Machine Learning and Probabilistic Models of Vision

Alan Yuille and Xuming He

Abstract

This paper draws connections between Probabilistic Methods and Machine Learning Approaches. In particular, we show that many Support Vector Machine criteria – binary, multi-class, and latent – can be obtained as upper bound approximations to standard probabilistic formulations. The advantage of these 'Machine Learning bounds' is that it greatly simplifies the computation and, possibly, may yield greater robustness. These connections enable us to take complex models formulated in terms of probabilistic distribution defined over graph structure and approximate/bound them by machine learning techniques. We illustrate this with examples from the literature which are applied to a range of vision problems — including image labeling, object detection and parsing, and motion estimation – and which achieve state of the art results.

I. Introduction

Machine learning techniques are sometimes proposed as radical alternatives to standard probabilistic methods. In this paper we show how many machine learning methods can be obtained as bounds of probabilistic approaches. In particular, this enables us to apply machine learning methods to problems that can be formulated as probabilistic inference over graphical models. This is particularly important for vision applications, because these graphical models are able to capture the spatial relations which are often neglected in machine learning approaches (because non-visual data often does not have such relations). We will illustrate the advantages by describing state of the art results from the literature for image labeling, object detection and parsing, and motion estimation.

The advance is conceptual. It enables us to relate practical machine learning methods with the sophistication representation of probabilistic models defined on graph structures. This may help probabilistic theories by obtaining good classes of approximations and bounds. Conversely it may enable machine learning methods to be extended to a larger range of problems. Our results also show that machine learning intuitions – such as the desirability of maximizing margins to ensure generalization – correspond to fitting probabilistic model while encouraging large variance (which also helps prevent over-fitting).

Papers by Hastie, Friedman, Tishbirani [1] have helped illuminate some relations between machine learning models and standard probabilistic estimation – e.g., by showing that AdaBoost learning converges to a posterior distribution in the limit of large amounts of data. Poggio and Smale [2] formulate learning as function approximation using regularization, state that the maximizing margin criteria can be re-expressed in terms of regularization, and point out the connections to Bayesian formulations (but there are some limitations – they do not have a separate loss function and probability model), while we have the freedom to specify them separately, and neither do they have models with hidden variables). Nevertheless, the vast majority of the literature starts from the standard perspective based on margins and empirical loss functions. (CHECK: Lafferty and Lebanon!!).

We stress that our goal is to relate probabilistic models defined over graphs to machine learning methods based on maximizing margins and using support vector methods. There is, of course, a large 'machine learning' literature such as boltzmann machine and deep belief networks which is already formulated in terms of probability distributions defined on graphs (REFS!!).

II. Background: Support Vector Machines with Multi-Variate and Hidden variables.

We first discuss binary classification and the relationships between support vector machine methods and probabilistic approaches. It will be helpful to show our results for this important special case. It enables

us to introduce our results, and their intuitions, without being burdened by too much notation. Also it enables us introduce notation which can easily be extended to the more complex cases involving multiple variables and hidden variables.

Binary classification starts with training data $\{(y_i, x_i) : i = 1, ..., N\}$, where the output $y_i \in \{-1, +1\}$ is binary-valued and the input x_i can be anything (for vision applications, it may be the intensity values defined over an image window).

The goal of binary classification is to be able to divide the input data into two classes by a decision rule of form:

$$\hat{y} = \arg\max yw \cdot \phi(x).$$

This classifies input data x as positive if $w \cdot \phi(x) > 0$ and negative otherwise. An important special case is when $\phi(x) = (1, x)$ and $w = (w_0, w_1)$, then the decision rule is specified by whether the input data x is above, or below, the hyperplane $w_0 + w_1 \cdot x = 0$. The use of more general functions $\phi(x)$ enables the use of the kernel trick (ref!!).

Support Vector Machines specify a criterion to determine the optimal w from the training dataset. The standard criterion requests minimizing the following criterion:

$$L(w) = \frac{1}{2}|w|^2 + C\sum_{i=1}^{N} \max\{0, 1 - y_i w \cdot \phi(x_i)\}$$

This criterion maximizes the margin $1/|w|$ while requiring that data points on the wrong side of the margin – i.e. $y_i w \cdot \phi(x_i) < 1$ – pay a penalty $C\{1 - y_i w \cdot \phi(x_i)\} > 0$.

This can be re-expressed in terms of function $L(w, \zeta : \alpha, \gamma)$ which includes slack variables $\zeta_i\}$ and lagrange parameters $\{\alpha_i, \gamma_i\}$. This function $L(w, \zeta : \alpha, \gamma)$ should be minimized with respect to w, ζ and maximized with respect to α, γ and takes the form:

$$L(w, \zeta : \alpha, \gamma) = \frac{1}{2}|w|^2 + C\sum_{i=1}^{N} \zeta_i - \sum_{i=1}^{N} \alpha_i(y_i w \cdot \phi(x_i) - 1 + \zeta_i) - \sum_{i=1}^{N} \alpha_i \zeta_i$$

The second term penalizes the amount of slack variables used, the third term ensures that datapoints are on the correct side of the margin after 'borrowing' the slack variable, and the fourth term constrains the slack variables to be non-negative. We can obtain our original formulation by observing that the slack variable ζ_i should be zero unless the data point (x_i, y_i) is on the wrong side of the discriminant, in which case it takes value $1 - y_i w \cdot \phi(x_i)$.

The re-formulation is useful because: (I) it leads directly to the concept of support vectors – differentiating wrt w yields $w = \sum_{i=1}^{N} \alpha_i y_i \phi(x_i)$ – hence w is expressed in terms of those data points (x_i, y_i), the support vectors, for which $\alpha_i \neq 0$. (II) We can eliminate w, ζ and obtain a dual formulation which is a function of α only – hence we can solve by maximizing the dual to estimate α and using the result above to solve for w. (III) The form of final decision rule $\hat{y}(x) = \arg\max yw \cdot \phi(x)$, taken with the expression for w in terms of the support vectors shows that $\hat{y}(x) = \arg\max y \sum_{i=1}^{N} \alpha_i y_i \phi(x_i) \cdot \phi(x)$, and hence depends of the features $\phi(x)$ only in terms of their dot product which is the kernel $K(x, x_i) = \phi(x_i) \cdot \phi(x)$.

In this paper, we will only be concerned with the first formulation of the max-margin criterion. But it is important to realize that the connection to the second formulation is direct, and allows concepts like support vectors and the kernel trick to arrive naturally.

We now describe the more general formulation which allows the extension of max-margin criteria to cases where y is multi-class or multi-variate, and where there can also be hidden variables h. The full case, with multi-variate y and hidden variables h, will correspond to the general case of probabilistic inference on graphs.

First consider, multi-class or multi-variate y but without hidden variables. This can correspond to classifying an object as one of several different classes, or classifying a sequence of letters (Koller,

Tasker), or supervised learning a stochastic grammar (Koller, Manning, Tasker) – and we will describe vision applications later.

The problem is formulation as estimating $\hat{y}(x) = \arg\max_y w \cdot \phi(x, y)$, where the potentials $\phi(.,.)$ now contain both x and y as arguments. The criterion is of form:

$$L(w) = \frac{1}{2}|w|^2 + C\sum_{i=1}^{N}\{\max_y\{l(y_i, y) + w \cdot \phi(x, y)\} - w \cdot \phi(x_i, y_i)\}.$$

Here $l(y_i, y)$ is a loss function which pays a penalty depending on the difference between the solution y_i and the one found by the algorithm y.

This is a natural generalization of (!!) and so there also exists a formulation with lagrange parameters, support vectors, the kernel trick, and a dual formulation.

To obtain the binary formulation as a special case, we restrict $y \in \{-1, 1\}$ and express $\phi(x, y) = (1/2)y\phi(x) + c(x)$, where $c(x)$ can be dropped (or can it??). Then estimating $\hat{y}(x) = \arg\max_y w \cdot \phi(x, y)$ reduces to $\hat{y}(x) = \arg\max_y yw \cdot \phi(x)$ as above. We now compare the max-margin criteria and, in particular, the terms $\max\{0, 1 - y_i w \cdot \phi(x_i)\}$ with $\max_y\{l(y_i, y) + w \cdot \phi(x_i, y)\} - w \cdot \phi(x_i, y_i)$. For the later term, we consider the two possibilities – either the $\max_y$ occurs for y_i or for $-y_i$. If it occurs for y_i, and $l(y_i, y_i) = 0$, then we obtain 0 – the same as for the other criteria. But if it occurs for $-y_i$ we obtain $l(-y_i, y_i) + w \cdot \phi(x_i, -y_i) - w\phi(x_i, y_i)$ – which is as before, if we set $l(-y_i, y_i) = 1$ and recall that $\phi(x_i, y_i) = (1/2)y_i\phi(x_i) = -\phi(x_i, -y_i)$.

III. Overview: MultiClass and Multivariate Estimation

We first state a general result for estimating, which we will then reduce to the standard binary-classification task.

Suppose we have observed data $\mathbf{x}$ and output variables $\mathbf{y}$. There is a training dataset $\{(\mathbf{x}_i, \mathbf{y}_i) : i = 1, ..., N\}$. We have a loss function $\exp\{l(\mathbf{y}, \mathbf{y}_i)\}$ which is the cost of making decision $\mathbf{y}$ when the true distribution is $\mathbf{y}_i$.

The *probabilistic approach* is to estimate the parameters $\mathbf{w}$ from the training data. We have a generative model $P(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{1}{Z[\mathbf{x},\mathbf{w}]}\exp\{\mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y})\}$ and a prior $P(\mathbf{w}) = \frac{1}{Z}\exp\{E(\mathbf{w})\}$. The inference algorithm estimate $\hat{\mathbf{y}}(\mathbf{x}) = \arg\max \mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y})$ (note: if we have a loss function, then we should really set $\hat{\mathbf{y}}(\mathbf{x}) = \arg\max_{\tilde{\mathbf{y}}} \sum_{\mathbf{y}} l(\mathbf{y}, \tilde{\mathbf{y}})\mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y})$). We learn the parameters by maximizing a cost function:

$$F_R(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N}\log\sum_{\mathbf{y}}\exp\{l(\mathbf{y}, \mathbf{y}_i)\}P(\mathbf{y}|\mathbf{x}_i, \mathbf{w}). \tag{1}$$

The *SVM machine learning approach* performs the same inference $\hat{\mathbf{y}}(\mathbf{x}) = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y})$. The parameters are learnt by minimizing the cost function:

$$F_{ML}(\mathbf{w}) = \frac{1}{2}|\mathbf{w}|^2 + C\sum_{i=1}^{N}\max_{\mathbf{y}}\{l(\mathbf{y}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})\} - \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i). \tag{2}$$

We will prove that $-(1/C)F_{ML}(\mathbf{w})$ can be used to lower and upper bound $F_R(\mathbf{w})$ provided we set the prior $P(\mathbf{w})$ to be a Gaussian (i.e. identify $E(\mathbf{w})$ with $-(1/2C)|\mathbf{w}|^2$). More precisely, $-F_{ML}(\mathbf{w}) + K_1 \leq CF_R(\mathbf{w}) \leq -F_{ML}(\mathbf{w}) + K_2$ for some constants K_1, K_2.

These bounds are obtained using the property $\max_{\mathbf{y}} f(\mathbf{y}) \leq \log\sum_{\mathbf{y}}\exp\{f(\mathbf{y})\} \leq |Y|\max_{\mathbf{y}} f(\mathbf{y})$ for any function $f(.)$, where $|Y|$ is the number of states of $\mathbf{y}$. This bounds the summation by its largest term (lower bound) or by its largest term multiplied by the length of the series (upper bound). Clearly these bounds are not tight. We will use variants of these bounds on occasion, particularly when we introduce hidden variables.

From this perspective it follows that machine learning (SVM criteria) can be obtained as bounds of the standard probabilistic/bayesian loss functions. Machine learning can be thought of an approximation which helps simplify the computation. The margin criterion $1/2|\mathbf{w}|^2$ simply corresponds to the prior of the probability model. The machine learning requirement that the margin should be maximized to ensure good generalization corresponds to requiring that the learnt probabilistic model obeys the prior – hence has large variance if the margin criterion is $1/2|\mathbf{w}|^2$ (i.e. encourages small $\mathbf{w}$ which means big variance for $P(\mathbf{y}|\mathbf{x},\mathbf{w})$), which also intuitively helps generalization.

But these bounds may, in fact, be better than the original probability models in situations where the probability models are not accurate or where there is not sufficient data to determine them accurately. By using only the largest term in the partition function we are paying most attention to the states which can be most easily misclassified. This is the standard argument for machine learning methods – namely that they pay attention to the data at decision boundaries and do not attempt to model the data elsewhere.

IV. Case I: No loss function or hidden variables

A. Exponential Models and Bounds

We start by evaluating bounds for exponential models of form:

$$P(\mathbf{y}|\mathbf{x},\mathbf{w}) = \frac{\exp\{\mathbf{w}\cdot\phi(\mathbf{x},\mathbf{y})\}}{Z[\mathbf{x},\mathbf{w}]}, \quad Z[\mathbf{x},\mathbf{y}] = \sum_{\mathbf{y}} \exp\{\mathbf{w}\cdot\phi(\mathbf{x},\mathbf{y})\}. \tag{3}$$

We can place a prior $P(\mathbf{w})$ on the model parameters $\mathbf{w}$. This prior can be expressed in the form:

$$P(\mathbf{w}) = \frac{\exp\{E(\mathbf{w})\}}{Z}. \tag{4}$$

Given a set of training examples $\{(\mathbf{x}_i,\mathbf{y}_i) : i = 1,...,N\}$ we can estimate the parameters $\mathbf{w}$ by maximum likelihood (ML) or maximum a posteriori (MAP) estimation. These correspond to maximizing the functions $F_{ML}(\mathbf{w})$ and $F_{MAP}(\mathbf{w})$ respectively:

$$F_{ML}(\mathbf{w}) = \sum_i \{\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i) - \log Z[\mathbf{x}_i,\mathbf{w}]\}, \tag{5}$$

$$F_{MAP}(\mathbf{w}) = E(\mathbf{w}) + \sum_i \{\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i) - \log Z[\mathbf{x}_i,\mathbf{w}]\} \tag{6}$$

To make connections to Machine Learning requires putting bounds on the partition function $Z[\mathbf{x}_i,\mathbf{y}] = \sum_{\mathbf{y}} \exp\{\mathbf{w}\cdot\phi(\mathbf{x},\mathbf{y})\}$. We can obtain lower and upper bounds by considering the dominant term $\mathbf{w}\cdot\phi(\mathbf{x}_i,\hat{\mathbf{y}}_i) = \max_{\mathbf{y}} \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y})$, where $\hat{\mathbf{y}}_i = \arg\max_{\mathbf{y}} \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y})$. This gives:

$$\mathbf{w}\cdot\phi(\mathbf{x},\hat{\mathbf{y}}_i) \leq \log Z[\mathbf{x}_i,\mathbf{w}] \leq \log|Y| + \mathbf{w}\cdot\phi(\mathbf{x},\hat{\mathbf{y}}_i), \tag{7}$$

where $|Y|$ is the number of possible states of $\mathbf{y}$. Note that these bounds differ only by a constant $\log|Y|$ and so both are equivalent when we seek to extremize with respect to $\mathbf{w}$.

Hence we can bound the MAP criteria by the following criteria:

$$F^{Bound}_{MAP} = E(\mathbf{w}) + \sum_{i=1}^{N} \{\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{w}\cdot(\mathbf{x}_i,\hat{\mathbf{y}}_i)\}. \tag{8}$$

Note: we can clearly get tighter bounds by considering other terms in the summation of $Z[\mathbf{x},\mathbf{w}]$ (instead of only the maximum term). The bounds will become tight in the limit as $|\mathbf{w}| \mapsto \infty$, but we will usually not be in this regime (as we will see).

B. Support Vector Machines

Support vector machines can be formulated by assuming that the solution can be expressed in form $\hat{\mathbf{y}}(\mathbf{x}) = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y})$ where the weights $\mathbf{w}$ are chosen to minimize the following function:

$$F_{SVM}(\mathbf{w}) = \frac{1}{2}|\mathbf{w}|^2 + C \sum_{i=1}^{N} \sum_{i=1}^{N} \zeta_i, \tag{9}$$

subject to the constraints that

$$\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}) - \zeta_i, \quad \forall \mathbf{y}. \tag{10}$$

Here ζ_i is a slack variable which is constrained to be non-negative. It follows that ζ_i must satisfy:

$$\zeta_i \geq \max_{\mathbf{y}} \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}) - \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i), \ \forall \mathbf{y}. \tag{11}$$

Hence, we can re-express $F_{SVM}(\mathbf{w})$ as:

$$F_{SVM}(\mathbf{w}) = \frac{1}{2}|\mathbf{w}|^2 + C \sum_{i=1}^{N} \{\mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i) - \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i)\}. \tag{12}$$

Note: we can obtain the standard dual formulation of support vector machines by introducing lagrange parameters $\{\alpha_i\}$ to impose the constraints in equation (10). It follows directly that the solutions can be expressed in terms of support vectors.

C. Comparing Probability Bounds with SVMs

By comparing equations (12) and (8), we see that we can derive the SVM formulation by choosing a Gaussian prior $P(\mathbf{w}) = \frac{1}{Z} \exp\{-\frac{|\mathbf{w}|^2}{2C}\}$ and using either the upper or lower bound given in equation (7). Changing the margin term $1/2|\mathbf{w}|^2$ in the SVM criterion (e.g., by altering it to an L-1 penalty $|\mathbf{w}|$) will simply correspond to altering the prior.

Observe that the margin term $1/2|\mathbf{w}|^2$ is used by SVMs to make the margin as big as possible in order to help generalization to data in the testing set. From our probabilistic perspective, this prior encourages the conditional distribution in equation (13) to have as large a variance as possible (or to have as high a temperature as possible for statistical physicists). This clearly make sense from the point of view of generalization to novel data – a model with higher variance is less likely to overfit the training data.

The differences between the full probabilistic criterion and their bounds (i.e., the SVM criteria) arise from replacing the terms $\mathbf{w} \cdot (\mathbf{x}_i, \mathbf{y}_i) - \log \sum_{\mathbf{y}} \exp\{\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})\}$ by $\mathbf{w} \cdot (\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i)$. For standard MAP we have to consider the values of $\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})$ for all $\mathbf{y}$, while for the bound we only need consider the maximum value $\mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i)$. This partially explains why we get support vectors for SVM, but not for MAP – we only have to consider one possible alternative state $\hat{\mathbf{y}}_i$ instead of all possible states. (Not really true – it doesn't explain why the best solution $\hat{\mathbf{w}}$ only depends on some of the data examples).

V. Extension to Hidden Variables

We now introduce hidden variables $\mathbf{h}$ which are unobserved during the training data. We extend the probabilistic model to:

$$P(\mathbf{y}, \mathbf{h}|\mathbf{x}, \mathbf{w}) = \frac{\exp\{\mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y}, \mathbf{h})\}}{Z[\mathbf{x}, \mathbf{w}]}, \quad Z[\mathbf{x}, \mathbf{y}] = \sum_{\mathbf{y}} \exp\{\mathbf{w} \cdot \phi(\mathbf{x}, \mathbf{y}, \mathbf{h})\}. \tag{13}$$

The MAP formulation can be expressed as maximizing:

$$F^H_{MAP}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N} \log \sum_{bh} P(\mathbf{y}_i, \mathbf{h}|\mathbf{x}_i, \mathbf{w}). \tag{14}$$

We can bound $\sum_{bh} P(\mathbf{y}_i, \mathbf{h}|\mathbf{x}_i, \mathbf{w})$ below and above by using $\tilde{\mathbf{h}}_i = \arg\max P(\mathbf{y}_i, \mathbf{h}|\mathbf{x}_i, \mathbf{w})$ and H (total number of hidden states):

$$\log P(\mathbf{y}_i, \tilde{\mathbf{h}}_i|\mathbf{x}_i, \mathbf{w}) \leq \log \sum_{bh} P(\mathbf{y}_i, \mathbf{h}|\mathbf{x}_i, \mathbf{w}) \leq \log |H| + \log P(\mathbf{y}_i, \tilde{\mathbf{h}}_i|\mathbf{x}_i, \mathbf{w}) \tag{15}$$

We can express $\log P(\mathbf{y}_i, \tilde{\mathbf{h}}_i|\mathbf{x}_i, \mathbf{w})$ as $\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i, \tilde{\mathbf{h}}_i) - \log \sum_{\mathbf{y},\mathbf{h}} \exp\{\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}, \mathbf{h})\}$. We can bound $\log Z[\mathbf{x}_i, \mathbf{w}] = \log \sum_{\mathbf{y},\mathbf{h}} \exp\{\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}, \mathbf{h})\}$ below and above by:

$$\mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i) \leq \log Z[\mathbf{x}_i, \mathbf{w}] \leq \log |H| + \log |Y| + \mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i). \tag{16}$$

Hence we can bound $F^H_{MAP}(\mathbf{w})$ below and above (with different constant terms) by the quantity:

$$F^{H,B}_{MAP}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N} \{\mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i, \tilde{\mathbf{h}}_i) - \mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i)\}$$

$$F^{H,B}_{MAP}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N} \{\max_{\mathbf{h}} \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i) - \max_{\mathbf{y},\mathbf{h}} \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}, \mathbf{h}_i)\}. \tag{17}$$

After changing the sign, and setting $E(\mathbf{w}) = \frac{1}{2C}|\mathbf{w}|^2$, we obtain the latent SVM formulation with zero loss function.

Note that there are two bounds required here. First, we bound the sum over hidden variables by using the most probable hidden variable. Secondly, we bound the partition function by similar methods as before.

Note: the probabilistic model uses the EM algorithm (or a stochastic variant like Data Augmentation) to deal with the hidden variables. The machine learning criterion can be expressed as the sum of convex and concave parts, which can be minimized (local minima only) using the CCCP algorithm. This is a two step algorithm which is similar to running EM at zero temperature.

VI. Loss Functions

We now show how to include loss functions into the formulation. We first study the case without hidden variables and then introduce them.

We extend the probabilistic formulation by including a loss function $\exp l(\mathbf{y}, \mathbf{y}_i)$ which is the loss of making decision $\mathbf{y}$ when the true state is $\mathbf{y}_i$. The maximum likelihood criterion is replaced by the risk $R(\mathbf{w})$ defined by:

$$\begin{aligned} R(\mathbf{w}) &= \sum_{i=1}^{N} \log \sum_{\mathbf{y}} \exp\{l(\mathbf{y}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}) - \log Z[\mathbf{x}_i, \mathbf{w}]\}, \\ &= \sum_{i=1}^{N} \{-\log Z[\mathbf{x}_i, \mathbf{w}] + \sum_{\mathbf{y}} \exp\{l(\mathbf{y}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})\}\}. \end{aligned} \tag{18}$$

We can obtain lower and upper bounds for both terms in the summation. The bounds for $\log Z[\mathbf{x}_i, \mathbf{w}]$ are obtained as before in terms of $\max_{\mathbf{y}} \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y}) = \mathbf{w} \cdot \phi(\mathbf{x}_i, \tilde{\mathbf{y}}_i)$, where $\tilde{\mathbf{y}}_i = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})$. The bounds for the $\sum_{\mathbf{y}}$ terms are obtained similarly to those for hidden variables in the previous section, by computing $\max_{\mathbf{y}}\{l(\mathbf{y}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})\} = l(\hat{\mathbf{y}}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \hat{\mathbf{y}})$ where $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} l(\mathbf{y}, \mathbf{y}_i) + \mathbf{w} \cdot \phi(\mathbf{x}_i, \mathbf{y})$.

This gives us bounds

$$F^B_{Loss}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N}\{-\mathbf{w}\cdot\phi(\mathbf{x}_i,\tilde{\mathbf{y}}_i) + l(\hat{\mathbf{y}},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\hat{\mathbf{y}})\},$$

$$F^B_{Loss}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N}\{-\max_{\mathbf{y}}\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i) + \max\{l(\mathbf{y},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y})\}\}. \tag{19}$$

There is one problem – because of the sign difference of these two terms – the cost function is not concave (or convex!!). This also occurred for the latent SVM models in the previous section. Hence we cannot find an algorithm that is guaranteed to converge to the global minimum.

We can obtain a looser bound which is concave – and which will relate directly to the standard SVM formulations. This is obtained by bounding $\log Z[\mathbf{x}_i,\mathbf{w}] \geq \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i)$. This will generally be a looser bound although, of course, we hope that after learning that $\mathbf{y}_i \approx \tilde{\mathbf{y}}$. We can only use this to give a lower bound for $\log Z[\mathbf{x}_i,\mathbf{w}]$ which translates into an upper bound for $F_{Loss}(\mathbf{w})$. It is given by:

$$F^B_{Loss}(\mathbf{w}) = E(\mathbf{w}) + \sum_{i=1}^{N}\{-\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i) + l(\hat{\mathbf{y}},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\hat{\mathbf{y}})\}. \tag{20}$$

It is straightforward to check that this is negative of the standard SVM criterion.

Note that this criterion is usually derived by considering the empirical risk $1/2|\mathbf{w}|^2 + C\sum_{i=1}^{N} l(\mathbf{y}_i,\hat{\mathbf{y}}_i)$ where $\hat{\mathbf{y}}_i = \arg\max \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y})$. Then a series of bounds (from Joachim's group – see my course notes) are used to derive the SVM criterion (the empirical risk is a non-convex function of $\mathbf{w}$ and so it is not easy to minimize).

VII. Everything Together

We now consider the full problem where we have a loss function and hidden variables.

The risk for each sample $\mathbf{x}_i,\mathbf{y}_i$ is given by

$$\begin{aligned}\log\sum_{\mathbf{y}}\exp\{l(\mathbf{y},\mathbf{y}_i)\}P(\mathbf{y}|\mathbf{x}_i,\mathbf{w}) &= \log\sum_{\mathbf{y},\mathbf{h}}\exp\{l(\mathbf{y},\mathbf{y}_i)\}P(\mathbf{y},\mathbf{h}|\mathbf{x}_i,\mathbf{w}),\\ &= -\log Z[\mathbf{x}_i,\mathbf{w}] + \log\sum_{\mathbf{y},\mathbf{h}}\exp\{l(\mathbf{y},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y},\mathbf{h})\}.\end{aligned} \tag{21}$$

We can use our (now) standard techniques to put upper and lower bounds on both terms. This gives:

$$-\max_{\mathbf{y},\mathbf{h}}\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y},\mathbf{h}) + \max_{\mathbf{y},\mathbf{h}}\{l(\mathbf{y},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y},\mathbf{h})\}. \tag{22}$$

Alternatively, we can obtain an upper bound only by a simplification (similar to the previous section) where we replace the first term by $-\max_{\mathbf{h}}\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{h})$ (i.e. plug in $\mathbf{y}_i$ instead of maximizing with respect to $\mathbf{y}$). This gives terms of form:

$$-\max_{\mathbf{h}}\mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{h}) + \max_{\mathbf{y},\mathbf{h}}\{l(\mathbf{y},\mathbf{y}_i) + \mathbf{w}\cdot\phi(\mathbf{x}_i,\mathbf{y},\mathbf{h})\}, \tag{23}$$

which are those used in latent SVMs.

$$F_{ML}(\mathbf{w}) = \frac{1}{2}|\mathbf{w}|^2 + C\sum_{i=1}^{N} l(\hat{\mathbf{y}}(\mathbf{x}_i),\mathbf{y}_i). \tag{24}$$

VIII. EXAMPLES

We now report examples for computer vision for image labeling, object detection, object parsing, and motion estimation. These methods typically yield state of the art performance.

Probabilistic models on structured representations are defined by specifying a probability distribution over a graph structure. The graph is organized into layers, where the bottom layer is the input. Each node is connected by edges to a neighborhood of nodes. These include sibling nodes at the same layer and child/parent nodes at the layers above and below. In addition, there can be connections to the input layer l=0. State variables are defined at each node. We define potentials and parameters over the edges. These specify prior constraints between the node and its siblings (horizontal) and children (vertical).

For objects RCMs, the state variables represent the poses of subparts of objects – for applications involving objects (i.e. detection, localization, parsing, recognition). specify the ways that parts can connect with subparts. The vertical potentials encode spatial relationships between the poses of parts and their subparts. The horizontal potential terms encode spatial relations between parts of an object at similar levels of the hierarchy. Finally the data terms are the direct input to a graph node from the input image. For image labeling and segmentation tasks, the state variables represent segmentation templates which give descriptions of the image at different levels of resolution. The potential terms encode prior knowledge about likely image structures – e.g. edges tends to be straight, cows tend to be above grass and below sky, and image descriptions at different resolution levels should be consistent.

$$\begin{aligned}
\text{Graph} : (\mathcal{V}, \mathcal{E}) : \ & \mathcal{V} \text{ nodes}, \ \mathcal{E} \text{ edges}. \\
& \mathcal{V}^l : \ \text{nodes level } l. \\
\text{Children } ch(\mu) \subset \mathcal{V}^{l-1}, & \text{siblings } sib(\mu) \subset \mathcal{V}^l. \\
& \text{State variables} : w_\mu \\
w_{ch(\mu)}, w_{sib(\mu)} & \text{ states of children and siblings.}
\end{aligned}$$

$$\begin{aligned}
\text{Vertical Potentials } \boldsymbol{\phi}^V(w_\mu, w_{ch(\mu)}) : & \ \text{Weights } \boldsymbol{\lambda}^V_\mu. \\
\text{Horizontal Potentials } \boldsymbol{\phi}^H(w_\mu, w_{sib(\mu)}) : & \ \text{Weights } \boldsymbol{\lambda}^H_\mu. \\
\text{Data Potentials } \boldsymbol{\phi}^D(w_\mu, \mathbf{I}) : & \ \text{Weights } \boldsymbol{\lambda}^D_\mu.
\end{aligned}$$

$$\begin{aligned}
P(\mathbf{W}|\mathbf{I}) = \frac{1}{Z[\boldsymbol{\lambda}^D, \mathbf{I}]} \times \\
\exp\{-\sum_{\mu \in \mathcal{G}} \boldsymbol{\lambda}^D_\mu \cdot \boldsymbol{\phi}^D(w_\mu, \mathbf{I}) - \sum_{\mu \in \mathcal{V}} \boldsymbol{\lambda}^P_\mu \cdot^P (w_\mu, w_{nh(\mu)})\}. \\
\sum_{\mu \in \mathcal{V}} \boldsymbol{\lambda}^V_\mu \cdot \boldsymbol{\phi}^V(w_\mu, w_{ch(\mu)}) + \sum_{\mu \in \mathcal{V}} \boldsymbol{\lambda}^H_\mu \cdot \boldsymbol{\phi}^H(w_\mu, w_{sib(\mu)}).
\end{aligned}$$

We show some specific examples.

Image Labeling For image labeling, the graph structure is a quadtree structure, illustrated in figure (**??**). Each node is required to give a description of the image region underneath it. This description is specified by the state w_μ which specifies $w_\mu = (c_\mu, \vec{s}_\mu)$, where $c_\mu \in D$ is a segmentation-template, which partitions the image region into subregions, and $\vec{s}_\mu$ assigns labels to each subregion. These labels come from a set specified by the Microsoft Research Cambridge dataset (23??). The graph structure is hierarchical so that coarse descriptions are obtained at high-levels and more detailed descriptions at the lower-levels. The potentials include data-terms – based on dictionaries of features – and prior terms which are based on dictionaries of potentials which encode compatibility between state variables at different levels of

the hierarchy (but covering the same image region), priors on the segmentation templates and labeling. *Inference* can be performed by pruned dynamic programming exploiting the tree-like structure of the hierarchy. *Learning* is performed by the *structure perceptron* algorithm. As described earlier, this is an approximate algorithm for minimizing a max-margin criterion. No hidden variables are used here, since we can use groundtruth at the pixel level to estimate the states at the graph nodes of the tree. See figure (1).

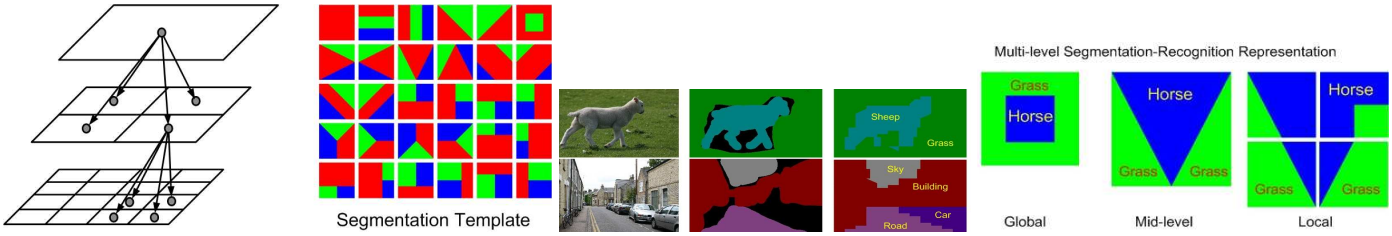

Fig. 1. Left: the graph structure of the recursive compositional model used for image labeling in [**?**]. The state variable at each graph node describes an image region by a segmentation template, chosen from the set of templates (right), with labels for each region of the template. Center: examples on the MRC dataset (image, groundtruth, label specified by model. Right: coarse-to-fine representation of image regions.

Object Detection and Pose Estimation. Another example is used to illustrate this approach for modeling a highly deformable articulated object – baseball players and horses. The object is represented by a compositional model which is composed in terms of elementary parts. The graph structure has variable topology and switch variables, enabling 100 different topologies to be expressed in a single graph. This us essentially a mixture of 100 models expressed much more compactly by exploiting that parts are shared between different poses. *Inference* is performed by pruned dynamic programming which enables rapid search over these 100 different mixture models. *Learning* is performed by structured max-margin. There are no hidden variables since groundtruth is estimated for all levels of the hierarchy. The loss function penalizes configurations where the estimate state of a node differs from the correct (groundtruth) state by more than a threshold.

Motion Estimation. This work is aimed at estimating the motion flow between two images. It is defined on a hierarchical group with closed loops.It contains dataterms which requires the corresponding points in the two images have similar intensity properties and by a prior which encourages smoothness and slowness of the velocity field. This prior has parameters which need to be learnt. *Inference* is done by a variant of loops belief propagation. The *learning* is done by structure perceptron using the groundtruth from the Middlesbury dataset.

Object Detection. We also report a model for detecting objects which represents objects by a mixture of hierarchical tree models. Each model consists of a hierarchy of parts with three layers – 1 part at the top level, 9 at the second level, and 36 at the bottom level. Each part can move with independently. The data terms are determined by HOG features with parameters that must be learnt. Similarly there are prior terms which specify the relative positions of part of parent nodes and of their children. The *inference* is done by a combination of dynamic programming and window search. The *learning* does involve hidden/latent variables since the mixture component (i.e. which mixture model is chosen) and the position of parts are not specified – the groundtruth simply says whether the object is present in an image window or not.

REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Feldman. The Elements of Statistical Learning. Springer. 2009.
[2] T. Poggio and S. Smale. "The Mathematics of Learning: Dealing with Data". American Mathematical Society. 2003.

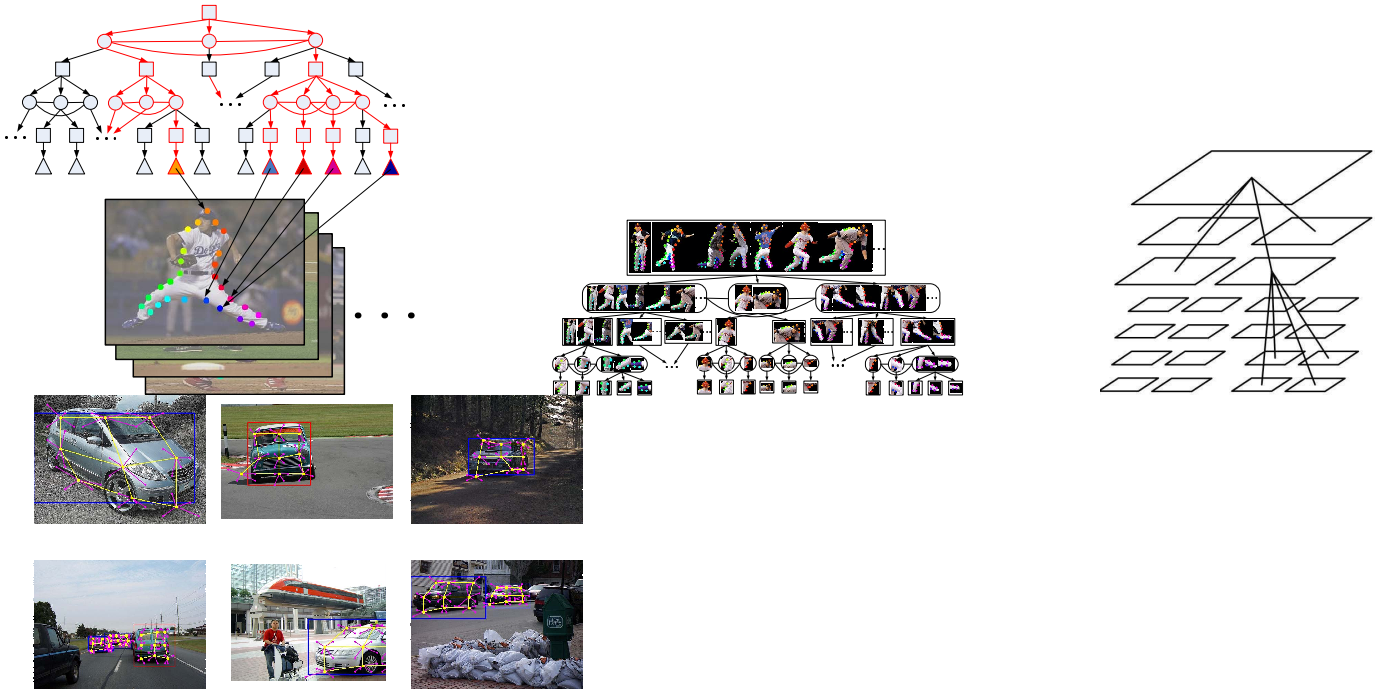

Fig. 2. Left: And/Or Tree graphical models are capable of representing 100 different poses of a baseball player using a compact representation which exploits part sharing [**?**]. Center: RCMs represented by a hierarchical model of parts, where several models (not shown) are used as mixture components to model different poses of objects [**?**]. Right: These models can detect objects from different viewpoints, with changes in geometry (spatial warps) and even partial occlusion. They are trained by Latent SVM techniques where the groundtruth only specifies that an object is present or not (i.e. does not give the positions of the parts or the mixture component). This approach obtained the second best performance on the 2010 Pascal Challenge for object detection (ECCV 2010).