). This is the case for all higher-order functions of Boolean variables which are defined as:

$$f^h(\boldsymbol{y}) = \mathcal{F}\left(\sum_{i\in\mathcal{V}} y_i\right), \tag{14}$$

where $\mathcal{F}$ is a concave function. The worst case time complexity of the procedure described above is polynomial in the number of variables. A related family of higher order submodular functions which can be efficiently minimized was characterized in (Stobbe & Krause, 2010). Next, we consider the loss augmented inference for the label-count loss in more detail.

4.3 Label-count loss augmented inference

The minimization of the negative label-count based loss (7) can be transformed to the following pairwise submodular function minimization problem:

$$\begin{aligned}
&\min_{\boldsymbol{y}} -\Delta^{count}_{\boldsymbol{y}^*}(\boldsymbol{y}) \\
&= \min_{\boldsymbol{y}} -\left|\sum_{i\in\mathcal{V}} y_i - \sum_{i\in\mathcal{V}} y_i^*\right| \qquad (15)\\
&= \min_{\boldsymbol{y},z\in\{0,1\}} -z\left(\sum_{i\in\mathcal{V}} y_i - \sum_{i\in\mathcal{V}} y_i^*\right) \\
&\qquad -(1-z)\left(\sum_{i\in\mathcal{V}} y_i^* - \sum_{i\in\mathcal{V}} y_i\right) \\
&= \min_{\boldsymbol{y},z\in\{0,1\}} 2z\left(\sum_{i\in\mathcal{V}} y_i^* - \sum_{i\in\mathcal{V}} y_i\right) + \sum_{i\in\mathcal{V}} y_i - \sum_{i\in\mathcal{V}} y_i^*.
\end{aligned}$$

The full energy minimization for the count loss augmented inference reads as follows

$$\min_{\boldsymbol{y},z\in\{0,1\}} E(\boldsymbol{y},\boldsymbol{x},\boldsymbol{w}) + 2z\left(\sum_{i\in\mathcal{V}} y_i^* - \sum_{i\in\mathcal{V}} y_i\right) + \sum_{i\in\mathcal{V}} y_i - \sum_{i\in\mathcal{V}} y_i^*. \tag{16}$$

We assume that the original energy $E(\boldsymbol{y},\boldsymbol{x},\boldsymbol{w})$ is submodular. The pairwise problem above is exactly solved by graph-cut (Boykov, 2001) on the original graph $\mathcal{G}$ where we add one node for the variable z and $|\mathcal{V}|$ new edges connecting each segmentation variable y_i to the auxiliary variable z. The pairwise energy construction is visualized in Fig. 2.

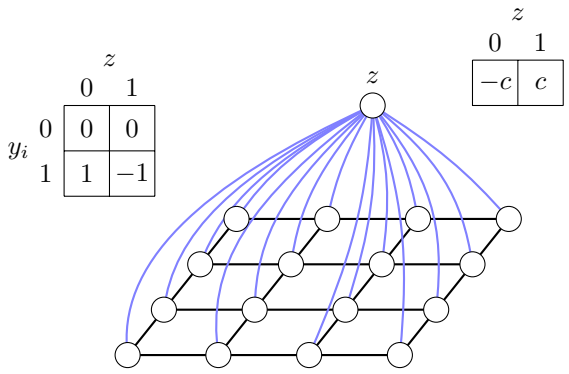


Figure 2: Pairwise graph used for solving the label-count loss augmented inference problem. The potentials of the edges connecting the segmentation nodes y_i to the auxiliary node z (which are shown in blue) are visualized to the left. The unary potential of the auxiliary variable z to the right, where $c := \sum_i y_i^*$. Standard graph-cut solvers can be applied to this problem.

Unfortunately, we found the de-facto standard computer vision graph-cut algorithm by Boykov & Kolmogorov (2004) to run fairly slowly on these problem instances. We attribute this to the dense connectivity of the auxiliary node z. This problem is in theory, and as is turns out also in practice, solved by the recent iterative breadth-first search (IBFS) graph-cut algorithm introduced in (Goldberg et al., 2011). We found this algorithm to be roughly an order of magnitude more efficient than the Boykov-Kolmogorov algorithm. Learning on a small subset of the data discussed in the next section took two minutes when IBFS was used and around 25 minutes with the Boykov-Kolmogorov algorithm.

Alternatively, for minimizing the loss augmented energy with a single Boolean z, as in (16), we can solve the minimization efficiently by performing energy minimization twice in the original graph (for $z = 0$ and $z = 1$). Each choice of z results in different unaries. This approach does however not scale to the case where we have multiple zs as the number of sub-problems grows exponentially. If we have a loss function with 10 zs we will have to do the minimization 2^{10} times.

5 Experiments

We implemented the max-margin learning in Matlab. For solving the QP the MOSEK solver was used. The loss augmented inference with IBFS was implemented

in C++ through a MEX wrapper. The IBFS code was downloaded from the authors webpage and modified to allow for double precision energies (as opposed to integer precision). Submodularity of the model was explicitly enforced in training by ensuring that all the edge potential's off-diagonals are larger than the diagonals. This can be achieved by adding additional constraints to the QP. The loss is always normalized by the number of pixels such that the loss is upper bounded by one.

5.1 Cell segmentation

Counting tasks naturally arise in many medical applications. The estimation of the progression of cancer in a tissue or the density of cells in microscope images are two examples. As a first experiment we study the problem of counting the number of mitochondria cell pixels in an image. The dataset is visualized in Fig. 3. The images have been provided by Ángel Merchán and Javier de Felipe from the Cajal Blue Brain team at the Universidad Politécnica de Madrid. Three images

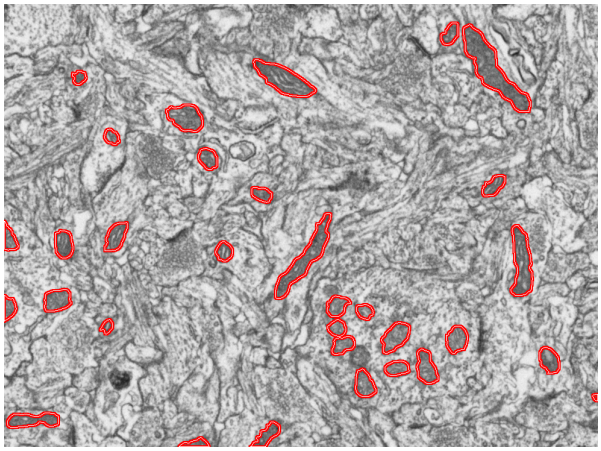


Figure 3: Electroscopic image showing the mitochondria cells in red.

were used for learning, two images for the validation and the remaining five images for testing. The images have a resolution of 986×735. The pairwise CRF consisted of a unary term with three features (the response of a unary classifier for mitochondria and synapse detection and an additional bias feature). The pairwise term incorporated two features (color difference between neighboring pixels and a bias). The results are shown in a box plot in Fig. 4. As expected the label-count loss trained model performs better than the Hamming loss trained model if the label-count loss is used for the evaluation and vice-versa if evaluated on the Hamming loss.

We also compared our lower envelope inference approach to the COMPOSE max-product algorithm (Duchi et al., 2006) which is used in (Tarlow & Zemel, 2011). The latter inference approach is in general only approximate. However, for the cell segmentation problem in combination with the label-count loss, the solutions obtained using the two different loss-augmented inference algorithms were almost identical. The running time of the two approaches is also comparable. Our inference algorithm is slightly more efficient, but also more adapted to the count-loss.

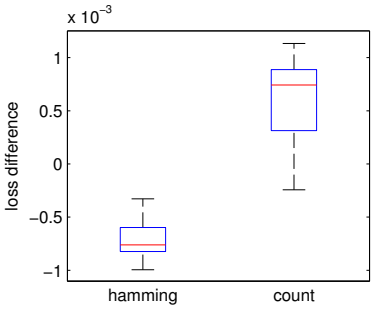
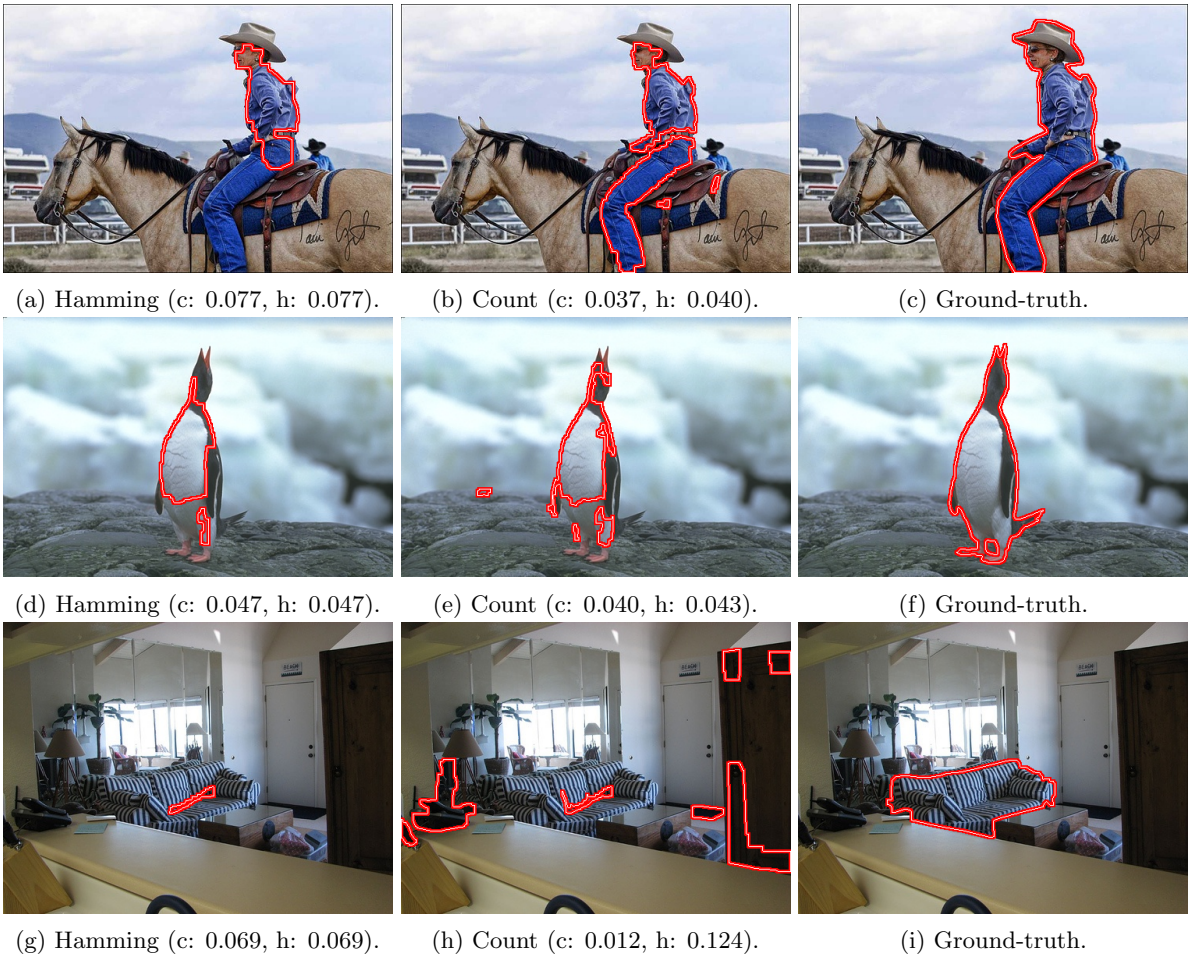


Figure 4: Results for the mitochondria segmentation. We plot the normalized loss difference between the Hamming loss trained model and the one trained using label-count loss. The x-axis shows the loss used for the evaluation of the predictions. The negative value for the Hamming loss evaluation indicates that if Hamming loss is used for evaluation, training with the Hamming loss is superior. The opposite is true when evaluation considers the label-count loss as learning with the label-count results in a lower loss.

5.2 Foreground-background segmentation

We check the effectiveness of the label-count loss for the task of background-foreground segmentation on the Grabcut dataset (Blake et al., 2004). We use the extended dataset from (Gulshan et al., 2010). The dataset consists of 151 images, each comes with a ground truth segmentation. Furthermore, for each image an initial user seed is specified by strokes marking pixels belonging to the foreground or to the background, respectively. As unary features we used the three color channels together with the background and foreground posterior probabilities as computed by the Gaussian mixture model algorithm used in Grabcut. Additionally we also included a constant feature to correct for class bias. For the pairwise features we used the color difference between the two pixels and again a bias feature. The standard four-connected grid graph is used as the basic model. Each edge is parametrized by the same parameter. We also experimented with extensions of this basic model: In one variant we consider the eight-connected grid, in the other variant each di-



(a) Hamming (c: 0.077, h: 0.077). (b) Count (c: 0.037, h: 0.040). (c) Ground-truth.

(d) Hamming (c: 0.047, h: 0.047). (e) Count (c: 0.040, h: 0.043). (f) Ground-truth.

(g) Hamming (c: 0.069, h: 0.069). (h) Count (c: 0.012, h: 0.124). (i) Ground-truth.

Figure 5: Segmentations on the test set for models trained using the Hamming loss (left) and the label-count loss (middle). The image on the right shows the ground truth segmentation. We show the measured count loss and Hamming loss in brackets. The bottom row shows a case where the model trained using the count loss shows a much better count loss, however the Hamming loss substantially deteriorates due to the false positives. For the first two images, the label-count loss trained model even outperforms the Hamming loss trained model in terms of Hamming loss.

rection of the edge is parameterized using a different parameter. The basic model is therefore specified by an eight dimensional parameter, the eight-connected model where each direction has its own parameter by a 14-dimensional parameter. For learning 60 images were used, 20 for the validation of the regularization parameter λ, the remaining images were used for testing.

Fig. 5 shows some of the learned segmentations and Table 1 gives a comparison of the models trained using the Hamming loss and the label-count loss. The results were averaged over four different data splits. As expected, we observe that if the label-count loss is used for the evaluation, the model that is trained using this loss performs superior. More interesting is the result for the case when the Hamming loss is used for the evaluation. Despite the fact that the appropriate loss is used in training, we do not identify a statistically significant advantage of the Hamming loss over the label-count loss. This could be explained by the max-margin objective only considering an upper bound on the loss, and not the actual loss itself. The label-count loss might suffer less from this upper bounding than the Hamming loss.

6 Discussion

We have demonstrated, for the first time, how low-order models like pairwise CRFs can be encouraged to preserve higher order statistics by introducing higher order loss functions in the learning process. The learning involves the minimization of the loss augmented energy, which we show can be performed exactly for certain loss functions by employing a transformation

Eval \ Train		Hamming better (%)	Count better (%)
4/S	Hamming	52.1 ± 7.0	47.9 ± 7.0
	Count	33.8 ± 8.3	66.2 ± 8.3
4/D	Hamming	39.4 ± 6.1	60.6 ± 6.1
	Count	29.6 ± 8.3	70.4 ± 8.3
8/S	Hamming	48.2 ± 11.9	51.8 ± 11.9
	Count	32.0 ± 13.1	68.0 ± 13.1
8/D	Hamming	50.0 ± 9.2	50.0 ± 9.2
	Count	40.5 ± 14.3	59.5 ± 14.3

Table 1: Test performance of models trained using the Hamming and the label-count loss for different model structures. The structure of the model is shown on the far left (4 vs. 8 grid, same vs. different parameterization of the edges). The second column shows the percentage of images for which the model trained using Hamming loss has a lower evaluation loss. The third column shows the same information for the label-count loss. The rows show the loss used in the evaluation. If the loss function affects training, we would expect both columns to show values considerably above 50% for the corresponding loss. For learning with the label-count loss this is the case, for the Hamming loss the two learned models perform roughly the same.

scheme. We demonstrate the efficacy of our method by using a label-count loss while learning a pairwise CRF model for binary image segmentation. The label-count loss function is useful for applications that require the count of positively labeled pixels in an image to match the count observed on a ground truth segmentation. Our proposed algorithm enables efficient max-margin learning under the label-count loss, and leads to models that produces solutions with statistics that are closer to the ground truth, compared to solutions of models learned using the standard Hamming loss.

Acknowledgements

We would like to thank Pablo Márquez Neila for sharing the mitochondria cell segmentation data set and the unary classifier responses. We would also like to thank D. Tarlow for helping us with getting the COMPOSE inference code to work in combination with the label-count loss.

References

Blake, A., Rother, C., Brown, M., Pérez, P., and Torr, P. H. S. Interactive image segmentation using an adaptive gmmrf model. In *ECCV*, pp. 428–441, 2004.

Boros, E. and Hammer, P.L. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 2002.

Boykov, Y. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 8(4):413–1239, 2001.

Boykov, Y. and Jolly, M.P. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, 2001.

Boykov, Y. and Kolmogorov, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.

Duchi, J., Tarlow, D., Elidan, G., and Koller, D. Using Combinatorial Optimization within Max-Product Belief Propagation. In *NIPS*, 2006.

Finley, T. and Joachims, T. Training structural SVMs when exact inference is intractable. *Proceedings of the 25th international conference on Machine learning - ICML '08*, pp. 304–311, 2008.

Fujishige, S. *Submodular functios and optimization.* Annals of Discrete Mathematics, Amsterdam, 1991.

Goldberg, A. V., Hed, S., Kaplan, H., Tarjan, R. E., and Werneck, R. F. Maximum flows by incremental breadth-first search. In *ESA*, pp. 457–468, 2011.

Gould, S. Max-margin learning for lower linear envelope potentials in binary markov random fields. *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

Gulshan, V., Rother, C., Criminisi, A., Blake, A., and Zisserman, A. Geodesic star convexity for interactive image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

Jebara, T. MAP estimation, message passing, and perfect graphs. In *Uncertainty in Artificial Intelligence*, 2009.

Kohli, P. and Kumar, M. P. Energy minimization for linear envelope MRFs. In *CVPR*, 2010.

Kohli, P., Kumar, M. P., and Torr, P. H. S. P^3 and beyond: Solving energies with higher order cliques. In *CVPR*, 2007.

Kohli, P., Ladicky, L., and Torr, P. Robust higher order potentials for enforcing label consistency. In *CVPR*, 2008.

Kolev, K. and Cremers, D. Integration of multiview stereo and silhouettes via convex functionals on convex domains. In *ECCV*, 2008.

Kolmogorov, V. and Zabih, R. What energy functions can be minimized via graph cuts?. *PAMI*, 2004.

Lan, X., Roth, S., Huttenlocher, D. P., and Black, M. J. Efficient belief propagation with learned higher-order markov random fields. In *ECCV (2)*, pp. 269–282, 2006.

Lempitsky, V. and Zisserman, A. Learning To Count Objects in Images. In *NIPS*, 2010.

Lovasz, L. Submodular functions and convexity. In *Mathematical Programming: The State of the Art*, pp. 235–257, 1983.

Pearl, J. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29(3):241–288, 1986.

Potetz, B. Efficient belief propagation for vision using linear constraint nodes. In *CVPR*, 2007.

Roth, S. and Black, M. J. Fields of experts: A framework for learning image priors. In *CVPR*, pp. 860–867, 2005.

Rother, C., Kohli, P., Feng, W., and Jia, J. Minimizing sparse higher order energy functions of discrete variables. In *CVPR*, 2009.

Snow, D., Viola, P., and Zabih, R. Exact voxel occupancy with graph cuts (470), 2000.

Stobbe, P. and Krause, A. Efficient minimization of decomposable submodular functions. In *Proc. Neural Information Processing Systems (NIPS)*, 2010.

Sudderth, E. and Jordan, M. Shared segmentation of natural scenes using dependent pitman-yor processes. In *NIPS*, pp. 1585–1592, 2008.

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., and Rother, C. A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(6):1068–1080, 2008.

Szummer, M., Kohli, P., and Hoiem, D. Learning CRFs using graph cuts. In *ECCV*, pp. 582–595, 2008.

Tarlow, D. and Zemel, R. Big and tall: Large margin learning with high order losses. In *CVPR 2011 Workshop on Inference in Graphical Models with Structured Potentials*, 2011.

Taskar, B., Guestrin, C., and Koller, D. Max-Margin Markov Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

Woodford, O., Rother, C., and Kolmogorov, V. A global perspective on MAP inference for low-level vision. In *ICCV*, 2009.

A Submodularity

For the formal definition of submodular functions, consider a function $f(\cdot)$ that is defined over the set of variables $\boldsymbol{y} = \{y_1, y_2, ..., y_n\}$ where each y_i takes values from the label set $\mathcal{L} = \{l_1, l_2, ...l_L\}$. Then, given an *ordering* over the label set $\mathcal{L}$, the function $f(\cdot)$ is submodular if all its projections[3] on two variables satisfy the constraint:

$$f^p(a,b) + f^p(a+1,b+1) \leq f^p(a,b+1) + f^p(a+1,b), \quad (17)$$

for all $a, b \in \mathcal{L}$.

[3] A *projection* of any function $f(\cdot)$ is a function f^p which is obtained by fixing the values of some of the arguments of $f(\cdot)$. For instance, fixing the value of k variables of the function $f_1 : \mathbb{R}^n \to \mathbb{R}$ produces the projection $f_1^p : \mathbb{R}^{n-k} \to \mathbb{R}$.