Hidden Part Models for Human Action Recognition: Probabilistic vs. Max-Margin

Yang Wang and Greg Mori, *Member, IEEE*

Abstract—We present a discriminative part-based approach for human action recognition from video sequences using motion features. Our model is based on the recently proposed hidden conditional random field (HCRF) for object recognition. Similar to HCRF for object recognition, we model a human action by a flexible constellation of parts conditioned on image observations. Different from object recognition, our model combines both large-scale global features and local patch features to distinguish various actions. Our experimental results show that our model is comparable to other state-of-the-art approaches in action recognition. In particular, our experimental results demonstrate that combining large-scale global features and local patch features performs significantly better than directly applying HCRF on local patches alone. We also propose an alternative for learning the parameters of an HCRF model in a max-margin framework. We call this method the max-margin hidden conditional random field (MMHCRF). We demonstrate that MMHCRF outperforms HCRF in human action recognition. In addition, MMHCRF can handle a much broader range of complex hidden structures arising in various problems in computer vision.

Index Terms—Human action recognition, part-based model, discriminative learning, max-margin, hidden conditional random field

1 Introduction

A good image representation is the key to the solution of many recognition problems in computer vision. In the literature, there has been a lot of work on designing good image representations (i.e. features). Different image features typically capture different aspects of image statistics. Some features (e.g. GIST [1]) capture the global scene of a image, while others (e.g. SIFT [2]) capture the image statistics of local patches.

A primary goal of this work is to address the following question: is there a principled way to combine both large scale and local patch features? This is an important question in many visual recognition problems. The work in [3] has shown the benefit of combining large scale features and local patch features for object detection. In this paper, we apply the same intuition to the problem of recognizing human actions from video sequences. In particular, we represent a human action by combining large-scale template features and part-based local features in a principled way, see Fig. 1.

Recognizing human actions from videos is a task of obvious scientific and practical importance. In this paper, we consider the problem of recognizing human actions from video sequences on a frame-by-frame basis. We develop a discriminatively trained hidden part model to represent human actions. Our model is inspired by the hidden conditional random field (HCRF) model [4] in object recognition.

- *Y. Wang is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. E-mail: yangwang@uiuc.edu*
- *G. Mori is with the School of Computing Science, Simon Fraser University, Burnaby, BC, V5A 1S6, Canada. E-mail: mori@cs.sfu.ca*

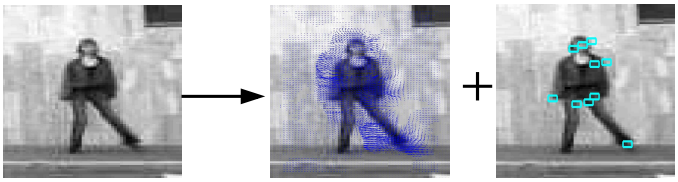


Fig. 1. High-level overview of our model: a human action is represented by the combination of large-scale template feature (we use optical flow in this work) and a collection of local "parts". Our goal is to unify both features in a principled way for action recognition.

In object recognition, there are three major representations: global template (rigid, e.g. [5], or deformable, e.g. [6]), bag-of-words [7], and part-based [3], [8]. All three representations have been shown to be effective on certain object recognition tasks. In particular, recent work [3] has shown that part-based models outperform global templates and bag-of-words on challenging object recognition tasks.

A lot of the ideas used in object recognition can also be found in action recognition. For example, there is work [9] that treats actions as space-time shapes and reduces the problem of action recognition to 3D object recognition. In action recognition, both global template [10] and bag-of-words models [11]–[14] have been shown to be effective on certain tasks. Although conceptually appealing and promising, the merit of part-based models has not yet been widely recognized in action recognition. One goal of this work is to address this issue, and to show the benefits of combining large-scale global template features and part models based on local patches.

A major contribution of this work is that we combine the flexibility of part-based approaches with the global perspectives of large-scale template features in a discriminative model for action recognition. We show that the combination of part-based and large-scale template features improves the final results.

Another contribution of this paper is to introduce a new learning method for training HCRF models based on the max-margin criterion. The new learning method, which we call the *Max-Margin Hidden Conditional Random Field(MMHCRF)*, is based on the idea of latent SVM (LSVM) in [3]. The advantage of MMHCRF is that the model can be solved efficiently for a large variety of complex hidden structures. The difference between our approach and LSVM is that we directly deal with multi-class classification, while LSVM in [3] only deals with binary classification. It turns out the multi-class case cannot be easily solved using the off-the-shelf SVM solver, unlike LSVM. Instead, we develop our own optimization technique for solving our problem. Furthermore, we directly compare probabilistic vs. max-margin learning criteria on identical models. We provide both experimental and theoretical evidences for the effectiveness of the max-margin learning.

Previous versions of this paper are published in [15] and [16]. The rest of this paper is organized as follows. Section 2 reviews previous work. Section 3 introduces our hidden part model for human action recognition, based on the hidden conditional random field. Section 3 also gives the details of learning and inference in HCRF. Section 4 presents a new approach called the *Max Margin Hidden Conditional Random Field (MMHCRF)* for learning the parameters in the hidden part model proposed in Section 3. In Section 5, we give a detailed analysis and comparison of HCRF and MMHCRF from a theoretical point of view. We present our experimental results on two benchmark datasets in Sec. 6 and conclude in Sec. 7.

2 Related Work

A lot of work has been done in recognizing actions from video sequences. Much of this work is focused on analyzing patterns of motion. For example, Cutler & Davis [17], and Polana & Nelson [18] detect and classify periodic motions. Little & Boyd [19] analyze the periodic structure of optical flow patterns for gait recognition. Rao et al. [20] describe a view-invariant representation for 2D trajectories of tracked skin blobs. There is also work using both motion and shape cues. For example, Bobick & Davis [21] use a representation known as "temporal templates" to capture both motion and shape, represented as evolving silhouettes. Nowozin et al. [13] find discriminative subsequences in videos for action recognition. Jhuang et al. [22] develop a biologically inspired system for action recognition.

Recently space-time interest points [23] were introduced into the action recognition community [14]. Most of the approaches use bag-of-words representation to model the space-time interest points [11], [12], [14]. This representation suffers the same drawbacks as their 2D analogies in object recognition, i.e., spatial information is ignored. There is also some recent work [24] that tries to combine space-time interest points with a generative model similar to the constellation model [25] in object recognition.

Our work is partly inspired by a recent work in part-based event detection [26]. In that work, template matching is combined with a pictorial structure model to detect and localize actions in crowded videos. One limitation of that work is that one has to manually specify the parts. Unlike Ke et al. [26], the parts in our model are initialized automatically.

Our work is also related to discriminative learning methods in machine learning, in particular, learning with structured outputs [27]–[30] and latent variables [3], [4].

3 Hidden Conditional Random Fields for Human Actions

Our part-based representation for human actions is inspired by the hidden conditional random field model [4], which was originally proposed for object recognition and has also been applied in sequence labeling. Objects are modeled as flexible constellations of parts conditioned on the appearances of local patches found by interest point operators. The probability of the assignment of parts to local features is modeled by a conditional random field (CRF) [27]. The advantage of the HCRF is that it relaxes the conditional independence assumption commonly used in the bag-of-words approaches of object recognition. In standard bag-of-words approaches, all the local patches in an image are independently of each other given the class label. This assumption is somewhat restrictive. The HCRF relaxes this assumption by allowing nearby patches to interact with each other.

Similarly, local patches can also be used to distinguish actions. Figure. 6(a) shows some examples of human motion and the local patches that can be used to distinguish them. A bag-of-words representation can be used to model these local patches for action recognition. However, it suffers from the same restriction of conditional independence assumption that ignores the spatial structures of the parts. In this work, we use similar ideas to model the constellation of these local patches in order to alleviate this restriction.

There are also some important differences between objects and actions. For objects, local patches could carry enough information for recognition. But for actions, we believe local patches are not sufficiently informative. In our approach, we modify the HCRF model to combine local patches and large-scale global features. The large-scale global features are represented by a root model that takes the frame as a whole. Another important difference with [4] is that we use the learned root model to find discriminative local patches, rather than using a generic interest-point operator.

3.1 Motion Features

Our model is built upon the optical flow features in [10]. This motion descriptor has been shown to perform reliably with noisy image sequences, and has been applied in various tasks, such as action classification, motion synthesis, etc.

To calculate the motion descriptor, we first need to track and stabilize the persons in a video sequence. Any reasonable tracking or human detection algorithm can be used, since the motion descriptor we use is very robust to jitters introduced by the tracking. Given a stabilized video sequence in which the person of interest appears in the center of the field of view, we compute the optical flow at each frame using the Lucas-Kanade [31] algorithm. The optical flow vector field F is then split into two scalar fields F_x and F_y, corresponding to the x and y components of F. F_x and F_y are further half-wave rectified into four non-negative channels F_x^+, F_x^-, F_y^+, F_y^-, so that $F_x = F_x^+ - F_x^-$ and $F_y = F_y^+ - F_y^-$. These four non-negative channels are then blurred with a Gaussian kernel and normalized to obtain the final four channels Fb_x^+,Fb_x^-,Fb_y^+,Fb_y^- (see Fig. 2).

3.2 Hidden Part Model

Now we describe how we model a frame I in a video sequence. Let $\mathbf{x}$ be the motion feature of this frame, and y be the corresponding class label of this frame, ranging over a finite label alphabet $\mathcal{Y}$. Our task is to learn a mapping from $\mathbf{x}$ to y. We assume each image I contains a set of salient patches $\{I_1, I_2, ..., I_m\}$. we will describe how to find these salient patches in Sec. 3.3. Our training set consists of labeled images $\langle \mathbf{x}^{(t)}, y^{(t)} \rangle$ (as a notation convention, we use superscripts in brackets to index training images and subscripts to index patches) for $t = 1, 2, ..., N$, where $y^{(t)} \in \mathcal{Y}$ and $\mathbf{x}^{(t)} = (x_1^{(t)}, x_2^{(t)} ..., x_m^{(t)})$. $x_i^{(t)} = \mathbf{x}^{(t)}(I_i^{(t)})$ is the feature vector extracted from the global motion feature $\mathbf{x}^{(t)}$ at the location of the patch $I_i^{(t)}$. For each image $I = \{I_1, I_2, ..., I_m\}$, we assume there exists a vector of hidden "part" variables $\mathbf{h} = \{h_1, h_2, ..., h_m\}$, where each h_i takes values from a finite set $\mathcal{H}$ of possible parts. Intuitively, each h_i assigns a part label to the patch I_i, where $i = 1, 2, ..., m$. For example, for the action "waving-two-hands", these parts may be used to characterize the movement patterns of the left and right arms. The values of $\mathbf{h}$ are not observed in the training set, and will become the hidden variables of the model.

We assume there are certain constraints between some pairs of (h_j, h_k). For example, in the case of "waving-two-hands", two patches h_j and h_k at the left hand might have the constraint that they tend to have the same part label, since both of them are characterized by the movement of the left hand. If we consider $h_i (i = 1, 2, ..., m)$ to be vertices in a graph $\mathcal{G} = (\mathcal{E}, \mathcal{V})$, the constraint between h_j and h_k is denoted by an edge $(j, k) \in \mathcal{E}$. See Fig. 3 for an illustration of our model. Note that the graph structure can be different for different images. We will describe how to find the graph structure $\mathcal{E}$ in Sec. 3.3.

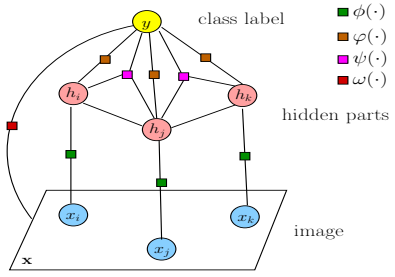


Fig. 3. Illustration of the model. Each circle corresponds to a variable, and each square corresponds to a factor in the model.

Given the motion feature $\mathbf{x}$ of an image I, its corresponding class label y, and part labels $\mathbf{h}$, a hidden conditional random field is defined as

$$p(y, \mathbf{h}|\mathbf{x}; \theta) = \frac{\exp(\theta^\top \cdot \Phi(\mathbf{x}, \mathbf{h}, y))}{\sum_{\hat{y} \in \mathcal{Y}} \sum_{\hat{\mathbf{h}} \in \mathcal{H}^m} \exp(\theta^\top \cdot \Phi(\mathbf{x}, \hat{\mathbf{h}}, \hat{y}))}$$

where θ is the model parameter, and $\Phi(y, \mathbf{h}, \mathbf{x})$ is a feature vector depending on the motion feature $\mathbf{x}$, the class label y, and the part labels $\mathbf{h}$. We use $\mathcal{H}^m$ to denote the set of all the possible labelings of m hidden parts. It follows that

$$\begin{aligned} p(y|\mathbf{x}; \theta) &= \sum_{\mathbf{h} \in \mathcal{H}^m} p(y, \mathbf{h}|\mathbf{x}; \theta) \\ &= \frac{\sum_{\mathbf{h} \in \mathcal{H}^m} \exp(\theta^\top \cdot \Phi(\mathbf{x}, \mathbf{h}, y))}{\sum_{\hat{y} \in \mathcal{Y}} \sum_{\mathbf{h} \in \mathcal{H}^m} \exp(\theta^\top \cdot \Phi(\mathbf{x}, \mathbf{h}, \hat{y}))} \end{aligned}$$

We assume $\theta^\top \cdot \Phi(y, \mathbf{h}, \mathbf{x})$ has the following form:

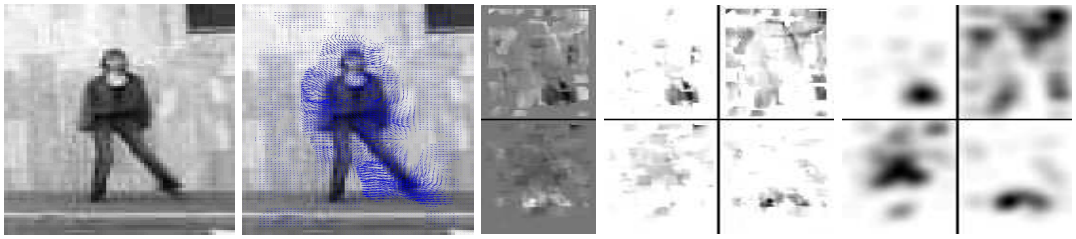
$$\begin{aligned} \theta^\top \cdot \Phi(\mathbf{h}, \mathbf{x}, y) &= \sum_{j \in \mathcal{V}} \alpha^\top \cdot \phi(x_j, h_j) + \sum_{j \in \mathcal{V}} \beta^\top \cdot \varphi(y, h_j) \\ &+ \sum_{(j,k) \in \mathcal{E}} \gamma^\top \cdot \psi(y, h_j, h_k) + \eta^\top \cdot \omega(y, \mathbf{x}) \end{aligned} \quad (1)$$

where $\phi(\cdot)$ and $\varphi(\cdot)$ are feature vectors depending on unary h_j's, $\psi(\cdot)$ is a feature vector depending on pairs of (h_j, h_k), $\omega(\cdot)$ is a feature vector that does not depend on the values of hidden variables. The details of these feature vectors are described in the following.

Unary potential $\alpha^\top \cdot \phi(x_j, h_j)$: This potential function models the compatibility between x_j and the part label h_j, i.e., how likely the patch x_j is labeled as part h_j. It is parametrized as

$$\alpha^\top \cdot \phi(x_j, h_j) = \sum_{c \in \mathcal{H}} \alpha_c^\top \cdot \mathbb{1}_{\{h_j = c\}} \cdot [f^a(x_j)\ f^s(x_j)] \quad (2)$$

where we use $[f^a(x_j)\ f^s(x_j)]$ to denote the concatenation of two vectors $f^a(x_j)$ and $f^s(x_j)$. $f^a(x_j)$ is a feature vector describing the appearance of the patch x_j. In our case, $f^a(x_j)$ is simply the concatenation of four channels of the motion features at patch x_j, i.e., $f^a(x_j) = [Fb_x^+(x_j)\ Fb_x^-(x_j)\ Fb_y^+(x_j)\ Fb_y^-(x_j)]$. $f^s(x_j)$ is



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

Fig. 2. Construction of the motion descriptor. (a) original image; (b) optical flow; (c) x and y components of optical flow vectors F_x, F_y; (d) half-wave rectification of x and y components to obtain 4 separate channels $F_x^+, F_x^-, F_y^+, F_y^-$; (e) final blurry motion descriptors $Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-$.

a feature vector describing the spatial location of the patch x_j. We discretize the whole image locations into l bins, and $f^s(x_j)$ is a length l vector of all zeros with a single one for the bin occupied by x_j. The parameter α_c can be interpreted as the measurement of compatibility between feature vector $[f^a(x_j)\ f^s(x_j)]$ and the part label $h_j = c$. The parameter α is simply the concatenation of α_c for all $c \in \mathcal{H}$.

Unary potential $\beta^\top \cdot \varphi(y, h_j)$: This potential function models the compatibility between class label y and part label h_j, i.e., how likely an image with class label y contains a patch with part label h_j. It is parametrized as

$$\beta^\top \cdot \varphi(y, h_j) = \sum_{a \in \mathcal{Y}} \sum_{b \in \mathcal{H}} \beta_{a,b} \cdot \mathbb{1}_{\{y=a\}} \cdot \mathbb{1}_{\{h_j=b\}} \quad (3)$$

where $\beta_{a,b}$ indicates the compatibility between $y = a$ and $h_j = b$.

Pairwise potential $\gamma^\top \cdot \psi(y, h_j, h_k)$**:** This pairwise potential function models the compatibility between class label y and a pair of part labels (h_j, h_k), i.e., how likely an image with class label y contains a pair of patches with part labels h_j and h_k, where $(j, k) \in \mathcal{E}$ corresponds to an edge in the graph. It is parametrized as

$$\gamma^\top \cdot \psi(y, h_j, h_k) = \sum_{a \in \mathcal{Y}} \sum_{b \in \mathcal{H}} \sum_{c \in \mathcal{H}} \gamma_{a,b,c} \cdot \mathbb{1}_{\{y=a\}} \cdot \mathbb{1}_{\{h_j=b\}} \cdot \mathbb{1}_{\{h_k=c\}} \quad (4)$$

where $\gamma_{a,b,c}$ indicates the compatibility of $y = a$, $h_j = b$ and $h_k = c$ for the edge $(j, k) \in \mathcal{E}$.

Root model $\eta^\top \cdot \omega(y, \mathbf{x})$**:** The root model is a potential function that models the compatibility of class label y and the large-scale global feature of the whole image. It is parametrized as

$$\eta^\top \cdot \omega(y, \mathbf{x}) = \sum_{a \in \mathcal{Y}} \eta_a^\top \cdot \mathbb{1}_{\{y=a\}} \cdot g(\mathbf{x}) \quad (5)$$

where $g(\mathbf{x})$ is a feature vector describing the appearance of the whole image. In our case, $g(\mathbf{x})$ is the concatenation of all the four channels of the motion features in the image, i.e., $g(\mathbf{x}) = [Fb_x^+\ Fb_x^-\ Fb_y^+\ Fb_y^-]$. η_a can be interpreted as a root filter that measures the compatibility between the appearance of an image $g(\mathbf{x})$ and a class label $y = a$. And η is simply the concatenation of η_a for all $a \in \mathcal{Y}$.

The parametrization of $\theta^\top \cdot \Phi(y, \mathbf{h}, \mathbf{x})$ is similar to that used in object recognition [4]. But there are two important differences. First of all, our definition of the unary potential function $\phi(\cdot)$ encodes both appearance and spatial information of the patches. Secondly, we have a potential function $\omega(\cdot)$ describing the large scale appearance of the whole image. The representation in Quattoni et al. [4] only models local patches extracted from the image. This may be appropriate for object recognition. But for human action recognition, it is not clear that local patches can be sufficiently informative. We will demonstrate this experimentally in Sec. 6.

3.3 Learning and Inference

Let $D = (\langle \mathbf{x}^{(1)}, y^{(1)} \rangle, \langle \mathbf{x}^{(2)}, y^{(2)} \rangle, ..., \langle \mathbf{x}^{(N)}, y^{(N)} \rangle)$ be a set of labeled training examples, the model parameters θ are learned by maximizing the conditional log-likelihood on the training images:

$$\begin{aligned} \theta^* &= \arg\max_\theta \mathcal{L}(\theta) = \arg\max_\theta \sum_{t=1}^{N} \mathcal{L}^t(\theta) \\ &= \arg\max_\theta \sum_{t=1}^{N} \log p(y^{(t)} | \mathbf{x}^{(t)}; \theta) \\ &= \arg\max_\theta \sum_{t=1}^{N} \log \left(\sum_{\mathbf{h}} p(y^{(t)}, \mathbf{h} | \mathbf{x}^{(t)}; \theta) \right) \end{aligned} \quad (6)$$

where $\mathcal{L}^t(\theta)$ denotes the conditional log-likelihood of the t-th training example, and $\mathcal{L}(\theta)$ denotes the conditional log-likelihood of the whole training set D. Different from conditional random field (CRF) [27], the objective function $\mathcal{L}(\theta)$ of HCRF is not concave, due to the hidden variables $\mathbf{h}$. But we can still use gradient ascent to find θ that is locally optimal. The gradient of the conditional log-likelihood $\mathcal{L}^t(\theta)$ with respect to the t-th training image $(\mathbf{x}^{(t)}, y^{(t)})$ can be calculated as:

$$\begin{aligned} \frac{\partial \mathcal{L}^t(\theta)}{\partial \alpha} &= \sum_{j \in \mathcal{V}} \Big[\mathbb{E}_{p(h_j | y^{(t)}, \mathbf{x}^{(t)}; \theta)} \phi(x_j^{(t)}, h_j) \\ &\quad - \mathbb{E}_{p(h_j, y | \mathbf{x}^{(t)}; \theta)} \phi(x_j^{(t)}, h_j) \Big] \\ \frac{\partial \mathcal{L}^t(\theta)}{\partial \beta} &= \sum_{j \in \mathcal{V}} \Big[\mathbb{E}_{p(h_j | y^{(t)}, \mathbf{x}^{(t)}; \theta)} \varphi(h_j, y^{(t)}) \end{aligned}$$

$$
\begin{aligned}
& -\mathbb{E}_{p(h_j,y|\mathbf{x}^{(t)};\theta)}\varphi(h_j,y)\Big] \\
\frac{\partial \mathcal{L}^t(\theta)}{\partial \gamma} &= \sum_{(j,k)\in\mathcal{E}} \Big[\mathbb{E}_{p(h_j,h_k|y^{(t)},\mathbf{x}^{(t)};\theta)}\psi(y^{(t)},h_j,h_k) \\
& -\mathbb{E}_{p(h_j,h_k,y|\mathbf{x}^{(t)};\theta)}\psi(y,h_j,h_k)\Big] \\
\frac{\partial \mathcal{L}^t(\theta)}{\partial \eta} &= \omega(y^{(t)},\mathbf{x}^{(t)}) - \mathbb{E}_{p(y|\mathbf{x}^{(t)};\theta)}\omega(y,\mathbf{x}^{(t)}) \qquad (7)
\end{aligned}
$$

The expectations needed for computing the gradient can be obtained by belief propagation [4].

3.4 Implementation Details

Now we describe several details about how the above ideas are implemented.

Learning root filter η: Given a set of training images $\langle \mathbf{x}^{(t)}, y^{(t)} \rangle$, we firstly learn the root filter η by solving the following optimization problem:

$$
\begin{aligned}
\eta^* &= \arg\max_{\eta} \sum_{t=1}^{N} \log \mathcal{L}^{root}(y^{(t)}|\mathbf{x}^{(t)};\eta) \\
&= \arg\max_{\eta} \sum_{t=1}^{N} \log \frac{\exp\left(\eta^\top \cdot \omega(y^{(t)},\mathbf{x}^{(t)})\right)}{\sum_y \exp\left(\eta\top \cdot \omega(y,\mathbf{x}^{(t)})\right)} \qquad (8)
\end{aligned}
$$

In other words, η^* is learned by only considering the feature vector $\omega(\cdot)$. This reduces the number of parameters need to be considered initially. Similar tricks have been used in [3], [32]. We then use η^* as the starting point for η in the gradient ascent (Eq. 7). Other parameters α, β, γ are initialized randomly.

Patch initialization: We use a simple heuristic similar to that used in [3] to initialize ten salient patches on every training image from the root filter η^* trained above. For each training image I with class label a, we apply the root filter η_a on I, then select an rectangle region of size 5×5 in the image that has the most positive energy. We zero out the weights in this region and repeat until ten patches are selected. See Fig. 6(a) for examples of the patches found in some images. The tree $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is formed by running a minimum spanning tree algorithm over the ten patches.

Inference: During testing, we do not know the class label of a given test image, so we cannot use the patch initialization described above to initialize the patches, since we do not know which root filter to use. Instead, we run root filters from all the classes on a test image, then calculate the probabilities of all possible instantiations of patches under our learned model, and classify the image by picking the class label that gives the maximum of the these probabilities.

4 Max-Margin Hidden Conditional Random Fields

In this section, we present an alternative training method for learning the model parameter θ. Our learning method is inspired by the success of max-margin methods in machine learning [3], [28], [30], [33]. Given a learned model, the classification is achieved by first finding the best labeling of the hidden parts for each action, then picking the action label with the highest score. The learning algorithm aims to set the model parameters so that the scores of correct action labels on the training data are higher than the scores of incorrect action labels by a large margin. We call our approach *Max-Margin Hidden Conditional Random Fields (MMHCRF).*

4.1 Model Formulation

We assume an $\langle \mathbf{x}, y \rangle$ pair is scored by a function of the form:

$$
f_\theta(\mathbf{x}, y) = \max_{\mathbf{h}} \theta^\top \Phi(\mathbf{x}, \mathbf{h}, y) \qquad (9)
$$

Similar to HCRF, θ is the model parameter and $\mathbf{h}$ is a vector of hidden variables. Please refer to Sec. 3.2 for details description of $\theta^\top \Phi(\mathbf{x}, \mathbf{h}, y)$. In this paper, we consider the case in which $\mathbf{h} = (h_1, h_2, ..., h_m)$ forms a tree-structured undirected graphical model, but our proposed model is a rather general framework and can be applied to a wide variety of structures. We will briefly discuss them in Sec. 4.4. Similar to latent SVMs, MMHCRFs are instances of the general class of energy-based models [34].

The goal of learning is to learn the model parameter θ, so that for a new example $\mathbf{x}$, we can classify $\mathbf{x}$ to be class y^* if $y^* = \arg\max_y f(\mathbf{x}, y)$.

In analogy to classical SVMs, we would like to train θ from labeled examples D by solving the following optimization problem:

$$
\begin{aligned}
\min_{\theta,\xi} \quad & \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{N} \xi^{(t)} \\
\text{s.t.} \quad & f_\theta(\mathbf{x}^{(t)}, y) - f_\theta(\mathbf{x}^{(t)}, y^{(t)}) \le \xi^{(t)} - 1, \quad \forall t, \quad \forall y \neq y^t \\
& \xi^{(t)} \ge 0, \qquad \forall t \qquad (10)
\end{aligned}
$$

where C is the trade-off parameter similar to that in SVMs, and $\xi^{(t)}$ is the slack variable for the t-th training example to handle the case of soft margin.

The optimization problem in (10) is equivalent to the following optimization problem:

$$
\begin{aligned}
\min_{\theta,\xi} \quad & \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{N} \xi^{(t)} \\
\text{s.t.} \quad & \max_{\mathbf{h}} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \max_{\mathbf{h}'} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}', y^{(t)}) \\
& \le \xi^{(t)} - \delta(y, y^{(t)}), \quad \forall t, \quad \forall y \\
\text{where} \quad & \delta(y, y^{(t)}) = \begin{cases} 1 & \text{if } y \neq y^{(t)} \\ 0 & \text{otherwise} \end{cases} \qquad (11)
\end{aligned}
$$

An alternative to the formulation in Eq. 11 is to convert the multi-class classification problem into several binary classifications (e.g. one-against-all), then solve each of them using LSVM. Although this alternative is simple and powerful, it cannot capture correlations between different classes since those binary problems are independent [33]. We will demonstrate experimentally (Sec. 6)

that this alternative does not perform as well as our proposed method.

4.2 Semi-Convexity and Primal Optimization

Similar to LSVMs, MMHCRFs have the property of semi-convexity. Note that $f_\theta(\mathbf{x}, y)$ is a maximum of a set of functions, each of which is linear in θ, so $f_\theta(\mathbf{x}, y)$ is convex in θ. If we restrict the domain of $\mathbf{h}'$ in (11) to a single choice, the optimization problem of (11) becomes convex [35]. This is in analog to restricting the domain of the latent variables for the positive examples to a single choice in LSVMs [3]. But here we are dealing with multi-class classification, our "positive examples" are those $\langle \mathbf{x}^{(t)}, y\rangle$ pairs where $y = y^{(t)}$.

We can compute a local optimum of (11) using a coordinate descent algorithm similar to LSVMs [3]:

1) Holding θ, ξ fixed, optimize the latent variables $\mathbf{h}'$ for the $\langle \mathbf{x}^{(t)}, y^{(t)}\rangle$ pair:

$$\mathbf{h}^{(t)}_{y^{(t)}} = \arg\max_{\mathbf{h}'} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}', y^{(t)})$$

2) Holding $\mathbf{h}^{(t)}_{y^{(t)}}$ fixed, optimize θ, ξ by solving the following optimization problem:

$$\begin{aligned} \min_{\theta,\xi} \quad & \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{N} \xi^{(t)} \\ \text{s.t.} \quad & \max_{\mathbf{h}} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_{y^{(t)}}, y^{(t)}) \\ & \le \xi^{(t)} - \delta(y, y^{(t)}), \quad \forall t, \quad \forall y \end{aligned} \tag{12}$$

It can be shown that both steps always improve or maintain the objective [3].

The optimization problem in Step 1 can be solved efficiently for certain structures of $\mathbf{h}'$ (see Sec. 4.4 for details). The optimization problem in Step 2 involves solving a quadratic program (QP) with piecewise linear constraints. Although it is possible to solve it directly using barrier methods [35], we will not be able to take advantage of existing highly optimized solvers (e.g., CPLEX) which only accept linear constraints. It is desirable to convert (12) into a standard quadratic program with only linear constraints.

One possible way to convert (12) into a standard QP is to solve the following convex optimization problem:

$$\begin{aligned} \min_{\theta,\xi} \quad & \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{N} \xi^{(t)} \\ \text{s.t.} \quad & \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_{y^{(t)}}, y^{(t)}) \\ & \le \xi^{(t)} - \delta(y, y^{(t)}), \qquad \forall t, \forall \mathbf{h}, \forall y \end{aligned} \tag{13}$$

It is easy to see that (12) and (13) are equivalent, and all the constraints in (13) are linear. Unfortunately, the optimization problem in (13) involves an exponential number of constraints – for each example $\mathbf{x}^{(t)}$ and each possible labeling y, there are exponentially many possible $\mathbf{h}$'s.

We would like to perform optimization over a much smaller set of constraints. One solution is to use a cutting plane algorithm similar to that used in structured SVMs [30] and CRFs [36]. In a nutshell, the algorithm starts with no constraints (which corresponds to a relaxed version of (13)), then iteratively finds the "most violated" constraints and adds those constraints. It can be shown that this algorithm computes arbitrarily close approximation to the original problem of (13) by evaluating only a polynomial number of constraints.

More importantly, the optimization problem in (13) has certain properties that allow us to find and add constraints in an efficient way. For a fixed example $\mathbf{x}^{(t)}$ and a possible label y, define $\mathbf{h}^{(t)}_y$ as follows:

$$\mathbf{h}^{(t)}_y = \arg\max_{\mathbf{h}} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y)$$

Consider the following two set of constraints for the $\langle \mathbf{x}^{(t)}, y\rangle$ pair:

$$\begin{aligned} & \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_y, y) - \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_{y^{(t)}}, y^{(t)}) \\ & \quad \le \xi^{(t)} - \delta(y, y^{(t)}) \end{aligned} \tag{14}$$

$$\begin{aligned} & \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_{y^{(t)}}, y^{(t)}) \\ & \quad \le \xi^{(t)} - \delta(y, y^{(t)}), \quad \forall \mathbf{h} \end{aligned} \tag{15}$$

It is easy to see that within a local neighborhood of θ, (14) and (15) define the same set of constraints, i.e., (14) implies (15) and vice versa. This suggests that for a fixed $\langle \mathbf{x}^{(t)}, y\rangle$ pair, we only need to consider the constraint involving $\mathbf{h}^{(t)}_y$.

Putting everything together, we learn the model parameter θ by iterating the following two steps.

1) Fixing θ, ξ, optimize the latent variable $\mathbf{h}$ for each pair $\langle \mathbf{x}^{(t)}, y\rangle$ of an example $\mathbf{x}^{(t)}$ and a possible labeling y:

$$\mathbf{h}^{(t)}_y = \arg\max_{\mathbf{h}} \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y)$$

2) Fixing $\mathbf{h}^{(t)}_y \quad \forall t, \quad \forall y$, optimize θ, ξ by solving the following optimization problem:

$$\begin{aligned} \min_{\theta,\xi} \quad & \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{N} \xi^{(t)} \\ \text{s.t.} \quad & \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_y, y) - \theta^\top \Phi(\mathbf{x}^{(t)}, \mathbf{h}^{(t)}_{y^{(t)}}, y^{(t)}) \\ & \le \xi^{(t)} - \delta(y, y^{(t)}), \quad \forall t, \quad \forall y \end{aligned} \tag{16}$$

Step 1 in the above algorithm can be efficiently solved for certain structured $\mathbf{h}$ (Sec. 4.4). Step 2 involves solving a quadratic program with $N \times |\mathcal{Y}|$ constraints.

The optimization in (16) is very similar to the primal problem of a standard multi-class SVM [33]. In fact, if $\mathbf{h}^{(t)}_y$ is the same for different y's, it is just a standard SVM and we can use an off-the-shelf SVM solver to optimize (16). Unfortunately, the fact that $\mathbf{h}^{(t)}_y$ can vary with different y's means that we cannot directly use standard SVM packages. We instead develop our own optimization algorithm.

4.3 Dual Optimization

In analog to classical SVMs, it is helpful to solve the problem in (16) by examining its dual. To simplify the notation, let us define $\Psi(\mathbf{x}^{(t)}, y) = \Phi(\mathbf{x}^{(t)}, \mathbf{h}_y^{(t)}, y) - \Phi(\mathbf{x}^{(t)}, \mathbf{h}_{y^{(t)}}^{(t)}, y^{(t)})$. Then the dual problem of (16) can be written as follows:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{t=1}^{N}\sum_{y}\alpha_{t,y}\delta(y, y^{(t)}) - \frac{1}{2}||\sum_{t=1}^{N}\sum_{y}\alpha_{t,y}\Psi(\mathbf{x}^{(t)}, y)||^2 \\ \text{s.t.} \quad & \sum_{y}\alpha_{t,y} = C, \quad \forall t \\ & \alpha_{t,y} \geq 0, \quad \forall t, \quad \forall y \end{aligned} \tag{17}$$

The primal variable θ can be obtained from the dual variables α as follows:

$$\theta = -\sum_{t=1}^{N}\sum_{y}\alpha_{t,y}\Psi(\mathbf{x}^{(t)}, y)$$

Note that (17) is quite similar to the dual form of standard multi-class SVMs. In fact, if $\mathbf{h}_y^{(t)}$ is a deterministic function of $\mathbf{x}^{(t)}$, (17) is just a standard dual form of SVMs.

Similar to classical SVMs, we can also obtain a kernelized version of the algorithm by defining a kernel function of size $N \times |\mathcal{Y}|$ by $N \times |\mathcal{Y}|$ in the following form:

$$K(t, y; s, y') = \Psi(\mathbf{x}^{(t)}, y)^\top \Psi(\mathbf{x}^{(s)}, y')$$

Let us define α as the concatenation of $\{\alpha_{t,y} : \forall t \ \ \forall y\}$, so the length of α is $N \times |\mathcal{Y}|$. Define Δ as a vector of the same length. The (t, y)-th entry of Δ is 1 if $y \neq y^{(t)}$, and 0 otherwise. Then (17) can be written as:

$$\begin{aligned} \max_{\alpha} \quad & \alpha^\top\Delta - \frac{1}{2}\alpha^\top K\alpha \\ \text{s.t.} \quad & \sum_{y}\alpha_{t,y} = C, \qquad \forall t \\ & \alpha_{t,y} \geq 0, \qquad \forall t, \quad \forall y \end{aligned} \tag{18}$$

Note the matrix K in (18) only depends on the dot-product between feature vectors of different $\langle \mathbf{x}^{(t)}, y\rangle$ pairs. So our model has a very intuitive and interesting interpretation – it defines a particular kernel function that respects the latent structures.

It is easy to show that the optimization problem in (17) is concave, so we can find its global optimum. But the number of variables is $N \times |\mathcal{Y}|$, where N is the number of training examples, and $|\mathcal{Y}|$ is the size of all possible class labels. So it is infeasible to use a generic QP solver to optimize it.

Instead, we decompose the optimization problem of (17) and solve a series of smaller QPs. This is similar to the sequential minimal optimization (SMO) used in SVM [33], [37] and M^{3}N [28]. The basic idea of this algorithm is to choose all the $\{\alpha_{t,y} : \forall y \in \mathcal{Y}\}$ for a particular training example $\mathbf{x}^{(t)}$ and fix all the other variables $\{\alpha_{s,y'} : \forall s : s \neq t, \forall y' \in \mathcal{Y}\}$ that do not involve $\mathbf{x}^{(t)}$. Then instead of solving a QP involving all the variables $\{\alpha_{t,y} : \forall t, \forall y\}$, we can solve a much smaller QP only involving $\{\alpha_{t,y} : \forall y\}$. The number of variables of this smaller QP is $|\mathcal{Y}|$, which is much smaller than $N \times |\mathcal{Y}|$.

First we write the objective of (17) in terms of $\{\alpha_{t,y} : \forall y\}$ as follows:

$$\begin{aligned} & \mathcal{L}(\{\alpha_{t,y} : \forall y\}) \\ = & \sum_{y}\alpha_{t,y}\delta(y, y^{(t)}) - \frac{1}{2}\Big[||\sum_{y}\alpha_{t,y}\Psi(\mathbf{x}^{(t)}, y)||^2 \\ & +2\Big(\sum_{y}\alpha_{t,y}\Psi(\mathbf{x}^{(t)}, y)\Big)^\top\Big(\sum_{s:s\neq t}\sum_{y'}\alpha_{s,y'}\Psi(\mathbf{x}^{(s)}, y')\Big)\Big] \\ & +\text{other terms not involving } \{\alpha_{t,y} : \forall y\} \end{aligned}$$

The smaller QP corresponding to $\langle \mathbf{x}^{(t)}, y^{(t)}\rangle$ can be written as follows:

$$\begin{aligned} \max_{\alpha_{t,y}:\forall y} \quad & \mathcal{L}(\{\alpha_{t,y} : \forall y\}) \\ \text{s.t.} \quad & \sum_{y}\alpha_{t,y} = C \\ & \alpha_{t,y} \geq 0, \quad \forall y \end{aligned} \tag{19}$$

Note $\sum_{s:s\neq t}\sum_{y'}\alpha_{s,y'}\Psi(\mathbf{x}^{(s)}, y')$ can be written as:

$$-\theta - \sum_{y}\alpha_{t,y}\Psi(\mathbf{x}^{(t)}, y)$$

So as long as we maintain (and keep updating) the global parameter θ and keep track of $\alpha_{t,y}$ and $\Psi(\mathbf{x}^{(t)}, y)$ for each example $\langle \mathbf{x}^{(t)}, y^{(t)}\rangle$, we do not need to actually do the summation $\sum_{s:s\neq t}\sum_{y'}$ when optimizing (19). In addition, when we solve the QP involving $\alpha_{t,y}$ for a fixed t, all the other constraints involving $\alpha_{s,y}$ where $s \neq t$ are not affected. This is not the case if we try to optimize the primal problem in (16). If we try to optimize the primal variable θ by only considering the constraints involving the t-th examples, it is possible that the new θ obtained from the optimization might violate the constraints imposed by other examples. There is also work [38] showing that the dual optimization has a better convergence rate.

4.4 Finding the Optimal $\mathbf{h}$

The alternating coordinate descent algorithm for learning the model parameter θ described in Sec. 4.2 assumes we have an inference algorithm for finding the optimal $\mathbf{h}^*$ for a fixed $\langle \mathbf{x}, y\rangle$ pair:

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \theta^\top\Phi(\mathbf{x}, \mathbf{h}, y) \tag{20}$$

In order to adopt our approach to problems involving different latent structures, this is the only component of the algorithm that needs to be changed.

If $\mathbf{h} = (h_1, h_2, ..., h_m)$ forms a tree-structured graphical model, the inference problem in (20) can be solved exactly, e.g., using the Viterbi dynamic programming

algorithm for trees. We can also solve it using standard linear programming as follows [29], [39]. We introduce variables $\mu_{j\mathfrak{a}}$ to denote the indicator $\mathbb{1}_{\{h_j=\mathfrak{a}\}}$ for all vertices $j \in \mathcal{V}$ and their values $\mathfrak{a} \in \mathcal{H}$. Similarly, we introduce variables $\mu_{jk\mathfrak{ab}}$ to denote the indicator $\mathbb{1}_{\{h_j=\mathfrak{a},h_k=\mathfrak{b}\}}$ for all edges $(j,k) \in \mathcal{E}$ and the values of their nodes, $\mathfrak{a} \in \mathcal{H}$, $\mathfrak{b} \in \mathcal{H}$. We use $\tau_j(h_j)$ to collectively represent the summation of all the unary potential functions in (1) that involve the node $j \in \mathcal{V}$. We use $\tau_{jk}(h_j, h_k)$ to collectively represent the summation of all the pairwise potential functions in (1) that involve the edge $(j,k) \in \mathcal{E}$. The problem of finding of optimal $\mathbf{h}^*$ can be formulated into the following linear programming (LP) problem:

$$\begin{aligned}
&\max_{0\leq\mu\leq 1} \sum_{j\in\mathcal{V}}\sum_{\mathfrak{a}\in\mathcal{H}} \mu_{j\mathfrak{a}}\tau_j(\mathfrak{a}) + \sum_{(j,k)\in\mathcal{E}}\sum_{\mathfrak{a}\in\mathcal{H}}\sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}}\tau_{jk}(\mathfrak{a},\mathfrak{b}) \\
&\text{s.t.} \quad \sum_{\mathfrak{a}\in\mathcal{H}} \mu_{j\mathfrak{a}} = 1, \quad \forall j \in \mathcal{V} \\
&\qquad \sum_{\mathfrak{a}\in\mathcal{H}}\sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = 1, \quad \forall (j,k) \in \mathcal{E} \\
&\qquad \sum_{\mathfrak{a}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = \mu_{k\mathfrak{b}}, \quad \forall (j,k) \in \mathcal{E}, \quad \forall \mathfrak{b} \in \mathcal{H} \\
&\qquad \sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = \mu_{j\mathfrak{a}}, \quad \forall (j,k) \in \mathcal{E}, \quad \forall \mathfrak{a} \in \mathcal{H}
\end{aligned} \tag{21}$$

If the optimal solution of this LP is integral, we can recover $\mathbf{h}^*$ from μ^* very easily. It has been shown that if $\mathcal{E}$ forms a forest, the optimal solution of this LP is guaranteed to be integral [29], [39]. For general graph topology, the optimal solution of this LP can be fractional, which is not surprising, since the problem in (20) is NP-hard for general graphs. Although the LP formulation does not seem to be particularly advantageous in the case of tree-structured models, since they can be solved by Viterbi dynamic programming anyway, the LP formulation provides a more general way of approaching other structures (e.g., Markov networks with sub-modular potentials, matching [29]).

5 Discussion

HCRF and MMHCRF can be thought of as two different approaches for solving the general problem of *classification with structured latent variables*. Many problems in computer vision can be formulated in this general problem setting. Consider the following three vision tasks. (1) *Pedestrian detection*: it can be formulated as a binary classification problem that classifies an image patch x to be +1 (pedestrian) or 0 (non-pedestrian). The locations of the body parts can be considered as latent variables in this case, since most of the existing training datasets for pedestrian detection do not provide this information. These latent variables are also *structured* – e.g., the location of the torso imposes certain constraints on the possible locations of other parts. Previous approaches [3], [40] usually use a tree-structured model to model these constraints. (2) *Object recognition*: this problem is to assign a class label to an image if it contains the object of interest. If we consider the figure/ground labeling of pixels of the image as latent variables, object recognition is also a problem of classification with latent variables. The latent variables are also structured, typically represented by a grid-structured graph. (3) *Object identification*: given two images, the task is to decide whether these are two images of the same object or not. If an image is represented by a set of patches found by interest point operators, one particular way to solve this problem is to first find the correspondence between patches in the two images, then learn a binary classifier based on the result of the correspondence [41]. Of course, the correspondence information is "latent" – not available in the training data, and "structured" – assuming one patch in one image matches to one or zero patches in the other image, this creates a combinatorial structure [29]. This general problem setting occurs in other fields as well. For example, in natural language processing, Cherry and Quick [42] use a similar idea for sentence classification by considering the parse tree of a sentence as the hidden variable. In bioinformatics, Yu & Joachims [43] solve the motif finding problem in yeast DNA by considering the positions of motifs as hidden variables.

One simplified approach to solve the above-mentioned problems is to ignore the latent structures, and treat them as standard classification problems, e.g., Dalal & Triggs [5] in the case of pedestrian detection. However, there is evidence [3], [4], [15], [40] showing that incorporating latent structures into the system can improve the performance.

Learning with latent structures has a long history in machine learning. In visual recognition, probabilistic latent semantic analysis (pLSA) [44] and latent Dirichlet allocation (LDA) [45] are representative examples of generative classification models with latent variables that have been widely used. As far as we know, however, there has only been some recent effort [3], [4], [15], [16], [43], [46] on incorporating latent variables into discriminative models. It is widely believed in the machine learning literature that discriminative methods typically outperform generative methods[1] (e.g. [27]), so it is desirable to develop discriminative methods with structured latent variables.

MMHCRFs are closely related to HCRFs. The main difference lies in their different learning criteria – maximizing the margin in MMHCRFs, and maximizing the conditional likelihood in HCRFs. As a result, the learning algorithms for MMHCRFs and HCRFs involve solving two different types of inference problems – maximizing over $\mathbf{h}$, versus summing over $\mathbf{h}$. In the following, we compare and analyze these two approaches from both computational and modeling perspectives, and argue why MMHCRFs are preferred over HCRFs in many applications.

1. We acknowledge that this view is not unchallenged, e.g. [47].

5.1 Computational Perspective

HCRFs and MMHCRFs share some commonality in terms of their learning algorithms. Both are iterative methods. During each iteration, both of them require solving an inference on each training example. Since the learning objectives of both learning algorithms are non-convex, both of them can only find local minimum. The computational complexity of both learning algorithms involve three factors: (1) the number of iterations; (2) the number of training examples; (3) the computation needed on each example during an iteration. Of course, the number of training examples remains the same in both algorithms. If we assume both algorithms take the same number of iterations, the main difference in the computational complexity of these two algorithms comes from the third factor, i.e. the computation needed on each example during an iteration. This complexity is mainly dominated by the run time of the inference algorithm for each algorithm – summing over $\mathbf{h}$ in HCRFs, and maximizing over $\mathbf{h}$ in MMHCRFs. In other words, the computational complexity of the learning problem for either HCRFs or MMHCRFs is determined by the complexity of the corresponding inference problem. So in order to understand the difference between HCRFs and MMHCRFs in terms of their learning algorithm complexity, we only need to focus on the complexity of their inference algorithms.

From the computational perspective, if $\mathbf{h}$ has a tree structure and $|\mathcal{H}|$ is relatively small, both inference problems (max vs. sum) can be solved exactly, using dynamic programming and belief propagation, respectively. But the inference problem (maximization) in MMHCRFs can deal with a much wider range of latent structures. Here are a few examples [29] (although these problems are not addressed in this paper) in computer vision:

- Binary Markov networks with sub-modular potentials, commonly encountered in figure/ground segmentation [48]. MMHCRFs can use LP [29] or graph-cut [48] to solve the inference problem. For HCRFs, the inference problem can only be solved approximately, e.g., using loopy BP or mean-field variational methods.
- Matching/correspondence (see the object identification example mentioned above, or the examples in [29]). The inference of this structure can be solved by MMHCRFs using LP [29], [49]. It is not clear how HCRFs can be used in this situation, since it requires summing over all the possible matchings.
- Tree-structures, but each node in $\mathbf{h}$ can have a large number of possible labels (e.g., all the possible pixel locations in an image), i.e., $|\mathcal{H}|$ is big. If the pairwise potentials have certain properties, distance transform [8] can be applied in MMHCRFs to solve the inference problem. This is essentially what has been done in [3]. This inference problem for HCRFs can be solved using convolution. But distance transform ($O(|\mathcal{H}|)$) is more efficient than convolution ($O(|\mathcal{H}|\log|\mathcal{H}|)$).

5.2 Modeling Perspective

From the modeling perspective, we believe MMHCRFs are better suited for classification than HCRFs. This is because HCRFs require summing over exponentially many $\mathbf{h}$'s. In order to maximize the conditional likelihoods, the learning algorithm of HCRFs has to try very hard to push the probabilities of many "wrong" labellings of $\mathbf{h}$'s to be close to zero. But in MMHCRFs, the learning algorithm only needs to push apart the "correct" labeling and its next best competitor. Conceptually, the modeling criterion of MMHCRFs is easier to achieve and more relevant to classification than that of HCRFs. In the following, we give a detailed analysis of the differences between HCRFs and MMHCRFs in order to gain the insights.

Max-margin vs. log-likelihood: The first difference between MMHCRFs and HCRFs lie in their different training criteria, i.e., maximizing the margin in MMHCRFs, and maximizing the conditional log-likelihood in HCRFs. Here we explain why max-margin is a better training criterion using a synthetic classification problem illustrated in Table 1. To simplify the discussion, we assume a regular classification problem (without hidden structures). We assume three possible class labels, i.e. $\mathcal{Y} = \{1, 2, 3\}$. For simplicity, let us assume there is only one datum $\mathbf{x}$ in the training dataset, and the ground-truth label for $\mathbf{x}$ is $Y = 2$. Suppose we have two choices of model parameters $\theta^{(1)}$ and $\theta^{(2)}$. The conditional probabilities $p(Y|\mathbf{x};\theta^{(1)})$ and $p(Y|\mathbf{x};\theta^{(2)})$ are shown in the two rows in Table 1. If we choose the model parameter by maximizing the conditional probability on the training data, we will choose $\theta^{(1)}$ over $\theta^{(2)}$, since $p(Y = 2|\mathbf{x};\theta^{(1)}) > p(Y = 2|\mathbf{x};\theta^{(2)})$. But the problem with $\theta^{(1)}$ is that $\mathbf{x}$ will be classified as $Y = 3$ if we use $\theta^{(1)}$ as the model parameter, since $p(Y = 3|\mathbf{x};\theta^{(1)}) > p(Y = 2|\mathbf{x};\theta^{(1)})$.

Let us use y^* to denote the ground-truth label of $\mathbf{x}$, and y' to denote the "best competitor" of y^*. If we define $y^{opt} = \arg\max_y p(Y = y|x;\theta)$, y' can be formally defined as follows:

$$y' = \begin{cases} y^{opt} & \text{if } y^* \neq y^{opt} \\ \arg\max_{y:y\neq y^*} p(Y = y|x;\theta) & \text{if } y^* = y^{opt} \end{cases}$$

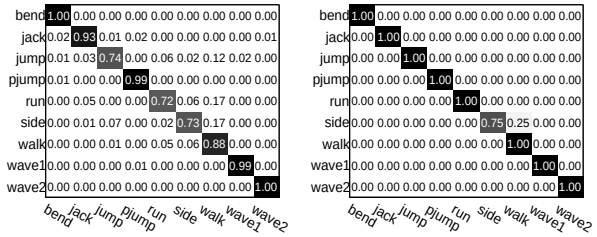
If we define $margin(\theta) = p(Y = y^*|\mathbf{x};\theta) - p(Y = y'|\mathbf{x};\theta)$, we can see that $margin(\theta^{(2)}) = 0.45 - 0.43 = 0.02$ and $margin(\theta^{(1)}) = 0.48 - 0.5 = -0.02$. So the max-margin criterion will choose $\theta^{(2)}$ over $\theta^{(1)}$. If we use $\theta^{(2)}$ as the model parameter to classify $\mathbf{x}$, we will get the correct label $Y = 2$.

This simple example shows that the max-margin training criterion is more closely related to the classification problem we are trying to solve in the first place.

Maximization vs. summation: Another difference between the MMHCRF and the HCRF is that the former

TABLE 1
A toy example illustrating why the learning criterion of MMHCRF (i.e., maximizing the margin) is more relevant to classification than that of HCRF (i.e., maximizing the log-likelihood). We assume a classification problem with three class labels (i.e., $\mathcal{Y} = \{1, 2, 3\}$) and one training datum. The table shows the conditional probability $p(Y|\mathbf{x}; \theta^{(1)})$ and $p(Y|\mathbf{x}; \theta^{(2)})$ where $Y = 1, 2, 3$ for two choices of model parameters $\theta^{(1)}$ and $\theta^{(2)}$.

	$Y=1$	$Y=2$	$Y=3$
$\theta = \theta^{(1)}$	0.02	0.48	0.5
$\theta = \theta^{(2)}$	0.12	0.45	0.43



(a) per-frame (b) per-video

Fig. 4. Confusion matrices of classification results (per-frame and per-video) of HCRFs on the Weizmann dataset. Rows are ground truths, and columns are predictions.

requires maximizing over $\mathbf{h}$'s, while the latter requires summing over $\mathbf{h}$'s. In addition to computational issues mentioned above, the summation over all the possible $\mathbf{h}$'s in HCRFs may cause other problems as well. To make the discussion more concrete, let us consider the pedestrian detection setting in [3]. In this setting, $\mathbf{x}$ is an image patch, y is a binary class label +1 or 0 to indicate whether the image patch $\mathbf{x}$ is a pedestrian or not. The hidden variable $\mathbf{h}$ represents the locations of body parts. Recall that HCRFs need to compute the summation of all possible $\mathbf{h}$'s in the following form:

$$p(y|\mathbf{x}; \theta) = \sum_{\mathbf{h}} p(y, \mathbf{h}|\mathbf{x}; \theta) \tag{22}$$

The intuition behind Eq. 22 is that a correct labeling of $\mathbf{h}$ will have a higher value of probability $p(y, \mathbf{h}|\mathbf{x}; \theta)$, while an incorrect labeling of $\mathbf{h}$ will have a lower value. Hence correct part locations will contribute more to $p(y|\mathbf{x}; \theta)$ in Eq. 22.

However, this is not necessarily the case. In an image, there are exponentially many possible placements of part locations. That means $\mathbf{h}$ has exponentially many possible configurations. But only a very small number of those configurations are "correct" ones, i.e. the ones that roughly correspond to correct locations of body parts. Ideally, a good model will put high probabilities on those "correct" $\mathbf{h}$'s and lower probabilities (close to 0) on "incorrect" ones. An incorrect $\mathbf{h}$ only carries a small probability, but since there are exponentially many of them, the summation in Eq. 22 can still be dominated by those incorrect $\mathbf{h}$'s. This surprising effect is exactly the opposite of what we have expected the model to behave. This surprising effect is partly due to the high dimensionality of the latent variable $\mathbf{h}$. Many counter-intuitive properties of high dimensional spaces have also been observed in statistics and machine learning.

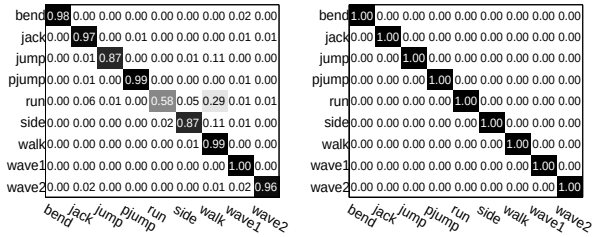
We would like to emphasize that we do not mean to discredit HCRFs. In fact, we believe HCRFs have their own advantages. The main merit of HCRFs lies in their rigorous probabilistic semantics. A probabilistic model has two major advantages compared with non-probabilistic alternatives (e.g., energy-based models like MMHCRFs). First of all, it is usually intuitive and straightforward to incorporate prior knowledge within a probabilistic model, e.g. by choosing appropriate prior distributions, or by building hierarchical probabilistic models. Secondly, in addition to predicting the best labels, a probabilistic model also provides posterior probabilities and confidence scores of its predictions. This might be valuable for certain scenarios, e.g. in building cascades of classifiers. HCRFs and MMHCRFs are simply two different ways of solving problems. They both have their own advantages and limitations. Which approach to use depends on the specific problem at hand.

6 Experiments

We test our algorithm on two publicly available datasets that have been widely used in action recognition: Weizmann human action dataset [9], and KTH human motion dataset [14]. Performance on these benchmarks is saturating – state-of-the-art approaches achieve near-perfect results. We show our method achieves results comparable to the state-of-the-art, and more importantly that our extended HCRF model significantly outperforms a direct application of the original HCRF model [4].

Weizmann dataset: The Weizmann human action dataset contains 83 video sequences showing nine different people, each performing nine different actions: running, walking, jumping jack, jumping forward on two legs, jumping in place on two legs, galloping sideways, waving two hands, waving one hand, bending. We track and stabilize the figures using the background subtraction masks that come with this dataset. See Figure 6(a) for sample frames of this dataset.

We randomly choose videos of five subjects as training set, and the videos in the remaining four subjects as test set. We learn three HCRF models with different sizes of possible part labels, $|\mathcal{H}| = 6, 10, 20$. Our model classifies every frame in a video sequence (i.e., per-frame classification), but we can also obtain the class label for the whole video sequence by the majority voting of the labels of its frames (i.e., per-video classification). We show the confusion matrices of HCRFs and MMHCRFs with $|\mathcal{H}| = 10$ for both per-frame and per-video classification in Fig. 4 and Fig. 5.



(a) per-frame (b) per-video

Fig. 5. Confusion matrices of classification results of MMHCRFs on Weizmann dataset.

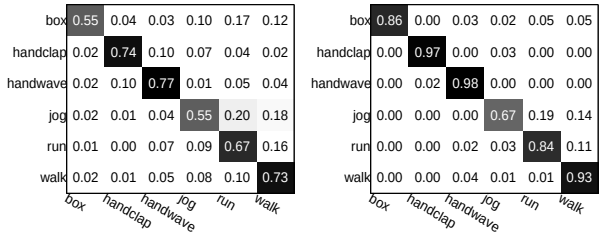
TABLE 2
Comparison of two baseline systems with our approach(HCRF and MMHCRF) on the Weizmann and KTH datasets.

method	Weizmann		KTH	
	per-frame	per-video	per-frame	per-video
root model	0.7470	0.8889	0.5377	0.7339
local HCRF				
$\|\mathcal{H}\|$=6	0.5722	0.5556	0.4749	0.5607
$\|\mathcal{H}\|$=10	0.6656	0.6944	0.4452	0.5814
$\|\mathcal{H}\|$=20	0.6383	0.6111	0.4282	0.5504
HCRF				
$\|\mathcal{H}\|$=6	**0.8682**	**0.9167**	**0.6633**	**0.7855**
$\|\mathcal{H}\|$=10	**0.9029**	**0.9722**	**0.6698**	**0.8760**
$\|\mathcal{H}\|$=20	**0.8557**	**0.9444**	**0.6444**	**0.7512**
MMHCRF				
$\|\mathcal{H}\|$=6	**0.8996**	**0.9722**	**0.7064**	**0.8475**
$\|\mathcal{H}\|$=10	**0.9311**	**1.0000**	**0.7853**	**0.9251**
$\|\mathcal{H}\|$=20	**0.8891**	**0.9722**	**0.7486**	**0.8966**

We compare our system to two baseline methods. The first baseline (root model) only uses the root filter $\eta^\top \cdot \omega(y, \mathbf{x})$, which is simply a discriminative version of Efros et al. [10]. The second baseline (local HCRF) is a direct application of the original HCRF model [4]. It is similar to our model, but without the root filter $\eta^\top \cdot \omega(y, \mathbf{x})$, i.e., local HCRF only uses the root filter to initialize the salient patches, but does not use it in the final model. The comparative results are shown in Table 2. Our approach significantly outperforms the two baseline methods. We also compare our results(with $|\mathcal{H}| = 10$) with previous work in Table 3. Note [9] classifies space-time cubes. It is not clear how it can be compared with other methods that classify frames or videos. Our result is significantly better than [24], and comparable to [22]. Although we accept the fact that the comparison is not completely fair, since [24] does not use any tracking or background subtraction.

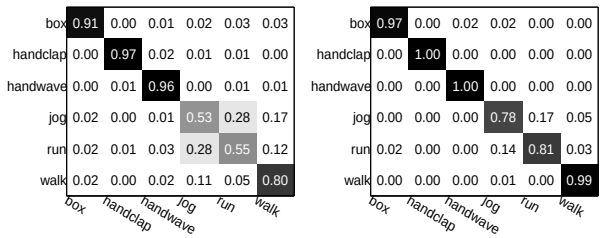
TABLE 3
Comparison of classification accuracy with previous work on the Weizmann dataset.

	per-frame	per-video	per-cube
MMHCRF	**0.9311**	**1**	N/A
HCRF	**0.9029**	**0.9722**	N/A
Jhuang et al. [22]	N/A	0.988	N/A
Niebles & Fei-Fei [24]	0.55	0.728	N/A
Blank et al. [9]	N/A	N/A	0.9964



(a) per-frame (b) per-video

Fig. 7. Confusion matrices of classification results of HCRFs on the KTH dataset.



(a) per-frame (b) per-video

Fig. 8. Confusion matrices of classification results of MMHCRFs on the KTH dataset.

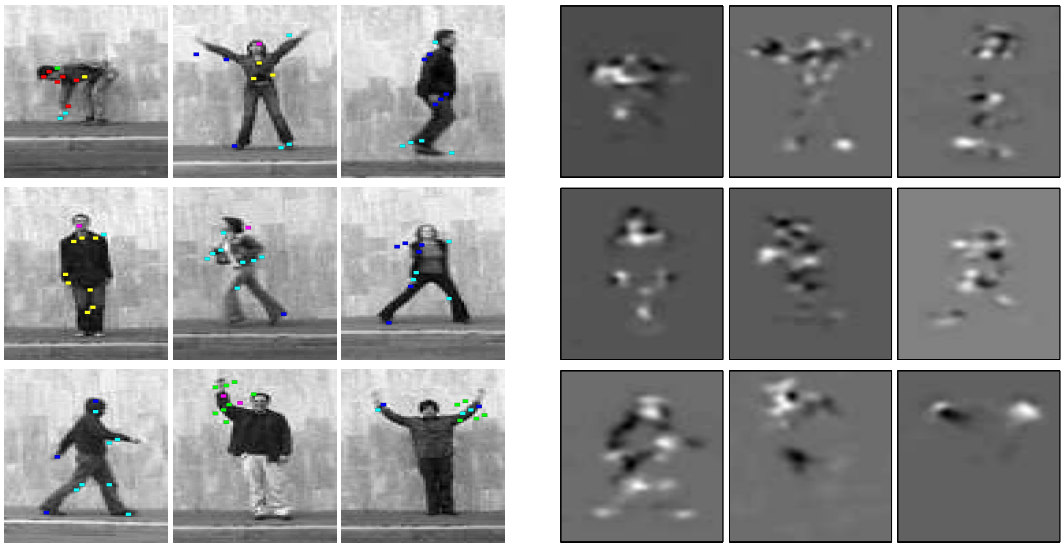
We visualize the parts by HCRFs in Fig. 6(a). Each patch is represented by a color that corresponds to the most likely part label of that patch. We also visualize the root filters applied on these images in Fig. 6(b). The visualization on MMHCRFs is similar, it is omitted due to space constraints.

KTH dataset: The KTH human motion dataset contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. See Figure 9(a) for sample frames. We first run an automatic preprocessing step to track and stabilize the video sequences, so that all the figures appear in the center of the field of view.

We split the videos roughly equally into training/test sets. The confusion matrices (with $|\mathcal{H}| = 10$) for both per-frame and per-video classification are shown in Fig. 7 and Fig. 8. The comparison with the two baseline algorithms is summarized in Table 2. Again, our approach outperforms the two baselines systems.

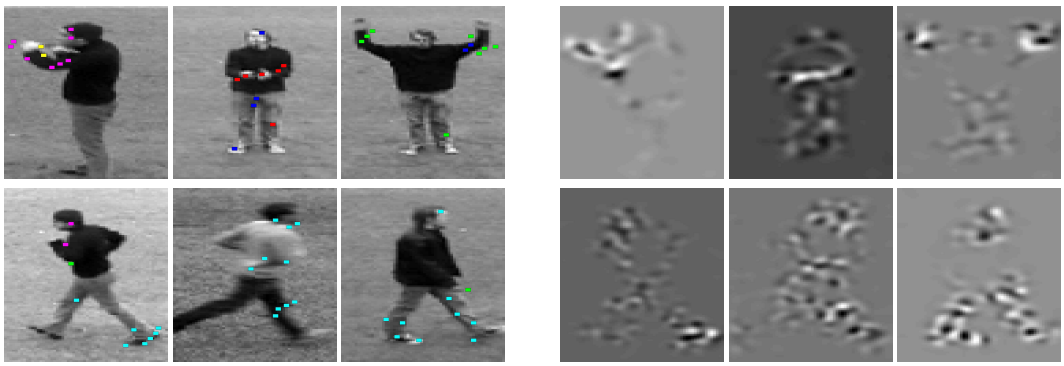
The comparison with other approaches is summarized in Table 4. We should emphasize that we do not attempt a direct comparison, since different methods listed in Table 4 have all sorts of variations in their experiments (e.g., different split of training/test data, whether temporal smoothing is used, whether per-frame classification can be performed, whether tracking/background subtraction is used, whether the whole dataset is used etc.), which make it impossible to directly compare them. We provide the results only to show that our approach is comparable to the state-of-the-art. Similarly, we visualize the learned parts on the KTH dataset in Fig. 9.

We perform additional diagnostic experiments to an-



(a) (b)

Fig. 6. Visualization of the learned model on the Weizmann dataset: (a) Visualization of the learned parts. Patches are colored according to their most likely part labels. Each color corresponds to a part label. Some interesting observations can be made. For example, the part label represented by red seems to correspond to the "moving down" patterns mostly observed in the "bending" action. The part label represented by green seems to correspond to the motion patterns distinctive of "hand-waving" actions; (b) Visualization of root filters applied on these images. For each image with class label c, we apply the root filter η_c. The results show the filter responses aggregated over four motion descriptor channels. Bright areas correspond to positive energies, i.e., areas that are discriminative for this class. This figure is best viewed in color with PDF magnification.



(a) (b)

Fig. 9. Visualization of the learned model on the KTH dataset: (a) Visualization of the learned parts; (b) Visualization of root filters applied on these images. See the caption of Fig. 6 for detailed explanations. Similarly, we can make some interesting observations. For example, parts colored in pink, red, and green are representative of "boxing", "handclapping" and "handwaving" actions, respectively. The part colored in light blue is shared among "jogging", "running" and "walking" actions. This figure is best viewed in color with PDF magnification.

swer the following questions regarding MMHCRFs.

With or without pairwise potentials? If we remove the pairwise potentials $\gamma^\top \psi(y, h_j, h_k)$ in the model, the learning and inference will become easier, since each part label can be independently assigned. One interesting question is whether the pairwise potentials really help the recognition. To answer this question, we run MMHCRFs without the pairwise potentials on both Weizmann and KTH datasets. The results are shown in Table 5 ("no pairwise"). Comparing with the results in Table 2, we can see that MMHCRF models without pairwise potentials do not perform as well as those with pairwise potentials.

Multi-class or one-against-all? MMHCRFs directly handle multi-class classification. An alternative is to convert the multi-class problem into several binary classification problems (e.g. one-against-all). The disadvantage of this alternative is discussed in [33]. Here we demonstrate it experimentally in our application. For a K-class classification problem, we convert it into K binary classification (one-against-all) problems. Each binary classification problem is solved using the formula-

TABLE 4
Comparison of per-video classification accuracy with previous approaches on the KTH dataset.

methods	accuracy
MMHCRF	**0.9251**
HCRF	**0.8760**
Liu & Shah [50]	0.9416
Jhuang et al. [22]	0.9170
Nowozin et al. [13]	0.8704
Niebles et al. [12]	0.8150
Dollár et al. [11]	0.8117
Schuldt et al. [14]	0.7172
Ke et al. [51]	0.6296

TABLE 5
Results of learning MMHCRF models without pairwise potentials ("no pairwise"), and by converting the multi-class problem into several binary classification problems ("one-against-all"), then calibrating scores of those binary classifiers by another SVM model ("one-against-all + SVM").

method	Weizmann		KTH	
	per-frame	per-video	per-frame	per-video
no pairwise				
$\|\mathcal{H}\| = 6$	0.8344	0.9062	0.6767	0.8527
$\|\mathcal{H}\| = 10$	0.8414	0.9688	0.7005	0.8941
$\|\mathcal{H}\| = 20$	0.8358	0.9688	0.6891	0.8734
one-against-all				
$\|\mathcal{H}\| = 6$	0.7525	0.8889	0.5171	0.6589
$\|\mathcal{H}\| = 10$	0.7507	0.8611	0.5052	0.6589
$\|\mathcal{H}\| = 20$	0.7447	0.8889	0.5052	0.6899
one-against-all + SVM				
$\|\mathcal{H}\| = 6$	0.8173	0.9444	0.5705	0.7209
$\|\mathcal{H}\| = 10$	0.8460	0.9444	0.5610	0.7287
$\|\mathcal{H}\| = 20$	0.8145	0.9444	0.5623	0.7442

tion of LSVMs in [3]. In the end, we get K binary LSVM models. For a test image, we assign its class label by choosing the corresponding LSVM model which gives the highest score on this image. Similarly, we perform per-frame classification and use majority voting to obtain per-video classification. The results are shown in Table 5 ("one-against-all"). We can see that the one-against-all strategy does not perform as well as our proposed method (see Table 2). We believe it is because those binary classifiers are trained independently of each other, and their scores are not calibrated to be comparable. To test this hypothesis, we perform another method (called "one-against-all + SVM") in Table 5. We take the output scores of K binary LSVM models and stack them into a K-dimensional vector. Then we train a multi-class SVM that takes this vector as its input and produces one of the K class labels. This multi-class SVM will calibrate the scores of different binary LSVM models. The results show that this calibration step significantly improves the performance of the one-against-all strategy. But the final results are still worse than MMHCRFs. So in summary, our experimental results show that the one-against-all strategy does not perform as well as MMHCRFs.

7 Conclusion

We have presented discriminatively learned part models for human action recognition. Our model combines both large-scale features used in global templates and local patch features used in bag-of-words models. Our experimental results show that our model is quite effective in recognizing actions. The results are comparable to the state-of-the-art approaches. In particular, we show that the combination of large-scale features and local patch features performs significantly better than using either of them alone. We also presented a new max-margin learning method for learning the model parameter. Our experimental results show that the max-margin learning outperforms the probabilistic learning based on maximizing conditional the log-likelihood of training data. More importantly, the max-margin learning allows us to deal with a large spectrum of different complex structures.

We have also given a detailed theoretical analysis of pros and cons of HCRFs and MMHCRFs in terms of both computational and modeling perspectives. Our analysis explains why the design choices of MMHCRFs (max-margin instead of maximizing likelihood, maximizing over $\mathbf{h}$ instead of summing over $\mathbf{h}$) lead to better performance in our experiments.

The applicability of the proposed models and learning algorithms goes beyond human action recognition. In computer vision and in many other fields, we are constantly dealing with data with rich, interdependent structures. We believe our proposed work will pave a new avenue to construct more powerful models for handling various forms of latent structures.

Our proposed models are still very simple. As future work, we would like to extend our models to handle things like partial occlusion, viewpoint change, etc. We also plan to investigate more robust features. These extensions will be necessary to make our models applicable on more challenging videos.

References

[1] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *IJCV*, vol. 42, no. 3, pp. 145–175, 2001.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[3] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *CVPR*, 2008.

[4] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell, "Hidden conditional random fields," *IEEE PAMI*, vol. 29, no. 10, pp. 1848–1852, June 2007.

[5] N. Dalal and B. Triggs, "Histogram of oriented gradients for human detection," in *CVPR*, 2005.

[6] A. C. Berg, T. L. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondence," in *CVPR*, 2005.

[7] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images," in *ICCV*, vol. 1, 2005, pp. 370–377.

[8] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *IJCV*, vol. 61, no. 1, pp. 55–79, January 2005.

[9] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *ICCV*, 2005.

[10] A. A. Efros, A. C. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *ICCV*, 2003, pp. 726–733.
[11] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *VS-PETS*, 2005.
[12] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," in *BMVC*, 2006.
[13] S. Nowozin, G. Bakir, and K. Tsuda, "Discriminative subsequence mining for action classification," in *ICCV*, 2007.
[14] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *ICPR*, vol. 3, 2004, pp. 32–36.
[15] Y. Wang and G. Mori, "Learning a discriminative hidden part model for human action recognition," in *NIPS*, 2009.
[16] ——, "Max-margin hidden conditional random fields for human action recognition," in *CVPR*, 2009.
[17] R. Cutler and L. S. Davis, "Robust real-time periodic motion detection, analysis, and applications," *IEEE PAMI*, vol. 22, no. 8, pp. 781–796, 2000.
[18] R. Polana and R. C. Nelson, "Detection and recognition of periodic, non-rigid motion," *IJCV*, vol. 23, no. 3, pp. 261–282, June 1997.
[19] J. L. Little and J. E. Boyd, "Recognizing people by their gait: The shape of motion," *Videre*, vol. 1, no. 2, pp. 1–32, 1998.
[20] C. Rao, A. Yilmaz, and M. Shah, "View-invariant representation and recognition of actions," *IJCV*, vol. 50, no. 2, pp. 203–226, 2002.
[21] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE PAMI*, vol. 23, no. 3, pp. 257–267, 2001.
[22] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *ICCV*, 2007.
[23] I. Laptev and T. Lindeberg, "Space-time interest points," in *ICCV*, 2003.
[24] J. C. Niebles and L. Fei-Fei, "A hierarchical model of shape and appearance for human action classification," in *CVPR*, 2007.
[25] R. Fergus, P. Perona, and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," in *CVPR*, 2003.
[26] Y. Ke, R. Sukthankar, and M. Hebert, "Event detection in crowded videos," in *ICCV*, 2007.
[27] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001.
[28] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *NIPS*, 2004.
[29] B. Taskar, S. Lacoste-Julien, and M. I. Jordan, "Structured prediction, dual extragradient and Bregman projections," *JMLR*, vol. 7, pp. 1627–1653, 2006.
[30] Y. Altun, T. Hofmann, and I. Tsochantaridis, "SVM learning for interdependent and structured output spaces," in *Machine Learning with Structured Outputs*, G. Bakir, T. Hofman, B. Scholkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, Eds. MIT Press, 2006.
[31] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *DARPA Image Understanding Workshop*, 1981.
[32] C. Desai, D. Ramanan, and C. Fowlkes, "Discriminative models for multi-class object layout," in *ICCV*, 2009.
[33] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *JMLR*, vol. 2, pp. 265–292, 2001.
[34] Y. LeCun, S. Schopra, R. Radsell, R. Marc'Aurelio, and F. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, T. Hofman, B. Scholkopf, A. Smola, and B. Taskar, Eds. MIT Press, 2006.
[35] S. Boyd and L. Vanderghe, *Convex Optimization*. Cambridge University Press, 2004.
[36] M. Szummer, P. Kohli, and D. Hoiem, "Learning CRFs using graph cuts," in *ECCV*, 2008.
[37] J. C. Platt, "Using analytic QP and sparseness to speed training of support vector machines," in *NIPS*, 1999.
[38] M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett, "Exponentiated gradient algorithms for conditional random fields and max-margin markov networks," *JMLR*, vol. 9, pp. 1757–1774, August 2008.
[39] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, "MAP estimation via agreement on trees: Message-passing and linear programming," *IEEE Trans. on Information Theory*, vol. 51, no. 11, pp. 3697–3717, November 2005.
[40] D. Tran and D. Forsyth, "Configuration estimates improve pedestrian finding," in *NIPS*, 2008.
[41] S. Belongie, G. Mori, and J. Malik, "Matching with shape contexts," in *Analysis and Statistics of Shapes*. Birkhäuser, 2005.
[42] C. Cherry and C. Quirk, "Discriminative, syntactic language modeling through latent SVMs," in *AMTA*, 2008.
[43] C.-N. Yu and T. Joachims, "Learning structural SVMs with latent variables," in *ICML*, 2009.
[44] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Machine Learning*, vol. 42, pp. 177–196, 2001.
[45] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *JMLR*, vol. 3, pp. 993–1022, 2003.
[46] P. Viola, J. C. Platt, and C. Zhang, "Multiple instance boosting for object recognition," in *NIPS*, 2006.
[47] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *NIPS*, 2002.
[48] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts," *IEEE PAMI*, vol. 26, no. 2, pp. 147–159, February 2004.
[49] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*. Berlin: Springer, 2003.
[50] J. Liu and M. Shah, "Learning human actions via information maximization," in *CVPR*, 2008.
[51] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *ICCV*, 2005.

PLACE PHOTO HERE

Yang Wang is currently an NSERC postdoctoral fellow at the Department of Computer Science, University of Illinois at Urbana-Champaign. He received his Ph.D. from Simon Fraser University (Canada), his M.Sc. from University of Alberta (Canada), and his B.Sc. from Harbin Institute of Technology (China), all in computer science. He was a research intern at Microsoft Research Cambridge in summer 2006. His research interests lie in high-level recognition problems in computer vision, in particular, human activity recognition, human pose estimation, object/scene recognition, etc. He also works on various topics in statistical machine learning, including structured prediction, probabilistic graphical models, semi-supervised learning, etc.

PLACE PHOTO HERE

Greg Mori was born in Vancouver and grew up in Richmond, BC. He received the Ph.D. degree in Computer Science from the University of California, Berkeley in 2004. He received an Hon. B.Sc. in Computer Science and Mathematics with High Distinction from the University of Toronto in 1999. He spent one year (1997-1998) as an intern at Advanced Telecommunications Research (ATR) in Kyoto, Japan. After graduating from Berkeley, he returned home to Vancouver and is currently an assistant professor in the School of Computing Science at Simon Fraser University. Dr. Mori's research interests are in computer vision, and include object recognition, human activity recognition, human body pose estimation. He serves on the program committee of major computer vision conferences (CVPR, ECCV, ICCV), and was the program co-chair of the Canadian Conference on Computer and Robot Vision (CRV) in 2006 and 2007. Dr. Mori received the Excellence in Undergraduate Teaching Award from the SFU Computing Science Student Society in 2006. Dr. Mori received the Canadian Image Processing and Pattern Recognition Society (CIPPRS) Award for Research Excellence and Service in 2008. Dr. Mori received an NSERC Discovery Accelerator Supplement in 2008.