Front. Electr. Electron. Eng.
DOI 10.1007/s11460-012-0170-6

RESEARCH ARTICLE

Alan YUILLE, Xuming HE

Probabilistic models of vision and max-margin methods

Abstract It is attractive to formulate problems in computer vision and related fields in term of probabilistic estimation where the probability models are defined over graphs, such as grammars. The graphical structures, and the state variables defined over them, give a rich knowledge representation which can describe the complex structures of objects and images. The probability distributions defined over the graphs capture the statistical variability of these structures. These probability models can be learnt from training data with limited amounts of supervision. But learning these models suffers from the difficulty of evaluating the normalization constant, or partition function, of the probability distributions which can be extremely computationally demanding. This paper shows that by placing bounds on the normalization constant we can obtain computationally tractable approximations. Surprisingly, for certain choices of loss functions, we obtain many of the standard max-margin criteria used in support vector machines (SVMs) and hence we reduce the learning to standard machine learning methods. We show that many machine learning methods can be obtained in this way as approximations to probabilistic methods including multi-class max-margin, ordinal regression, max-margin Markov networks and parsers, multiple-instance learning, and latent SVM. We illustrate this work by computer vision applications including image labeling, object detection and localization, and motion estimation. We speculate that better results can be obtained by using better bounds and approximations.

Keywords structured prediction, max-margin learning, probabilistic models, loss function

Received October 10, 2011; accepted November 22, 2011

Alan YUILLE (✉), Xuming HE
Department of Statistics, University of California at Los Angeles, Los Angeles, CA 90095, USA
E-mail: yuille@stat.ucla.edu

Alan YUILLE
Department of Brain and Cognitive Engineering, Korea University, Seoul 136-701, Korea

Xuming HE
NICTA Canberra Research Laboratory, Canberra, ACT 2601, Australia

1 Introduction

There has recently been major progress in the development of probability models defined over structured representations including graphs and grammars [1–7]. This interdisciplinary enterprize combines computer science expertise on representations with statistical modeling of probabilities. For example, natural language researchers have defined stochastic context free grammars (SCFGs) by putting probability distributions over context free grammars which encode the hierarchy of nouns, nouns phrases, and so on. The same approaches can be extended to more powerful grammars [2,7]. But the representational advantages of these models are balanced by their computational requirements and, in particular, whether they admit efficient learning algorithms.

In particular, it is attractive to formulate vision as probabilistic inference on structured probability representations. This seems both a natural way in which to deal with the complexities and ambiguities of image patterns [8,9] and also fits into a more unified framework for cognition and artificial intelligence [10]. But vision is a particularly challenging problem to formulate in this manner. The complexity of vision requires distributions defined over very complicated structures and requires principles such as compositionality and the use of graphical models with variable topology [11] and stochastic grammars [5]. We will give a brief review of probabilistic models of vision in Section 2. But these complex probability models also require significant computational requirements both to perform inference and for learning.

By contrast, machine learning techniques are often designed to be computationally tractable [12,13] although they are not as well-principled as probabilistic methods for dealing with complex tasks [14]. But how can we take advantage of the representational power of probabilistic models and the computational tractability of machine learning algorithms? There have been attempts to show relations between them, briefly reviewed in Section 2. There has been success in relating AdaBoost to probabilistic learning [15–18] but despite some interesting results [14,19–22]. The attempts to show connections between max-margin methods [13] and probabilities have not as yet made a big impact on the community.

In this paper, our starting point is a series of works published in the last few years [23–27], which formulated complex vision problems, such as the detection and parsing of objects, the labeling of image pixels, and the estimation of motion, in terms of probability models based on compositional principles. These models then used approximations which enabled the models to be learnt by machine learning techniques based on max-margin criteria. In this paper we clarify the nature of the approximations and show that they can be derived rigorously as bounds on the probabilistic criteria. For certain probabilistic criteria, involving appropriate loss functions, this obtains many standard max-margin criteria with or without hidden variables. This yields interesting relationships between probabilistic and margin learning concepts. For example, machine learning attempts to maximize the margin in order to improve generalization to novel data while probabilistic methods attempt to maximize the variance of the distribution in order to obtain a similar effect (i.e., prevents over-fitting the data), see Fig. 1. The notion of support vectors, i.e., the results that the learnt solution depends only on a subset of the training data, arises only from the approximations made to the partition function. We note that some results of this type are known in the natural language community [7].

Our approach has several advantages. We can relate probabilistic models to existing machine learning methods. We can formulate models of more complex phenomena using the well-principled probabilistic approach and then obtain novel machine learning methods. We can improve machine learning methods, and get better bounds for probability models, by making better approximations and bounds.

The structure of this paper is as follows. We first briefly mention some relevant probabilistic models from the computer vision literature and then sketch work which relates machine learning and probabilistic methods, see Section 2. Next we describe probabilistic models in Section 3 and introduce the types of models we will illustrate in the computer vision section. Then we describe the learning criteria used in probabilistic methods and machine learning and the relationships between them, see Section 4. Next in Section 5 we introduce our approach of deriving bounds for probabilistic models by considering models without hidden/latent variables. Then in Section 6 we extend our approach to other examples where hidden/latent variables are present. Then we illustrate this work on examples from our previous work in computer vision which originally motivated the bounds in this paper, see Section 7.

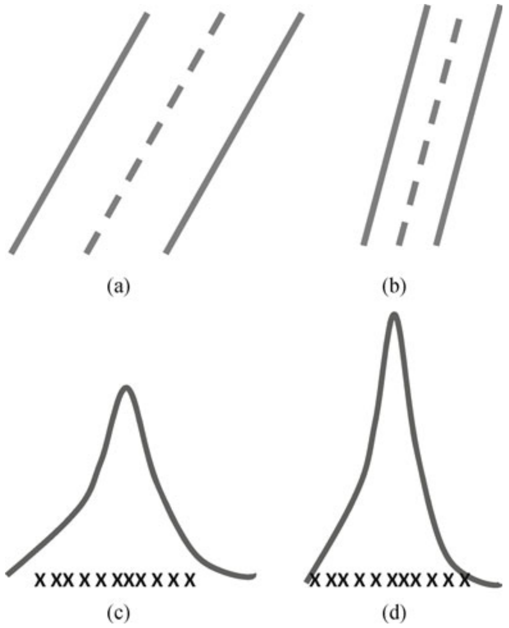


Fig. 1 Max-margin methods try to maximize the margin to encourage generalization preferring large margins (a) to smaller margins (b). For the probability distributions the corresponding intuition is that we will favors models with large covariance (c) over models with smaller covariance (d) in order to discourage over-fitting the data.

2 Background

2.1 Probabilistic models

There has been a long history in computer vision of modeling objects by deformable templates defined in terms of energies [28–30]. More recent models formulate objects and images in terms of probabilistic models defined over structured representations. There is an enormous literature on these topics and only mention here a small sample of some of the most closely related work.

One stream of research follows the inspiration of Grenander's pattern theory [8,9] leading to generative models build using compositionality [11] or stochastic grammars [5]. These generative approaches are computationally demanding despite the exploitation of discriminative models to perform inference by data driven Markov Chain Monte Carlo [31]. Another stream of research leads to discriminative models as exemplified by conditional random fields [32,33]. These have motivated a whole series of works on hierarchical models [34–36]. A particularly important example are latent support vector machine (SVM)'s [37], which we will mention in

Section 5. Other classes of grammatical models includes stochastic grammatical models for text [38] and hierarchical models of objects by Ahuja and Todorovic [39,40]. Yet another related strand of research are deep belief networks [41] and the models by Amit and his colleagues [42].

In this paper we will concern ourselves with a class of models of objects and images [23–27], which are built using the concepts of compositionality and stochastic grammars [5,11]. But these models are learnt discriminatively using the techniques described in this paper. Space did not enable us to also describe the closely related work by Kokkinos and Yuille [43,44], which makes use of multiple instance learning, see Subsection 6.2.

2.2 Machine learning and its relationship to probabilistic modeling

We briefly summarize previous work which addresses the relationship between probabilistic modeling and related methods developed by the machine learning community. In particular, there have been several studies showing that AdaBoost is very similar to sigmoid regression. For example, Hastie et al. [15] showed that AdaBoost converges to a conditional distribution in the limit of large amounts of data. Subsequent work [16–18] showed more detailed relations between the two methods, in particular when an $l1$-norm regularizer is imposed on the weight parameters in AdaBoost.

This paper is concerned with max-margin techniques. The relationship between regularization and max-margin criteria have long been known and are briefly discussed in Ref. [21] which also mentions how regularization can be re-interpreted as Bayesian estimation. Earlier related research on this topic includes the seminar work of Wahba [19]. Seeger [20] gives a thorough discussion of these issues and, in particular, how kernel methods can relate to Gaussian processors (e.g., covariance matrix over finite sets of points can be used to obtain either Gaussian process models or kernels for SVMs). Alternative work [45] is also closely related to Gaussian processes and provides probabilistic classification using an expectation-maximization (EM) training algorithm. Zhu and Xing [22] uses a maximum entropy discrimination framework for learning probabilistic models. More recently, Franc et al. [14] studied the binary-classification case and showed that exact equivalence could be obtained between SVMs and probabilistic leaning for exponential models if certain model parameters, such as the modulus of the weight vector, were treated as hyper-parameters and were estimated differently than the other model parameters. In Ref. [46], Pletscher et al. also suggested a unified view of log-loss and max-margin loss for structured model by introducing an extra temperature hyper-parameter, but did not establish a bounding relations between the different criteria.

In all this work, however, the loss functions are combined into the probability distributions while this paper keeps them separate which give greater flexibility for practical applications. Neither do the studies above address the types of complex models used for our computer vision applications.

3 Probabilistic models

We consider probability models formulated over graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ denotes the vertices and $\mathcal{E}$ denotes edges. State variables y_μ are defined at the nodes $\mu \in \mathcal{V}$. The variable $\boldsymbol{y} = \{y_\mu : \ \mu \in \mathcal{V}\}$ describes the state of the entire graph. The edges $\mathcal{E}$ in the graph specify which nodes are directly connected and define the cliques $\mathcal{C}l$, i.e., for all $\mu_1, \mu_2 \in \mathcal{C}l$ then $(\mu_1, \mu_2) \in \mathcal{E}$. We index $\mathcal{C}l$ by α and let $\boldsymbol{y}_\alpha$ represent the state of all variable in clique α. The computer vision models we consider in this paper has been built from elementary AND and OR cliques, see Fig. 2(a), which can be combined into larger models by composition, see Fig. 2(b). For example, the head of the baseball player in Fig. 3(b) can either be upright or oriented, while the object can be modeled by AND-ing the legs, the torso and the head. The specific models are given in Refs. [23–27] and will be given as illustrations in Section 7.

Potential functions $\boldsymbol{\phi}_\alpha(\boldsymbol{y}_\alpha, \boldsymbol{x})$ and parameters $\boldsymbol{w}_\alpha$ are defined over the cliques $\alpha \in \mathcal{C}l$ (note that these potentials allow direct input from the data $\boldsymbol{x}$). AND and OR nodes will have different types of potential functions. We also define potentials which depend on the states of individual graph nodes and are dependent only on the input $\boldsymbol{x}$.

Examples of the probability models are given in Fig. 3. The nodes $\mu \in \mathcal{V}$ represent subparts of the object where states y_μ specify whether the part is present/absent and, if it is present, its position and other attributes (e.g., orientation and size). For our computer vision applications the graphs are typically organized in layers where a node at one layer is connected to a subset of nodes at the lower level forming a clique. These cliques are of two types: (i) AND-cliques where the upper node represent the position/attributes of a part which is composed from subparts represented by the lower nodes, and (ii) OR-cliques where the upper node takes the same state as one of the lower nodes, i.e., it chooses between them.

We specify a conditional distribution:

$$P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{w}) = \frac{1}{Z[\boldsymbol{w},\boldsymbol{x}]} \times \exp\{\sum_{\mu\in\mathcal{V}} \boldsymbol{w}_\mu \cdot \boldsymbol{\phi}(y_\mu, \boldsymbol{x}) + \sum_{\alpha\in\mathcal{C}l} \boldsymbol{w}_\alpha \cdot \boldsymbol{\phi}_\alpha(\boldsymbol{y}_\alpha, \boldsymbol{x})\}. \quad (1)$$

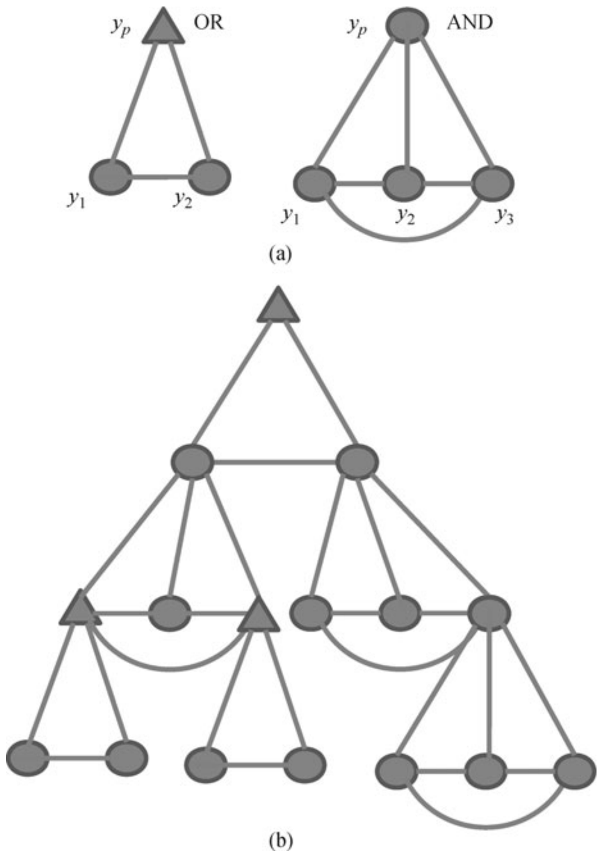


Fig. 2 The basic components of the probability graphs are AND and OR cliques (a). In an AND cliques the parent state y_p represents a part which is composed of elementary parts represented by y_1, y_2, y_3. For OR cliques, the state of the parent node y_p must select one of the child nodes y_1 or y_2. Probability models are constructed by treating AND and OR cliques are elementary components which can be combined to build arbitrarily complex models (b).

These models also involve a loss function $l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$ which is the cost of making decision $\boldsymbol{y}$ if the true state of the graph should be $\boldsymbol{y}_{\mathrm{T}}$. Hence inference requires estimating $\hat{\boldsymbol{y}} = \arg\min_{\boldsymbol{y}} \sum_{\boldsymbol{z}} l(\boldsymbol{y}, \boldsymbol{z}) P(\boldsymbol{z}|\boldsymbol{x} : \boldsymbol{w})$. Depending on the applications we use dynamic programming, junction trees, or approximate algorithms (e.g., belief propagation) to compute $\hat{\boldsymbol{y}}$. For example, the probabilistic model shown in Fig. 2(b) contains closed loops of limited size and hence inference can be performed by the junction trees algorithm.

During training/learning, we are given training examples which specify the states of the input $\{\boldsymbol{x}_i : i = 1, 2, \ldots, N\}$ and a set/subset of the state variables $\{\boldsymbol{y}_i : i = 1, 2, \ldots, N\}$. It is convenient to divide this into two cases. The first case is when all the $\{\boldsymbol{y}_i\}$ are specified. In the second, by a slight abuse of notation, we divide the states variables into $\{(\boldsymbol{y}_i, \boldsymbol{h}_i) : i = 1, 2, \ldots, N\}$ where the states $\{\boldsymbol{y}_i\}$ are specified during learning and the states $\{\boldsymbol{h}_i\}$ are not.

In both cases we end up with exponential models of respective form:

$$P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) = \frac{\exp\{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\}}{Z[\boldsymbol{x}, \boldsymbol{w}]}, \tag{2}$$

$$P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}, \boldsymbol{w}) = \frac{\exp\{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})\}}{Z[\boldsymbol{x}, \boldsymbol{w}]}, \tag{3}$$

where $\boldsymbol{w}$ and $\boldsymbol{\phi}(\cdot)$ are vectors representing all the parameters and potentials described above. In addition we have loss functions $l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$ and $l(\boldsymbol{y}, \boldsymbol{h}, \boldsymbol{y}_{\mathrm{T}})$ respectively.

However, learning the parameters of the models in Eqs. (2) and (3) is computationally demanding due to complex forms of the normalization terms, or partition functions, $Z[\boldsymbol{x}, \boldsymbol{w}]$. This motivates the need for approximate methods which simplify the computation in order to perform learning.

4 Max-margin criteria and probabilistic learning criteria

In this section we consider the max-margin criteria and variants of probabilistic criteria used for learning. The input is a dataset of examples. $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) : i = 1, 2, \ldots, N\}$. Here $\boldsymbol{x}$ is the input, e.g., an image, and $\boldsymbol{y}$ is the output which can either be multi-class, e.g., $\boldsymbol{y} \in \{1, 2, \ldots, M\}$ for some M, or a vector $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_M)$, whose components are multi-valued (as were used for learning Markov models [47], learning

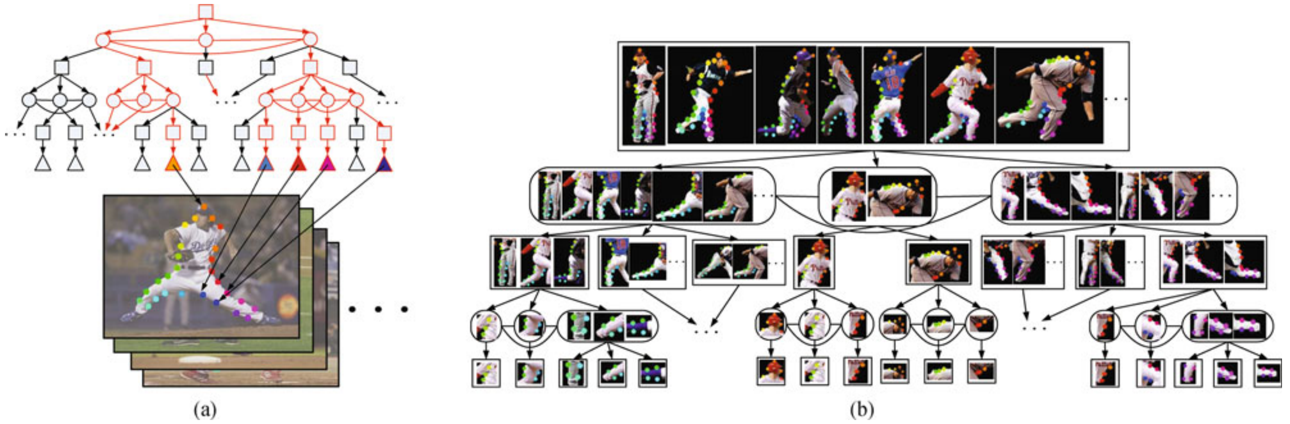


Fig. 3 AND/OR tree graphical models are capable of representing 100 different poses of a baseball player using a compact representation which exploits part sharing (a), and are efficient in representing different appearances of an object (b) [24].

grammars for natural language [48], and in some our computer vision where, for example, y_ν represents the position and attribute of the object part represented by node ν).

The max-machine criteria are designed to seek a decision rule $\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w})$ of form $\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}) = \arg\max_{\boldsymbol{y}} \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y})$ which minimizes a convex bound of the empirical risk obtained by summing a loss function $l(\boldsymbol{y},\boldsymbol{y}_{\mathrm{T}})$ over the data examples:

$$R_{\mathrm{emp}}(\hat{\boldsymbol{y}}) = C\sum_{i=1}^{N} l(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i), \tag{4}$$

combined with a penalty term, e.g., $(1/2)|\boldsymbol{w}|^2$, which regularizes the solution by encouraging a large *margin* $1/|\boldsymbol{w}|$.

A standard bound is of form [49]:

$$F_{\mathrm{ML}}(\boldsymbol{w}) = \frac{1}{2}|\boldsymbol{w}|^2 + C\sum_{i=1}^{N}\max_{\boldsymbol{y}}\{l(\boldsymbol{y},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}) - \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}_i)\}. \tag{5}$$

This can be obtained from the empirical risk (4) by applying the identities $\max_{\boldsymbol{y}}\{l(\boldsymbol{y},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y})\} \geqslant l(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w})))$ and $\boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}))$ [49]. The criterion in Eq. (5) reduces to the hinge loss criterion for binary classification [13] replacing $\boldsymbol{y}$ by $y \in \{-1,+1\}$, expressing the loss function as $l(y,y_{\mathrm{T}}) = 1-\delta_{y,y_{\mathrm{T}}}$, and setting $\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2}y\boldsymbol{\phi}(\boldsymbol{x})$, where $\boldsymbol{\phi}(\boldsymbol{x})$ is a potential in the data $\boldsymbol{x}$ only. The criterion reduces to $\frac{1}{2}|\boldsymbol{w}|^2 + \sum_{i=1}^{N}\max\{0, 1-y_i\boldsymbol{w}\cdot\boldsymbol{x}_i\}$, with solution, after training, given by $\hat{y}(\boldsymbol{x},\boldsymbol{w}) = \arg\max_y \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x})$.

The SVM criterion can be extended to problems where there are hidden/latent variables $\boldsymbol{h}$ (i.e., variables that are not specified by ground-truth during learning). These problems are formulated by introducing potential $\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})$ and expressing the solution as $(\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}),\hat{\boldsymbol{h}}(\boldsymbol{x},\boldsymbol{w})) = \arg\max_{\boldsymbol{y},\boldsymbol{h}} \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})$. The parameters $\boldsymbol{w}$ are learnt by minimizing the *latent SVM criterion* [37,50]:

$$L(\boldsymbol{w}) = \frac{1}{2}|\boldsymbol{w}|^2 + \sum_{i=1}^{N}\{\max_{\boldsymbol{y},\boldsymbol{h}}\{l(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h}_i)\} - \max_{\boldsymbol{h}}\boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}_i,\boldsymbol{h}_i)\}. \tag{6}$$

This criterion is non-convex since the last term $\sum_{i=1}^{N}\max_{\boldsymbol{h}}\boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}_i,\boldsymbol{h}_i)$ is concave, while the remaining terms are convex. We note that the latent SVM used in Ref. [37] is equivalent to the MI-SVM formulation of Multiple Instance Learning with SVM [51]. A similar learning criteria was also proposed by Xu et al. [52] for unsupervised learning problems which maximizes over missing outputs in SVM.

How do the two criteria given in Eqs. (5) and (6) relate to learning probabilistic models? The forms of Eqs. (5) and (6) are suggestive of the exponential models described in Section 3. There are other similarities, e.g., the inference algorithms used for latent SVMs are analogous to the EM algorithm for probability models, as described in Subsection 6.1.

Learning probability models can be formulated as MAP estimation, i.e., seek $\hat{\boldsymbol{w}} = \arg\max P(\boldsymbol{w})\prod_{i=1}^{N}P(\boldsymbol{y}_i|\boldsymbol{x}_i;\boldsymbol{w})$, where $P(\cdot|\cdot;\cdot)$ is specified by Eq. (2) and $P(\boldsymbol{w})$ is a prior on the model parameters $\boldsymbol{w}$. This corresponds to minimizing the log posterior:

$$R_{\mathrm{MAP}}(\boldsymbol{w}) = -\log P(\boldsymbol{w}) - \sum_{i=1}^{N}\log P(\boldsymbol{y}_i|\boldsymbol{x}_i;\boldsymbol{w}). \tag{7}$$

An alternative, which we employ in this paper, involves introducing a loss function $l(\boldsymbol{y},\boldsymbol{y}_i)$ to take into account the penalties that are paid for making an error in $\boldsymbol{y}$ during estimation. This yields the expected loss learning criteria R_{EL} which replaces $-\log P(\boldsymbol{y}_i|\boldsymbol{x}_i;\boldsymbol{w})$ in Eq. (7) by the log of the expected loss $\log\sum_{\boldsymbol{y}} l(\boldsymbol{y},\hat{\boldsymbol{y}}(\boldsymbol{x}))P(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$ which yields:

$$R_{\mathrm{EL}}(\boldsymbol{w}) = -\log P(\boldsymbol{w}) + \sum_{i=1}^{N}\log\sum_{\boldsymbol{y}} l(\boldsymbol{y},\boldsymbol{y}_i)P(\boldsymbol{y}|\boldsymbol{x}_i;\boldsymbol{w}). \tag{8}$$

To understand the motivation for this change, observe that after learning we estimate $\boldsymbol{y}$ from input $\boldsymbol{x}$ by minimizing the expected loss to obtain $\hat{\boldsymbol{y}}(\boldsymbol{x}:\boldsymbol{w}) = \arg\min\sum_{\boldsymbol{y}} l(\boldsymbol{y},\hat{\boldsymbol{y}}(\boldsymbol{x}))P(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$. This reduces to MAP estimation of $\boldsymbol{y}$ in the special case that the loss function $l(\boldsymbol{y},\boldsymbol{y}_{\mathrm{T}}) = 1-\delta_{\boldsymbol{y},\boldsymbol{y}_{\mathrm{T}}}$.

Observe that the probabilistic learning criteria, Eqs. (7) and (8), involve the entire probability distribution while the empirical risk and the max-margin criteria, Eqs. (4) and (5), only involve functions $\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w})$ and $(\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}),\hat{\boldsymbol{h}}(\boldsymbol{x},\boldsymbol{w}))$. There are, however, similarities between the two formulations.

Firstly, note that $\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w})$ and $(\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}),\hat{\boldsymbol{h}}(\boldsymbol{x},\boldsymbol{w}))$ are both maximum likelihood (ML) estimates of $\boldsymbol{y}$ and $\boldsymbol{y}$ from the respective probability models $P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{w})$ and $P(\boldsymbol{y},\boldsymbol{h}|\boldsymbol{x};\boldsymbol{w})$ given in Eqs. (1) and (3). In the limit as the magnitude $|\boldsymbol{w}|$ of the parameters gets large then the probability distributions become strongly peaked about their ML values — $\arg\max_{\boldsymbol{y}}\boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y})$ and $\arg\max_{\boldsymbol{y},\boldsymbol{h}}\boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})$. So in this limit there is little difference between using the full distributions and the decision rules $\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w})$ and $(\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}),\hat{\boldsymbol{h}}(\boldsymbol{x},\boldsymbol{w}))$.

Secondly, we can make a relationship between the criteria using Jensen's inequality. This yields $\sum_{i=1}^{N}\log\sum_{\boldsymbol{y}} l(\boldsymbol{y},\boldsymbol{y}_i)P(\boldsymbol{y}|\boldsymbol{x}_i;\boldsymbol{w}) \geqslant \sum_{i=1}^{N}\sum_{\boldsymbol{y}}\{\log l(\boldsymbol{y},\boldsymbol{y}_i)\}P(\boldsymbol{y}|\boldsymbol{x}_i;\boldsymbol{w})$ which will become equal to the empirical Bayes risk in the large $|\boldsymbol{w}|$ limit but with a different loss function (i.e., replacing $l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$ by $\log l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$).

Hence we can find some relationships between these max-margin and probabilistic criteria in the large $|\boldsymbol{w}|$ limit. But recall that $|\boldsymbol{w}|$ is the inverse of the margin and so large $|\boldsymbol{w}|$ corresponds to the small margin limit which is bad for generalization and which the regularizer is designed to avoid. From the probabilistic perspective, however, large $|\boldsymbol{w}|$ corresponds to small covariance (or low temperature in statistical physics). This is the limit where the probability distribution is most peaked about its maximum value which we will exploit in the next section. As we will see, there will be a tension between the requirements to make $|\boldsymbol{w}|$ large to encourage good generalization and the need to make $|\boldsymbol{w}|$ small to enable the approximations/bounds to be tight.

5 Exponential models, bounds and max-margin criteria

In this section we describe our basic result (which will be extended in the following section). We consider exponential models $P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$ for generating $\boldsymbol{y}$ in terms of input $\boldsymbol{x}$, with parameters $\boldsymbol{w}$, and a prior $P(\boldsymbol{w})$:

$$\begin{aligned} P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) &= \frac{\exp\{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\}}{Z[\boldsymbol{x}, \boldsymbol{w}]}, \\ P(\boldsymbol{w}) &= \frac{\exp\{-E(\boldsymbol{w})\}}{Z}, \end{aligned} \tag{9}$$

where $Z[\boldsymbol{x}, \boldsymbol{w}] = \sum_{\boldsymbol{y}} \exp\{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\}$.

We use the EL $R_{\mathrm{EL}}(\boldsymbol{w})$ which we re-express by setting $l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}}) = \exp \bar{l}(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$:

$$\begin{aligned} LogR_{\mathrm{EL}}(\boldsymbol{w}) &= -\log P(\boldsymbol{w}) \\ &\quad + \sum_{i=1}^{N} \log \sum_{\boldsymbol{y}} \exp\{\bar{l}(\boldsymbol{y}, \boldsymbol{y}_i)\} P(\boldsymbol{y}|\boldsymbol{x}_i, \boldsymbol{w}) \\ &= -\log P(\boldsymbol{w}) + \sum_{i=1}^{N} \{-\log Z[\boldsymbol{x}_i, \boldsymbol{w}] \\ &\quad + \log \sum_{\boldsymbol{y}} \exp\{\bar{l}(\boldsymbol{y}, \boldsymbol{y}_i) + \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})\}\}. \end{aligned} \tag{10}$$

We relate this to the structure SVM criterion by bounding the terms of the right-hand side of Eq. (10). This requires bounding $\log Z[\boldsymbol{x}_i, \boldsymbol{w}]$ and $\log \sum_{\boldsymbol{y}} \exp\{\bar{l}(\boldsymbol{y}, \boldsymbol{y}_i) + \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})\}$. In both cases, this is equivalent to bounding expressions of form $\log \sum_{a=1}^{M} \exp\{n_a\}$ for a set of numbers $\{n_a : a = 1, 2, \ldots, M\}$. *We use two methods to bound expressions of this type*: (i) We can obtain upper and lower bounds of form $n_{\max} \leqslant \log \sum_{a=1}^{M} \exp\{n_a\} \leqslant M n_{\max}$, where $n_{\max} = \max_a n_a$. (ii) We can obtain a weaker lower-bound for this expression by $n_b \leqslant \sum_{a=1}^{M} \exp\{n_a\}$ when n_b is any element of the set $\{n_a : a = 1, 2, \ldots, M\}$.

Using these bounding methods, we can now obtain two bounds for the log-risk of form:

$$\begin{aligned} LogR_{\mathrm{EL}}(\boldsymbol{w}) &\sim E(\boldsymbol{w}) \\ &\quad + \sum_{i=1}^{N} \max_{\boldsymbol{y}} \{\bar{l}(\boldsymbol{y}, \boldsymbol{y}_i) + \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})\} \\ &\quad - \sum_{i=1}^{N} \max_{\boldsymbol{y}} \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}), \end{aligned} \tag{11}$$

$$\begin{aligned} LogR_{\mathrm{EL}}(\boldsymbol{w}) &\leqslant E(\boldsymbol{w}) \\ &\quad + \sum_{i=1}^{N} \max_{\boldsymbol{y}} \{\bar{l}(\boldsymbol{y}, \boldsymbol{y}_i) + \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})\} \\ &\quad - \sum_{i=1}^{N} \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) + \overline{K}. \end{aligned} \tag{12}$$

The first bound, Eq. (11), is obtained by using the first bounding method for both the $\log Z$ and $\log \sum_{\boldsymbol{y}}$ terms. It gives upper and lower bounds for $LogR(\boldsymbol{w})$, indicated by $\sim$. The second bound, Eq. (12), uses the first bounding method for the $\log \sum_{\boldsymbol{y}}$ term and the second bounding method for $\log Z$ ($\overline{K}$ is a constant).

We see that the second bounds, Eq. (12), are identical to the structure SVM criterion provided we set $E(\boldsymbol{w}) = |\boldsymbol{w}|^2$ (i.e., assume a Gaussian prior). The first method, Eq. (11), provides a tighter upper bound, as well as a lower bound. But it has the disadvantage that the bound is not convex, since the final term is concave, and so it is harder to minimize.

The tightness of these bounds depends on the size of $|\boldsymbol{w}|$. In the large $|\boldsymbol{w}|$ limit the upper bound of bounding method (I) becomes exact because the largest term in the summation dominates. But this is the limit where the margin is very small and so generalization is poor.

These results enable us to relate probabilistic models to existing machine learning methods, which enables us to exploit the efficient learning algorithms such as structured perceptron [53] and structure max-margin algorithms [47–49]. Observing from this perspective, the regularizer term which attempts to make a large margin corresponds to a prior on the parameter $\boldsymbol{w}$ which tries to make the covariance as large as possible, hence yielding good generalization. Observe that support vector arise in this approximation although they do not occur in the original probability model — instead they result from replacing the partition functions by its dominant term which concentrates attention on those datapoints with high probability which hence are near the decision boundary. It is also possible to follow the logic of this approach and obtain better bounds by making better approximation to the partition function perhaps by using structured variational methods [54].

6 Further examples

This section describes how the bounding methods can be applied to obtain other max-margin criteria. This includes methods with hidden/latent variables, such as latent SVM and multiple instance learning (MIL), and ordinal regression. If hidden variables are present then we bound the summations over them in a similar manner.

6.1 Hidden variables and latent SVM

We now extend our analysis to deal with models with hidden/latent variables $\boldsymbol{h}$, which are un-specified in the training data. We extend the probabilistic model to

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

We define an exponential loss function $\bar{l}(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_{\mathrm{T}}) = \exp\{l(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_{\mathrm{T}})\}$ and modify the log-risk, Eq. (10), to be

$$\begin{aligned}
&LogR_{\mathrm{EL}}(\boldsymbol{w})\\
&= -\log P(\boldsymbol{w})\\
&\quad + \sum_{i=1}^{N} \log \sum_{\boldsymbol{y},\boldsymbol{h}} \exp \bar{l}(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_i) P(\boldsymbol{y},\boldsymbol{h}|\boldsymbol{x}_i,\boldsymbol{w})\\
&= -\log P(\boldsymbol{w})\\
&\quad + \sum_{i=1}^{N} \log \sum_{\boldsymbol{y},\boldsymbol{h}} \exp\{\bar{l}(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h})\}\\
&\quad - \log Z[\boldsymbol{x},\boldsymbol{w}]. \qquad (14)
\end{aligned}$$

Using our bounding methods, we can obtain two bounds for the log-risk:

$$\begin{aligned}
LogR_{\mathrm{EL}}(\boldsymbol{w}) \sim{}& E(\boldsymbol{w})\\
&+ \sum_{i=1}^{N} \max_{\boldsymbol{y},\boldsymbol{h}} \{\bar{l}(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h})\}\\
&- \sum_{i=1}^{N} \max_{\boldsymbol{y},\boldsymbol{h}} \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h}), \qquad (15)\\
LogR_{\mathrm{EL}}(\boldsymbol{w}) \leqslant{}& E(\boldsymbol{w})\\
&+ \sum_{i=1}^{N} \max_{\boldsymbol{y},\boldsymbol{h}} \{\bar{l}(\boldsymbol{y},\boldsymbol{h},\boldsymbol{y}_i) + \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h})\}\\
&- \sum_{i=1}^{N} \max_{\boldsymbol{h}} \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}_i,\boldsymbol{h}). \qquad (16)
\end{aligned}$$

The first bound, Eq. (15), is obtained by using the first bounding method for both variables $\boldsymbol{y},\boldsymbol{h}$ in the $\log Z$ and $\log\sum_{\boldsymbol{y},\boldsymbol{h}}$ terms. It gives upper and lower bounds for $LogR_{\mathrm{EL}}(\boldsymbol{w})$, indicated by $\sim$. The second bound, Eq. (16), uses the first bounding method for the $\log\sum_{\boldsymbol{y}}$ term, and bounds the $\log Z$ by using the first bounding method for $\boldsymbol{h}$ and the second bounding method for $\boldsymbol{y}$ (i.e., replace it by $\boldsymbol{y}_i$).

We see that the second bounds, Eq. (16), are identical to the latent structure SVM criterion provided we set $E(\boldsymbol{w}) = |\boldsymbol{w}|^2$. The first method, Eq. (15), provides a tighter upper bound, as well as a lower bound. Both bounds are non-convex, so the first bound may be preferred because it is tighter.

There are also parallels between the algorithms used to deal with hidden variables in probability models and those used by latent SVMs. For example, a standard algorithm for learning a latent SVM [50] proceeds by decomposing Eq. (16) into concave (the last term) and convex parts (everything else) and using a CCCP algorithm [55]. This yields an iterative two-step algorithm which estimates the parameters $\boldsymbol{w}$ keeping the hidden variables $\boldsymbol{h}$ fixed, corresponding to treating the $\boldsymbol{h}$ as observed variables and minimizing Eq. (5), and then estimating the $\boldsymbol{h}$ variables by maximum a posteriori treating the $\boldsymbol{w}$ are known. This iterative two-step algorithm is reminiscent of the EM algorithm [56] which instead would update the parameters $\boldsymbol{w}$ assuming a known distribution $q(\boldsymbol{h})$ for the hidden variables and then update $q(\boldsymbol{h})$ with $\boldsymbol{w}$ fixed. Hence, the main difference is that EM updates a probability distribution $q(\boldsymbol{h})$ while latent SVM updates the variable $\boldsymbol{h}$ directly. Note that would be exact in the large $|\boldsymbol{w}|$ limit.

6.2 Multiple instance learning (MIL)

In this section, we first show that the MI-SVM formulation of MIL can also be viewed as minimizing an upper bound of log-risk of a corresponding probabilistic model. This connection allows us to extend our framework to other types of MIL objective functions such as MI-SVM [51].

We can express the MI-SVM criterion as follows [51]:

$$\frac{1}{2}|\boldsymbol{w}|^2 + C\sum_{\boldsymbol{i}\in\mathcal{I}} \max\{0, 1 - y_i \max_{a\in\boldsymbol{i}} \boldsymbol{w}\cdot\boldsymbol{x}_{\boldsymbol{i}}^a\}, \quad (17)$$

where $\mathcal{I}$ is the set of 'bags', $\boldsymbol{i}\in\mathcal{I}$ is the bag index, and $a\in\boldsymbol{i}$ is an instance in the bag. If a bag is 'positive' then it has a single 'active' element, but it has no 'active' elements if it is 'negative'.

We derive the MI-SVM criterion in Eq. (17) by formulating it in terms of a probabilistic model with hidden variables and then adapting the arguments used for latent SVM in the last section. First, we introduce hidden variables $\{h_{\boldsymbol{i}}^a\}$ associated to the instances $\boldsymbol{x}_{\boldsymbol{i}}^a$ of all bags, such that $h_{\boldsymbol{i}}^a = 1$ if the instance is 'active' and $h_{\boldsymbol{i}}^a = 0$ if it is not 'active'. We have constraints that either: (i) $\sum_{a\in\boldsymbol{i}} h_{\boldsymbol{i}}^a = 1$ if the bag $\boldsymbol{i}$ is positive, and (ii) $\sum_{a\in\boldsymbol{i}} h_{\boldsymbol{i}}^a = 0$

if the bag is negative.

Next we define a joint probability distribution over the output y and hidden variables $h_{\boldsymbol{i}}$:

$$P(y, h_{\boldsymbol{i}}|\boldsymbol{x}_i) = \frac{1}{Z_i(\boldsymbol{w}, \boldsymbol{x}_i)}\delta(\frac{y}{2} + \frac{1}{2}, \sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a) \cdot \exp\{\sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a\}, \quad (18)$$

where $Z_i(\boldsymbol{w}, \boldsymbol{x}_i) = 1 + \sum_{a \in \boldsymbol{i}} \exp\{\boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a\}$. We first check that the MAP estimate from the probability model, Eq. (18), is equivalent to $\hat{y}(\boldsymbol{x}_i, \boldsymbol{w}) = \operatorname{sgn} \max_{a \in \boldsymbol{i}} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a$ which is the MI-SVM decision-rule. Denoting $(\hat{y}, \hat{h}_{\boldsymbol{i}}) = \arg\max \log P(y, h_{\boldsymbol{i}}|\boldsymbol{x}_i)$, we see that the equivalence holds since MAP compares $\log P(\hat{y} = 1, \hat{h}_{\boldsymbol{i}}|\boldsymbol{x}_i) = \max_{a \in \boldsymbol{i}} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a - \log Z_i(\boldsymbol{w}, \boldsymbol{x}_i)$ to $\log P(\hat{y}, \hat{h}_{\boldsymbol{i}}|\boldsymbol{x}_i) = -\log Z_i(\boldsymbol{w}, \boldsymbol{x}_i)$.

To see the equivalence of learning criteria, we apply the result for latent SVMs from the previous section, with loss function $\bar{l}(y, h, y_i) = 1$ if $y \neq y_i$, and $= 0$ otherwise. We set $\tilde{\boldsymbol{w}} = (\boldsymbol{w}, 1)$ and $\boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{i}}, y_i, h_{\boldsymbol{i}}) = (\sum_{a \in \boldsymbol{i}} h^a \boldsymbol{x}_{\boldsymbol{i}}^a, \log \delta(\frac{y}{2} + \frac{1}{2}, \sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a))$. It can be checked that: (i) $\max_h \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{i}}, y_i, h_{\boldsymbol{i}}) = \frac{y_{\boldsymbol{i}}+1}{2} \max_a \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a$, and (ii) $\max_{y,h}\{\bar{l}(y, h, y_i) + \tilde{\boldsymbol{w}} \cdot \boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{i}}, y_i, h_{\boldsymbol{i}})\} = \max\{\frac{y_{\boldsymbol{i}}+1}{2} \max_a \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a, 1 + \frac{1-y_{\boldsymbol{i}}}{2} \max_a \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a\}$. The result follows.

We can extend the above analysis to MI-SVM [51], which is a variant of MIL that allows multiple 'active' elements in a positive bag. We modify the probability distribution in Eq. (18) by relaxing the constraint that $\sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a = 1$, and replace it with $\sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a > 0$ if the bag $\boldsymbol{i}$ is positive. In other words, the joint distribution is defined as

$$P(y, h_{\boldsymbol{i}}|\boldsymbol{x}_i) = \frac{1}{Z_i(\boldsymbol{w}, \boldsymbol{x}_i)}\delta(\frac{y}{2} + \frac{1}{2}, H(\sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a)) \cdot \exp\{\sum_{a \in \boldsymbol{i}} h_{\boldsymbol{i}}^a \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a\}, \quad (19)$$

where $H(z) = 1$ if $z \geqslant 1$, $H(z) = 0$ otherwise, and $Z_i(\boldsymbol{w}, \boldsymbol{x}_i)$ is the partition function. Let $(\hat{y}, \hat{h}_{\boldsymbol{i}}) = \arg\max \log P(y, h_{\boldsymbol{i}}|\boldsymbol{x}_i)$. It follows that: (i) $\hat{y} = 0$ and $h_{\boldsymbol{i}}^a = 0$, $\forall a$ provided $\boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a \leqslant 0$, $\forall a$; and (ii) $\hat{y} = 1$ and $h_{\boldsymbol{i}}^a = 1$, $\forall a$ such that $\boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a > 0$. So MAP estimate selects $\hat{y} = 1$ provided $\sum_{a \in \boldsymbol{i}: \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a > 0} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a > 0$. This is equivalent to the MI-SVM decision criterion $\hat{y} = 1$ provided $\sum_{a \in \boldsymbol{i}} \frac{\operatorname{sgn}(\boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a)+1}{2} > 0$. We then derive the bound by using Eq. (19) and the same argument as for MI-SVM. This requires checking the following results: (i) $\max_h \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{i}}, y_{\boldsymbol{i}}, h_{\boldsymbol{i}}) = \sum_{a \in \boldsymbol{i}} \frac{y_{\boldsymbol{i}}^a+1}{2} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a$, and (ii) $\max_{y,h}\{\bar{l}(y, h, y_{\boldsymbol{i}}) + \tilde{\boldsymbol{w}} \cdot \boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{i}}, y_{\boldsymbol{i}}, h_{\boldsymbol{i}})\} = \max\{\sum_{a \in \boldsymbol{i}} \frac{y_{\boldsymbol{i}}^a+1}{2} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a, 1 + \sum_{a \in \boldsymbol{i}} \frac{-y_{\boldsymbol{i}}^a+1}{2} \boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a\}$, where $y_{\boldsymbol{i}}^a = \operatorname{sgn}(\boldsymbol{w} \cdot \boldsymbol{x}_{\boldsymbol{i}}^a)$ if $y_{\boldsymbol{i}} = 1$, and $y_{\boldsymbol{i}}^a = -1$ if $y_{\boldsymbol{i}} = -1$.

6.3 Ordinal regression with SVM (rank-SVM)

We can also apply our analysis to SVM based ordinal regression problems, or rank-SVMs (e.g., [57]). Let $\{(\boldsymbol{x}_i, y_i)_{i=1}^N\}$ be a training set, and y_i denotes the rank of data instance $\boldsymbol{x}_i$ and takes value from $\{1, 2, \ldots, R\}$. We want to learn a prediction function $h(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$, such that $y_i > y_j \iff h(\boldsymbol{x}_i) > h(\boldsymbol{x}_j)$. Let $\mathcal{P} = \{(i, j) : y_i > y_j\}$, and $m = |\mathcal{P}|$. The original rank-SVM training is formulated as the following optimization problem:

$$\min_{\boldsymbol{w}, \xi_{ij} \geqslant 0} \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{m} \sum_{(i,j) \in \mathcal{P}} \xi_{ij},$$
$$\text{s.t. } \forall (i, j) \in \mathcal{P} : \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i \geqslant \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_j + 1 - \xi_{ij}. \quad (20)$$

Let $z_{ij} = 1$ if $y_i > y_j$ and $z_{ij} = -1$ if $y_i < y_j$, we can see the rank-SVM uses the following learning cost function:

$$L(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{2m} \sum_{i,j} \max\{0, 1 - z_{ij}\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{x}_i - \boldsymbol{x}_j)\}, \quad (21)$$

which essentially an binary classification problem with $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ as input and z_{ij} as its binary label. The prediction function has a form of $H(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{w}^{\mathrm{T}}(\boldsymbol{x}_i - \boldsymbol{x}_j)$. As in Section 5, we can build a connection with the probabilistic model in Eq. (9) by defining $\boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{x}_j, z_{ij}) = \frac{1}{2}z_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)$ and using 0-1 loss.

7 Illustrations

Max-margin based learning criteria have been successfully applied to a range of problems including learning Markov models [47], learning grammars for natural language [48]. An important example in computer vision is the application of latent structure SVM to object detection [37]. In this section, we report several examples from computer vision for a range of tasks including image labeling, object detection and motion estimation, which yield close to or at the state-of-the-art on Microsoft Research Cambridge (MSRC) and PASCAL VOC databases [23–27].

This models are of the type described in Section 3 and later previous sections. The input to the models is the image intensity and so we set $\boldsymbol{x} = \boldsymbol{I}$. The graph states $\boldsymbol{y}$ can either be all observed variables or a combination of observed and hidden variables $\boldsymbol{y}$ and $\boldsymbol{h}$, depending on the application. A loss function is also specified depending on the application.

An inference algorithm is required to compute the MAP estimate of the probability. For the applications in which the graph structure is a tree, exact inference can be performed by dynamic programming or its variants in the $\boldsymbol{y}$ variables. In most cases the graphical models will have a large number of small closed loops, e.g., see Fig. 2(b), and so junction trees will converge to the global optimum. For other applications, approximate inference algorithms such as belief propagation or graph cuts can be used. The learning can be done by making the approximations described in the previous sections. For applications without hidden variables we can use the structure SVM criterion described in Section 5, while for applications with hidden units we use the latent SVM techniques described in Subsection 6.1.

Our first example is **image labeling**, where the task is to assign a label $s \in \mathcal{S}$ to every pixel in the image. For example, the MSRC dataset [58] has 21 different labels in $\mathcal{S}$. We consider a hierarchical model with a quadtree structure in Refs. [23,25], illustrated in Fig. 4(a). The state variable $\boldsymbol{y}_\mu$ of a node $\mu \in \mathcal{V}$ gives a description of the image region underneath it in terms of a partition into different subregions with a label assigned to each subregion. Formally, the state variables are written as $y_\mu = (c_\mu, \boldsymbol{s}_\mu)$, where $c_\mu \in \mathcal{C}$ is a *segmentation-template*, which partitions the image region into subregions, and $\boldsymbol{s}_\mu$ assigns labels to each subregion. The state variables at higher-levels of the graph give crude 'executive summaries' descriptions of image regions which are refined at lower-levels, see Fig. 4(c).

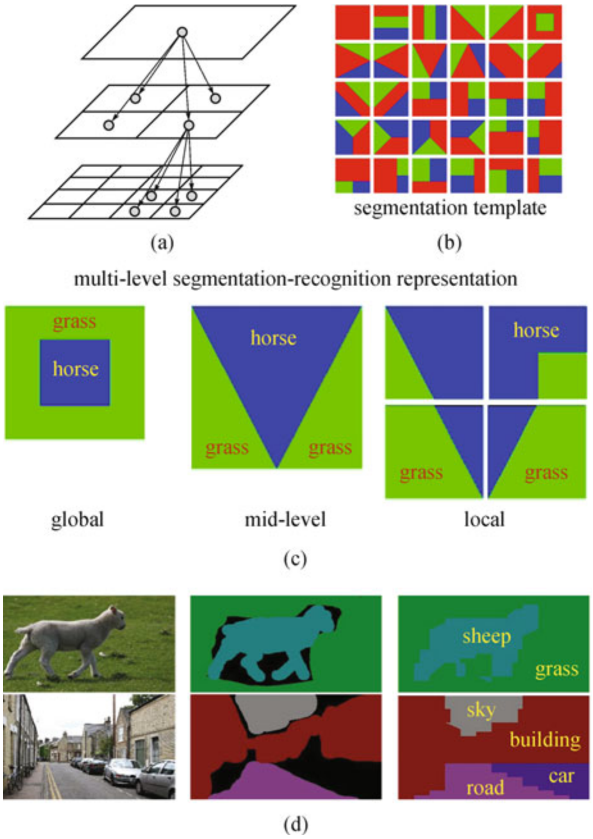


Fig. 4 (a) The graph structure of the recursive compositional model used for image labeling in Ref. [23]. The state variable at each graph node describes an image region by a labeled segmentation-template. (b) Thirty segmentation-templates. (c) The states at higher-levels of the graph give crude descriptions of image regions which are refined at lower-levels. (d) Examples on the MSRC dataset (image, groundtruth, model output).

Each node $\mu \in \mathcal{V}$ has a data potential $\boldsymbol{\phi}^D(y_\mu, \boldsymbol{I})$ which links its state y_μ to the input data $\boldsymbol{I}$. These potentials are based on filterbanks of image features. There are also clique potential terms $\boldsymbol{w}_\alpha \cdot \boldsymbol{\phi}_\alpha(\boldsymbol{y}_\alpha, \boldsymbol{I})$ which impose compatibility between state variables at different levels of the hierarchy and priors on the frequency of different segmentation-templates, the labels, and the spatial relations between labels (e.g., whether a cow is likely to be adjacent to a car or to the sky). These potentials are specified in advance and the learning task is to estimate their parameters $\boldsymbol{w}$, hence to select which potentials to use and assign weights to them.

The *inference* can be performed by pruned dynamic programming exploiting the tree-like structure of the graph. The state space of the variables is big so pruning is performed to remove states with low energy. The *learning* is performed using the training dataset $\mathcal{D} = \{(\boldsymbol{y}^\mu, \boldsymbol{x}^\mu) : \mu = 1, 2, \ldots, N\}$. The groundtruth is only specified at the pixels, but a simple automatic methods can be used to estimate the groundtruth for the state variables of the graph. In this work, the loss function penalizes all errors equally. The probabilistic criterion for learning the parameters $\boldsymbol{w}$ and its upper bound can be expressed in the form described in Section 5. For this application, the authors choose to solve the problem by minimizing the upper bound in the primal space using the structure perceptron algorithm [53]. The *performance* of this method was among the start-of-the-art on the MSRC dataset. The labeling performance was 74.4% for class-average accuracy and 81.4 % for pixel-average accuracy, using the same set up and criteria reported in Ref. [58], which were significant improvements (by 7.3% and 5.5%) over alternative methods at time of submission.

Object detection and pose estimation. Another example is to detect articulated objects which have multiple poses, such as people and horses, see Fig. 3 [24]. The object is represented by an AND/OR graph [5]. Essentially the object is constructed by composition which involves AND-ing and OR-ing elementary parts together to form the object. The OR-ing operation allows the model to deal with different poses of the object. This is formulated as probability defined on a graph with state variables $y_\mu = (p_\mu, t_\mu)$, where p_μ represents the pose (i.e., position, orientation, and scale) of an object part, and t_μ is a binary variable which indicates if the part is selected or not. The layers of the graph alternate between AND nodes, for which the part is a composition of the sub-parts of its child nodes (see Fig. 3), and OR nodes which requires a node to select one of its child nodes (i.e., a 'head' must be 'head-up' or 'head-down'). The t variables perform the selection for the OR nodes, similar to switch variables [5]. The graph structure has different topology due to the state of the t variables (i.e., the selections made at the OR nodes).

This means that a graph containing only 40 nodes can have 100 different topologies, which correspond to 100 different poses. An alternative strategy would be to have 100 different models — one for each pose of the object — and perform pose estimation by selection between these different models. The AND/OR graph is more compact and efficient, because it is able to *share parts* (i.e., the most elementary components) between different poses. The potentials $\boldsymbol{\phi}_\alpha(\boldsymbol{y}_\alpha)$ impose spatial relations on the relative positions of parts, their composition from more elementary parts, and the selection choices made at OR nodes. The data potentials $\boldsymbol{\phi}^D(y_\mu, \boldsymbol{I})$ relate the node at level $l = 1$ to the image (e.g., by encouraging the boundaries of parts to correspond to edge-type features in the image).

The graph structure for this model contains some closed loops because of the graph edges connecting siblings (e.g., the spatial representations between parts at the same scale). However, the number of closed loops is small enough that we can perform *inference* using dynamic programming with the junction trees algorithm. The size of the state space, however, is very large because there are many possible values for the pose p_μ of each part. Therefore, we perform pruning based on the energy and by surround suppression (penalizing states which are too similar to each other). We emphasize that we are performing inference over the state variables *including* the topology (i.e., the t_μ's), which means that we do inference over 100 different models efficiently be exploiting the re-use of the elementary parts.

The loss function $l(\boldsymbol{y}, \boldsymbol{y}_{\mathrm{T}})$ penalizes configurations where the estimated pose p_μ of a node $\mu \in \mathcal{V}$ differs from an image $\boldsymbol{I}_i$ differs from the correct (ground-truth) pose $p_{mu,i}$ by more than a threshold. The groundtruth for the baseball database is specified only for the nodes at level $l = 1$, but we can estimate it for the higher level nodes also. *Learning* can be approximated by the SVM criterion without hidden variables, as described in Section 5. For this application, we choose to perform learning in the dual space using structured max-margin algorithms [47–49].

Our next example is an extension of the latent SVM model successfully applied to **object detection and localization** [26]. Each object model is a mixture of six different hierarchical graphs, shown in Fig. 5(a). The nodes have variables p_μ which indicate the positions of the parts, and there is a variable V which specifies which of the six graphs is selected (corresponding to different views/poses). There is also a variable $\overline{y}$ which specifies if the object is detected. In the object model, there are potentials $\boldsymbol{w}_\alpha \cdot \boldsymbol{\phi}_\alpha(\boldsymbol{y}_\alpha)$ which impose spatial relations on the relative positions of the object parts. There are also data potentials $\boldsymbol{\phi}^D(y_\mu, \boldsymbol{I})$, which relate the position of each part to the image $\boldsymbol{I}$, based on HOG and SIFT features.

The *inference* requires estimating the best state of the six possible hierarchies which can be done by dynamic programming over each hierarchy, followed by exhaustive search over the six possibilities. The *learning* involves hidden variables since the training data is of form $\mathcal{D} = \{(\boldsymbol{I}_i, \overline{y}_i) : i = 1, 2, \ldots, N\}$, and only says whether an object is present or not in the image (i.e., it does not specify the mixture component V or the positions $\{p_\mu\}$ of the object parts). With a '0-1' loss function, the learning criterion can be expressed in the form given in Subsection 6.1. Again, we can view the learning objective as an upper bound, which was optimized by a variant of Yu and Joachims's procedure [50]. The performance of this approach achieves the state-of-the-art and has recently obtained second prize in the PASCAL object detection challenge in 2010.

The third task is **motion estimation**, which aims at estimating the motion flow between two images. The model used as our example [27] is defined on a hierarchical graph with closed loops (see Fig. 5(c)). The nodes of the graph represent the motion at different scales, so it defines motion priors and contextual effects at a range of scales. The potential functions of this model contain data terms which require the corresponding points in the two images have similar intensity properties and a hierarchical prior which encourages smoothness and slowness of the velocity field. The overall distribution of motion given image pair has the same form as Eq. (1).

The *inference* is done by a variant of loops belief propagation with adaptive quantization of velocity space. For *learning* algorithm, the same loss function as in **image labeling** is employed (after quantization), and the upper bound is minimized by structure perceptron based on the groundtruth from the Middlesbury dataset [59]. This model is shown to be able to handle both short-range and long-range motion, and adequately account for human performance in psychophysical experiments [27].

8 Summary

The purpose of this paper is to draw relationships between probabilistic methods and machine learning approaches which use max-margin criteria. We show that several standard max-margin criteria, with and without hidden/latent variables, can be obtained as upper or lower bounds to the probabilistic criteria. This perspective suggests that machine learning approaches can be applied to problems that can be formulated as probabilistic inference over structured representations (e.g., graphs and grammars), and hence can be applied to a range of computer vision problems as approximations. These relationships also suggest alternative machine learning criteria which may be more effective in

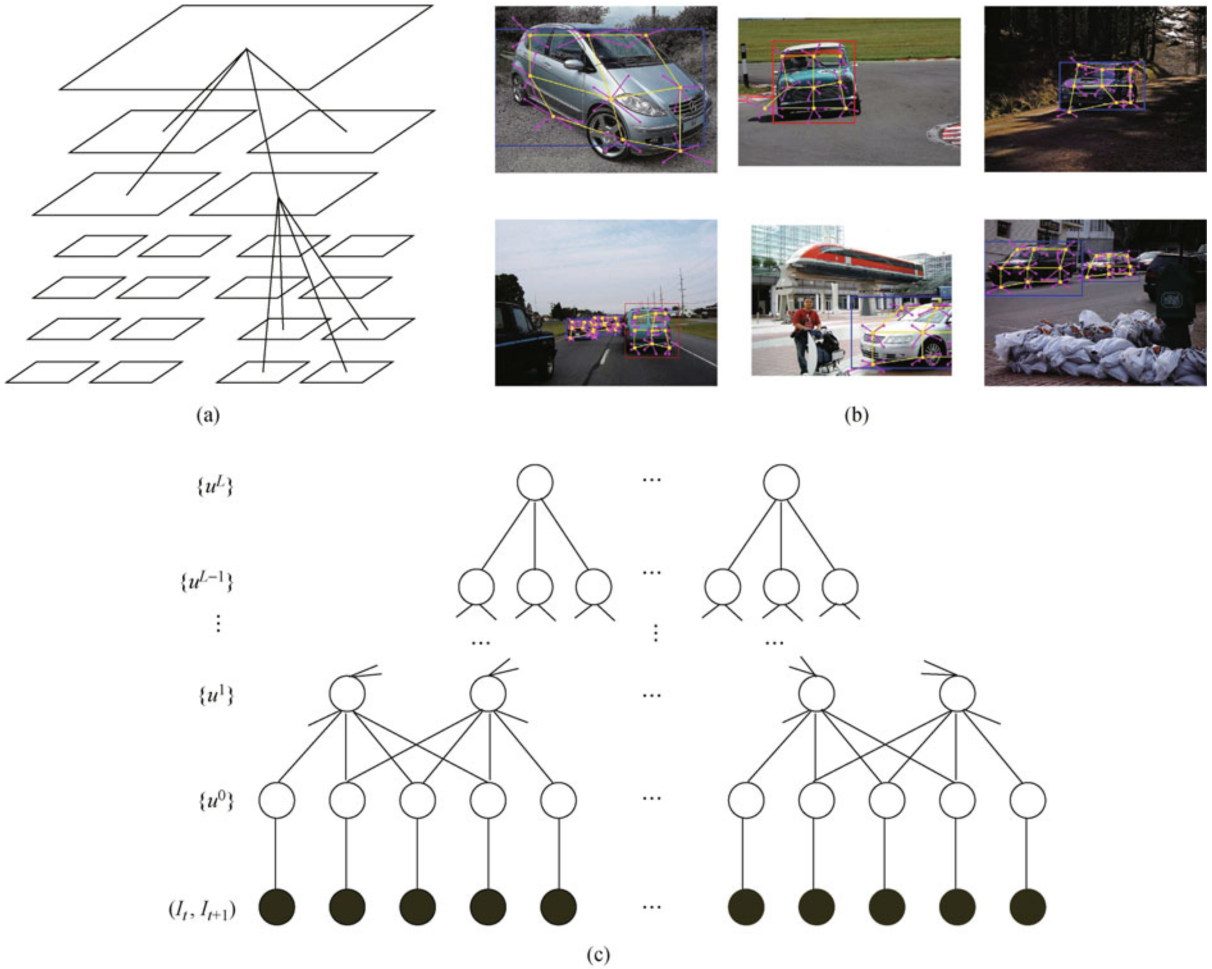


Fig. 5 (a) A hierarchical model of parts, which are used as mixture components to model different poses of objects [26]. (b) These models can detect objects from different viewpoints, with changes in geometry (spatial warps) and even partial occlusion. They are trained by latent SVM techniques where the groundtruth only specifies that an object is present or not. (c) A hierarchical model of motion in one dimension [27]. Each node represents motion at different locations and scales. A child node can have multiple parents, and the prior constraints on motion are expressed by parent-child interactions.

some cases than the standard ones (e.g., by using tighter upper bounds).

What are the trade-offs, from a probabilistic perspective, to using these types of bounds? There are clearly cases where making these bounds could cause serious problems, e.g., replacing the sum over hidden variables in the EM algorithm by their maximum value. However, on the other hand, it is possible that the bounds (i.e., the max-margin criterion) may be more robust than the probabilistic methods. The argument, original due to Vapnik [12], is that learning algorithms should pay most attention to regions near the decision boundaries rather than wasting resources trying to fit the parameters of the models away from the boundaries. In particular, if our probability model for the data is wrong, or non-robust, then attempting to fit the data perfectly by the probability models may cause errors as well as be computationally demanding. We note, however, that these bounds become exact in the large $|\boldsymbol{w}|$ limit, where the margin tends to zero, which is the situations where poor generalization is expected.

Acknowledgements We would like to acknowledge support for this project from the National Science Foundation (NSF grant IIS-9988642), the Multidisciplinary Research Program of the Department of Defense (MURI N00014-00-1-0637), and by the Korean Ministry of Education, Science, and Technology, under the National Research Foundation WCU program R31-10008.

References

1. Pearl J. Probabilistic Reasoning in Intelligent Systems. Morgan-Kaufmann, 1988
2. Manning C D, Schütze H. Foundations of Statistical Natural Language Processing. Cambridge, MA: MIT Press, 1999
3. Heckerman D. A tutorial on learning with Bayesian networks. Learning in Graphical Model. 1999, 301–354
4. Russell S, Norvig P. Artificial Intelligence: A Modern Approach. 2nd ed. Prentice Hall, 2003
5. Zhu S C, Mumford D. A stochastic grammar of images. Foundations and Trends in Computer Graphics and Vision, 2006, 2(4): 259–362
6. Tenenbaum J B, Griffiths T L, Kemp C. Theory-based Bayesian models of inductive learning and reasoning. Trends

in Cognitive Sciences, 2006, 10(7): 309–318

7. Smith N A. Linguistic Structure Prediction. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, 2011
8. Grenander U. Pattern Synthesis: Lectures in Pattern Theory 1. New York, NY: Springer, 1976
9. Grenander U. Pattern Analysis: Lectures in Pattern Theory 2. New York, NY: Springer, 1978
10. Tenenbaum J B, Yuille A L. IPAM Summer School: The Mathematics of the Mind. IPAM, UCLA, 2007
11. Jin Y, Geman S. Context and hierarchy in a probabilistic image model. In: Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition. 2006, 2145–2152
12. Vapnik V N. The Nature of Statistical Learning Theory. New York, NY: Springer-Verlag, 1995
13. Schölkopf B, Smola A. Leaning with Kernels. MIT Press, 2001
14. Franc V, Zien A, Schölkopf B. Support vector machines as probabilistic models. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). 2011, 665–672
15. Hastie T, Tibshirani R, Feldman J. The Elements of Statistical Learning. 1st ed. Springer, 2001
16. Lebanon G, Lafferty J. Boosting and maximum likelihood for exponential models. In: Proceedings of Neural Information Processing Systems (NIPS 2001), 2001
17. Rätsch G, Onoda T, Müller K R. Soft margins for AdaBoost. Machine Learning, 2001, 42(3): 287–320
18. Rätsch G, Mika S, Schölkopf B, Müller K R. Constructing boosting algorithms from SVMs: An application to one-class classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(9): 1184–1199
19. Wahba G. Spline Models for Observational Data. SIAM CBMS-NSF Regional Conference Series in Applied Mathematics. Volume 59. SIAM, Philadelphia, PA, 1990
20. Seeger M. Bayesian model selection for support vector machines, Gaussian processes and other kernel classifiers. In: Proceedings of Neural Information Processing Systems (NIPS 1999). 1999, 603–609
21. Poggio T, Smale S. The mathematics of learning: Dealing with data. American Mathematical Society, 2003
22. Zhu J, Xing E P. Maximum entropy discrimination Markov networks. Journal of Machine Learning Research, 2009, 10: 2531–2569
23. Zhu L, Chen Y, Lin Y, Yuille A L. A hierarchical image model for polynomial-time 2D parsing. In: Proceedings of Neural Information Processing Systems (NIPS 2008). 2008
24. Zhu L, Chen Y, Lu Y, Lin C, Yuille A L. Max margin AND/OR graph learning for parsing the human body. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. 2008
25. Zhu L, Chen Y, Yuille A L. Learning a hierarchical deformable template for rapid deformable object parsing. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32(6): 1029–1043
26. Zhu L, Chen Y, Yuille A L, Freeman W. Latent hierarchical structure learning for object detection. In: Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition. 2010
27. Wu S, He X, Lu H, Yuille A L. A unified model of short-range and long-range motion perception. In: Proceedings of Neural Information Processing Systems (NIPS 2010). 2010
28. Fischler M A, Elschlager R A. The representation and matching of pictorial structures. IEEE Transactions on Computers, 1973, 22(1): 67–92
29. Yuille A L, Hallinan P W, Cohen D S. Feature extraction from faces using deformable templates. International Journal of Computer Vision, 1992, 8(2): 99–111
30. Cootes T F, Edwards G J, Taylor C J. Active appearance models. Lecture Notes in Computer Science, 1998, 1407: 484–498
31. Tu Z, Chen X, Yuille A L, Zhu S C. Image parsing: Unifying segmentation, detection, and recognition. In: Proceedings of the 9th IEEE International Conference on Computer Vision. 2003, 1: 18–25
32. Lafferty J D, McCallum A, Pereira F C N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning. 2001, 282–289
33. Kumar S, Hebert M. A hierarchical field framework for unified context-based classification. In: Proceedings of the 10th IEEE International Conference on Computer Vision. 2005, 2: 1284–1291
34. He X, Zemel R S, Carreira-Perpiñán M Á. Multiscale conditional random fields for image labeling. In: Proceedings of the 2004 IEEE Conference on Computer Vision and Pattern Recognition. 2004, 2: 695–702
35. Winn J M, Jojic N. LOCUS: Learning object classes with unsupervised segmentation. In: Proceedings of the 10th IEEE International Conference on Computer Vision. 2005, 1: 756–763
36. Sudderth E B, Torralba A B, Freeman W T, Willsky A S. Learning hierarchical models of scenes, objects, and parts. In: Proceedings of the 10th IEEE International Conference on Computer Vision. 2005, 2: 1331–1338
37. Felzenszwalb P, Mcallester D, Ramanan D. A discriminatively trained, multiscale, deformable part model. In: Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition. 2008
38. Shilman M, Liang P, Viola P A. Learning non-generative grammatical models for document analysis. In: Proceedings of the 10th IEEE International Conference on Computer Vision. 2005, 2: 962–969
39. Ahuja N, Todorovic S. Learning the taxonomy and models of categories present in arbitrary image. In: Proceedings of the 11th IEEE International Conference on Computer Vision. 2007
40. Todorovic S, Ahuja N. Region-based hierarchical image matching. International Journal of Computer Vision, 2008, 78(1): 47–66
41. Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets. Neural Computation, 2006, 18(7): 1527–1554
42. Amit Y, Geman D, Fan X. A coarse-to-fine strategy for multiclass shape detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(12): 1606–1621

43. Kokkinos I, Yuille A L. Hop: Hierarchical object parsing. In: Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition. 2009, 802–809

44. Kokkinos I, Yuille A L. Inference and learning with hierarchical shape models. International Journal of Computer Vision, 2011, 93(2): 201–225

45. Tipping M E. Sparse Bayesian learning and the relevance vector machine. Journal of Machine Learning Research, 2001, 1: 211–244

46. Pletscher P, Ong C S, Buhmann J M. Entropy and margin maximization for structured output learning. In: Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases. 2010

47. Taskar B, Guestrin C, Koller D. Max-margin Markov networks. In: Proceedings of Neural Information Processing Systems (NIPS 2003). 2003

48. Taskar B, Klein D, Collins M, Koller D, Manning C. Max-margin parsing. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing. 2004

49. Tsochantaridis I, Joachims T, Hofmann T, Altun Y. Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research, 2006, 6: 1453–1484

50. Yu C N, Joachims T. Learning structural SVMs with latent variables. In: Proceedings of the 26th International Conference on Machine Learning. 2009

51. Andrews S, Tsochantaridis I, Hofmann T. Support vector machines for multiple-instance learning. In: Proceedings of Neural Information Processing Systems (NIPS 2002). 2002

52. Xu L, Wilkinson D, Schuurmans D. Discriminative unsupervised learning of structured predictors. In: Proceedings of the 23rd International Conference on Machine Learning. 2006

53. Collins M. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In: Proceedings of the ACL02 Conference on Empirical Methods in Natural Language Processing. 2002

54. Yedidia J S, Freeman W T, Weiss Y. Constructing free-energy approximations and generalized belief propagation algorithms. IEEE Transactions on Information Theory, 2005, 51(7): 2282–2312

55. Yuille A L, Rangarajan A. The concave-convex procedure (CCCP). In: Proceedings of Neural Information Processing Systems (NIPS 2001). 2001

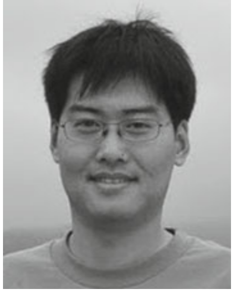
56. Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B: Methodological, 1977, 39(1): 1–38

57. Joachims T. Training linear SVMs in linear time. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2006

58. Shotton J, Winn J, Rother C, Criminisi A. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: Proceedings of European Conference on Computer Vision. 2006

59. Baker S, Roth S, Scharstein D, Black M J, Lewis J P, Szeliski R. A database and evaluation methodology for optical flow. In: Proceedings of the 11th IEEE International Conference on Computer Vision. 2007

Alan YUILLE received his B.A. degree in mathematics from the University of Cambridge in 1976, and completed his Ph.D. degree in theoretical physics at Cambridge in 1980 studying under Stephen Hawking. Following this, he held a postdoc position with the Physics Department, University of Texas at Austin, and the Institute for Theoretical Physics, Santa Barbara. He then joined the Artificial Intelligence Laboratory at MIT (1982–1986), and followed this with a faculty position in the Division of Applied Sciences at Harvard (1986–1995), rising to the position of associate professor. From 1995–2002 Alan worked as a senior scientist at the Smith-Kettlewell Eye Research Institute in San Francisco. In 2002 he accepted a position as full professor in the Department of Statistics at the University of California at Los Angeles. He has over one hundred and fifty peer-reviewed publications in vision, neural networks, and physics, and has co-authored two books: *Data Fusion for Sensory Information Processing Systems* (with J. J. Clark) and *Two- and Three-Dimensional Patterns of the Face* (with P. W. Hallinan, G. G. Gordon, P. J. Giblin and D. B. Mumford); he also co-edited the book *Active Vision* (with A. Blake). He has won several academic prizes and is a Fellow of IEEE.



Xuming HE received the B.Sc. and M.Sc. degrees in electronics engineering from Shanghai Jiao Tong University in 1998 and 2001, and Ph.D. degree in computer science from the University of Toronto in 2008. He held postdoctoral position at the University of California at Los Angeles (UCLA) before joining National ICT Australia (NICTA). He is currently a researcher in the Canberra Lab of NICTA and also an adjunct Research Fellow of Engineering Department at the Australian National University (ANU). His research interests include image segmentation and labeling, visual motion analysis, vision-based navigation, and undirected graphical models.