Max-Margin Hidden Conditional Random Fields for Human Action Recognition

Yang Wang
School of Computing Science
Simon Fraser University, Canada
ywang12@cs.sfu.ca

Greg Mori
School of Computing Science
Simon Fraser University, Canada
mori@cs.sfu.ca

Abstract

We present a new method for classification with structured latent variables. Our model is formulated using the max-margin formalism in the discriminative learning literature. We propose an efficient learning algorithm based on the cutting plane method and decomposed dual optimization. We apply our model to the problem of recognizing human actions from video sequences, where we model a human action as a global root template and a constellation of several "parts". We show that our model outperforms another similar method that uses hidden conditional random fields, and is comparable to other state-of-the-art approaches. More importantly, our proposed work is quite general and can potentially be applied in a wide variety of vision problems that involve various complex, interdependent latent structures.

1. Introduction

Many problems in computer vision can be formulated as *classification with structured latent variables*. 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 -1(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 [11, 30] usually use a tree-structured model to model these constraints. (2) *Object recognition*: this problem is to assign a class label to an image that 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 [2]. 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].

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 [8] in the case of pedestrian detection. However, there is evidence [11, 24, 30, 32] showing that incorporating latent structures into the system can improve the performance.

In this paper, we propose a max-margin learning framework for training classifiers with structured latent variables. We introduce an efficient learning algorithm based on the cutting plane method and decomposed dual optimization. We apply our proposed model to recognize human actions from video sequences, where we model a person by a root template and a constellation of several "parts". Our model is closely related to a recent work [32] on part-based human action recognition using hidden conditional random fields (HCRF) [24]. The main difference between our approach and the approach in [32] is that our approach uses a max-margin criterion for training. We call our approach the *Max-Margin Hidden Conditional Random Field (MMHCRF)*. We show experimentally that the MMHCRF achieves better performance than the HCRF. More importantly, the max-margin learning allows us to deal with more complex latent structures (e.g., matching/correspondence mentioned before).

This paper represents our first step towards solving the general problem of classification with structured latent variables using the max-margin framework. Here we only consider latent structures that form a tree-structured model, and we restrict ourselves to the application domain of human action recognition. But the proposed framework is quite general and can be applied in other domains that involve highly

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

complex latent structures.

2. Previous Work

The problem of action recognition has been approached in many ways in the literature. A lot of approaches use interest-points and bag-of-words representations [9, 17, 22]. There are also methods based on global templates [10]. Inspired by part-based models for object recognition (e.g.,[11, 12]), recent work has applied similar part-based models for actions [14, 21, 32].

Our work is closely related to the hidden conditional random field (HCRF) model [24]. The HCRF was originally proposed for object recognition. The basic idea is to model an object as a constellation of "parts" conditioned on local observations found by an interest point operator. For each object class, the probability of assigning part labels to local features is modeled by a conditional random field (CRF) [16]. The parameters of a HCRF model are learned by maximizing the conditional likelihood of the training data. HCRFs have also been applied to part-based human action recognition [32]. The limitation of HCRFs is that the training involves summing over all the possible labelings of the latent variables. If the latent variables form certain graph structures (e.g. trees, graphs of low tree-width), this summation can be done analytically. For latent structures of general graph topology, HCRFs have to resort to various approximations (e.g. loopy belief propagation, mean-field variational methods, etc.) for training. For even more complicated structures (e.g. matching/correspondence), it is not even clear how to approximate the computation. Compared with HCRFs, our model can handle a much wider range of latent structures. This is in analog to the advantage of the max-margin Markov network (M^3N) [28, 29] over the CRF in the structured-output learning literature.

Another closely related work is the latent SVM (LSVM) [11]. The LSVM is proposed for object detection (binary classification). It models an object as a global coarse template covering an entire object and several higher resolution part templates. The positions of the root template and the parts are latent in LSVMs. LSVMs learn the model parameters using a max-margin discriminative training method. Our work is different from LSVMs in several directions. First of all, our model directly handles multi-class classification. Unfortunately, unlike LSVMs, the resultant optimization problem is not in a standard form that can be solved by off-the-shelf SVM solvers (e.g., SVMLight). We propose an efficient algorithm based on the cutting-plane method and the decomposed dual optimization. Our learning algorithm is intuitive and easy to implement. It is also quite general – one only needs to change a small component of the algorithm in order to handle different latent structures. The potential functions and parametrization of our model are also significantly different from those used in LSVMs. In our model, a hidden variable corresponds to a "part label", while in LSVMs, a hidden variable corresponds to the location of a part. The parts in our model can have a variety of constraints, while in the model used in LSVMs, the parts only have constraints with respect to a root filter.

3. Model

Our approach uses motion features based on optical flows (Sec. 3.1). A frame in a video is represented by a global motion feature extracted from the whole frame and a set of salient local patches. Our model consists of a root filter and a constellation of several hidden parts. The root filter models the compatibility of the action label and the global motion feature of the whole frame. A hidden part assigns a latent "part label" to a local patch. Intuitively, those "part labels" correspond to local motion patterns that are useful for discriminating different actions. Figure 1 shows some sample frames and the learned hidden part labels. The constraints among the patches are captured by pairwise potential functions defined in the model (Sec. 3.2).

3.1. Motion Feature

Similar to [32], we use the optical flow feature in Efros et al. [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 people in a video sequence. Any tracking or human detection algorithm can be used, since the motion descriptor 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 of each frame using the Lucas-Kanade [20] 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^-.

3.2. Hidden Part Model

Let I be a frame in a video sequence. We assume I contains a set of salient patches $\{I_1, I_2, ..., I_m\}$. Let x be the feature vector extracted from the frame I, and $y \in \mathcal{Y}$ be a class label, with discrete domain $\mathcal{Y}$. We assume x has the form of $x = (s_0, s_1, ..., s_m)$, where s_0 is the motion feature vector extracted from the whole frame, and

$s_i (i = 1, 2, ..., m)$ is the motion feature vector extracted from the patch I_i. We assume I is associated with a vector of hidden variables of the form $h = (z_1, z_2, ..., z_m)$, where z_i takes values from a discrete set $\mathcal{H}$ of possible "part" labels. Intuitively, z_i assigns a "part label" to the patch I_i, where a "part label" corresponds to certain motion patterns distinctive of certain actions. For example, one "part label" might corresponds to the "moving down" pattern commonly observed in "bending" action, another "part label" might correspond to the motion pattern distinctive of "hand-waving" action. See Figure 1 for examples of learned part labels. In our model, there are certain constraints between some pairs of (z_j, z_k). We use an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to represent $(z_1, z_2, ..., z_m)$, where a vertex $v_i \in \mathcal{V}$ corresponds to the part label z_i, and an edge $(v_j, v_k) \in \mathcal{E}$ corresponds to the constraint between z_j and z_k. Similar to HCRFs [24, 32], $\mathcal{E}$ is assumed to form a tree. The tree structure is obtained by running a minimum spanning tree algorithm over the salient patches.

The $\langle x, y \rangle$ pair is scored by a function of the form $f_w(x, y) = \max_h w^\top \Phi(x, h, y)$. Here w is a vector of model parameters and h are the latent variables. The example x is classified according to $y^* = \arg\max_y f_w(x, y)$. The model parameters w have four components $w = \{w_0, w_1, w_2, w_3\}$, and $w^\top \Phi(x, h, y)$ is defined as:

$$w^\top \Phi(x, h, y) = w_0^\top \cdot \phi_0(y, s_0) + \sum_{j \in \mathcal{V}} w_1^\top \cdot \phi_1(s_j, z_j) + \sum_{j \in \mathcal{V}} w_2^\top \cdot \phi_2(y, z_j) + \sum_{(j,k) \in \mathcal{E}} w_3^\top \cdot \phi_3(y, z_j, z_k) \quad (1)$$

The forms of these potential functions and the parametrization are identical to those in [32]. $w_0^\top \phi_0(y, s_0)$ is a root filter that models the compatibility of the class label y and the large-scale global feature of the whole image. It is parametrized as $w_0^\top \phi_0(y, s_0) = \sum_{a \in \mathcal{Y}} w_{0a}^\top \cdot \mathbb{1}(y = a) \cdot g(s_0)$, where $g(s_0)$ is the concatenation of the motion features Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^- from the patch s_0 (i.e., the whole image), w_{0a} can be interpreted as a root filter that measures the compatibility of the class label a and the motion vector $g(s_0)$, w_0 is a simply the concatenation of w_{0a} for $\forall a \in \mathcal{Y}$. Similarly, $w_1^\top \phi_1(s_j, z_j) = \sum_{c \in \mathcal{H}} w_{1c}^\top \cdot \mathbb{1}(z_j = c) \cdot g(s_j)$ measures the compatibility of the feature vector $g(s_j)$ extracted from the j-th patch and a hidden part label z_j for the j-th patch. $w_2^\top \phi_2(y, z_j) = \sum_{a \in \mathcal{Y}} \sum_{b \in \mathcal{H}} w_{2ab} \cdot \mathbb{1}(y = a) \cdot \mathbb{1}(z_j = b)$ measures the compatibility of the class label y and a hidden part label z_j for the j-the patch. $w_3^\top \phi_3(y, z_j, z_k) = \sum_{a \in \mathcal{Y}} \sum_{b \in \mathcal{H}} \sum_{c \in \mathcal{H}} w_{3abc} \cdot \mathbb{1}(y = a) \cdot \mathbb{1}(z_j = b) \cdot \mathbb{1}(z_k = c)$ measures the compatibility of a class label y and a pair of hidden part labels (z_j, z_k) for a pair of (j, k)-th patches. These constraints defined on edges capture the intuition that patches with certain hidden part labels tend to appear together for certain actions.

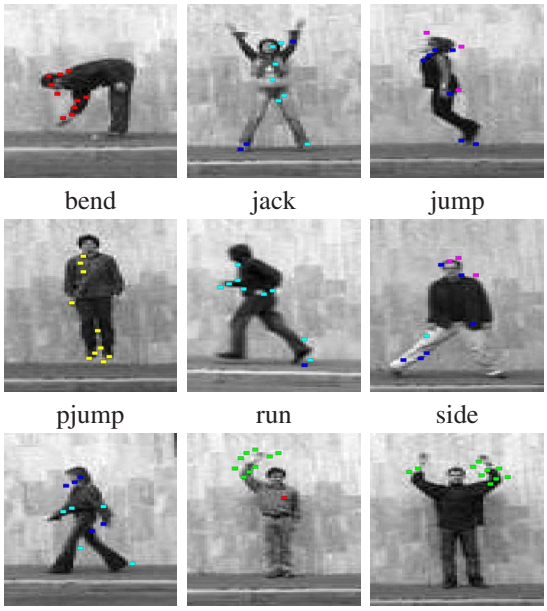


Figure 1. Visualization of the learned parts. Patches are colored according to their most likely part labels. Interesting observations can be made. For example, the parts colored in red, yellow and green seem to be distinctive of "bend" "pjump" and "wave" actions, respectively. We can also observe the "sharing of parts" among different actions. For example, the parts colored in blue and light blue are shared by "walk" "run" "jack" actions.

4. Learning

Our learning method is inspired by the success of max-margin methods in machine learning [1, 7, 11, 28]. 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.

4.1. Max-Margin HCRF

A max-margin hidden conditional random field (MMHCRF) is defined as follows. Recall that an $\langle x, y \rangle$ pair is scored by a function of the form:

$$f_w(x, y) = \max_h w^\top \Phi(x, h, y) \quad (2)$$

where w is the model parameter and h is a vector of hidden variables. In this paper, we consider the case in which $h = (z_1, z_2, ..., z_m)$ 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 [18].

Let $D = (\langle x_1, y_1\rangle, \langle x_2, y_2\rangle, ..., \langle x_N, y_N\rangle)$ be a set of labeled training examples. The goal of learning is to learn the model parameter w, so that for a new example x, we can classify x to be class y^* if $y^* = \arg\max_y f(x, y)$.

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

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & f_w(x_i, y) - f_w(x_i, y_i) \le \xi_i - 1, \ \ \forall i, \ \ \forall y \neq y_i \\ & \xi_i \ge 0, \qquad \forall i \end{aligned} \tag{3}$$

where C is the trade-off parameter similar to that in SVMs, and ξ_i is the slack variable for the i-the training example to handle the case of soft margin.

The optimization problem in (3) is equivalent to the following optimization problem:

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \max_h w^\top \Phi(x_i, h, y) - \max_{h'} w^\top \Phi(x_i, h', y_i) \\ & \le \xi_i - \delta(y, y_i), \ \ \forall i, \ \ \forall y \\ \text{where} \quad & \delta(y, y_i) = \begin{cases} 1 & \text{if } y \neq y_i \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4}$$

4.2. Semi-Convexity and Primal Optimization

Similar to LSVMs, MMHCRFs have the property of semi-convexity. Note that $f_w(x, y)$ is a maximum of a set of functions, each of which is linear in w, so $f_w(x)$ is convex in w. If we restrict the domain of h' in (4) to a single choice, the optimization problem of (4) becomes convex [4]. This is in analog to restricting the domain of the latent variables for the positive examples to a single choice in LSVMs [11]. But here we are dealing with multi-class classification, our "positive examples" are those $\langle x_i, y\rangle$ pairs where $y = y_i$.

We can compute a local optimum of (4) using a coordinate descent algorithm similar to LSVMs [11]:

1. Holding w, ξ fixed, optimize the latent variables h' for the $\langle x_i, y_i\rangle$ pair:

$$h_{i,y_i} = \arg\max_{h'} w^\top \Phi(x_i, h', y_i)$$

2. Holding h_{i,y_i} fixed, optimize w, ξ by solving the following optimization problem:

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \max_h w^\top \Phi(x_i, h, y) - w^\top \Phi(x_i, h_{i,y_i}, y_i) \\ & \le \xi_i - \delta(y, y_i), \ \ \forall i, \ \ \forall y \end{aligned} \tag{5}$$

It can be shown that both steps always improve or maintain the objective [11].

The optimization problem in Step 1 can be solved efficiently for certain structures of 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 [4], 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 (5) into a standard quadratic program with only linear constraints.

One possible way to convert (5) into a standard QP is to solve the following convex optimization problem:

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & w^\top \Phi(x_i, h, y) - w^\top \Phi(x_i, h_{i,y_i}, y_i) \\ & \le \xi_i - \delta(y, y_i), \qquad \forall i, \forall h, \forall y \end{aligned} \tag{6}$$

It is easy to see that (5) and (6) are equivalent, and all the constraints in (6) are linear. Unfortunately, the optimization problem in (6) involves an exponential number of constraints – for each example x_i and each possible labeling y, there are exponentially many possible 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 [1] and CRFs [27]. In a nutshell, the algorithm starts with no constraints (which corresponds to a relaxed version of (6)), 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 (6) by evaluating only a polynomial number of constraints.

More importantly, the optimization problem in (6) has certain properties that allow us to find and add constraints in an efficient way. For a fixed example x_i and a possible label y, define $h_{i,y}$ as follows:

$$h_{i,y} = \arg\max_h w^\top \Phi(x_i, h, y)$$

Consider the following two set of constraints for the $\langle x_i, y\rangle$ pair:

$$\begin{aligned} & w^\top \Phi(x_i, h_{i,y}, y) - w^\top \Phi(x_i, h_{i,y_i}, y_i) \\ & \le \xi_i - \delta(y, y_i) \end{aligned} \tag{7}$$

$$\begin{aligned} & w^\top \Phi(x_i, h, y) - w^\top \Phi(x_i, h_{i,y_i}, y_i) \\ & \le \xi_i - \delta(y, y_i), \quad \forall h \end{aligned} \tag{8}$$

It is easy to see that within a local neighborhood of w, (7) and (8) define the same set of constraints, i.e., (7) implies (8) and vice versa. This suggests that for a fixed $\langle x_i, y\rangle$ pair, we only need to consider the constraint involving $h_{i,y}$.

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

1. Fixing w, ξ, optimize the latent variable h for each pair $\langle x_i, y\rangle$ of an example x_i and a possible labeling y:

$$h_{i,y} = \arg\max_h w^\top \Phi(x_i, h, y)$$

2. Fixing $h_{i,y} \quad \forall i, \ \forall y$, optimize w, ξ by solving the following optimization problem:

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & w^\top \Phi(x_i, h_{i,y}, y) - w^\top \Phi(x_i, h_{i,y_i}, y_i) \\ & \le \xi_i - \delta(y, y_i), \ \ \forall i, \ \ \forall y \end{aligned} \tag{9}$$

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

The optimization in (9) is very similar to the primal problem of a standard multi-class SVM [7]. In fact, if $h_{i,y}$ 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 (9). Unfortunately, the fact that $h_{i,y}$ 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 (9) by examining its dual. To simplify the notation, let us define $\Psi(x_i, y) = \Phi(x_i, h_{i,y}, y) - \Phi(x_i, h_{i,y_i}, y_i)$. Then the dual problem of (9) can be written as follows:

$$\begin{aligned} \max_\alpha \quad & \sum_i \sum_y \alpha_{i,y}\delta(y, y_i) - \frac{1}{2}||\sum_i \sum_y \alpha_{i,y}\Psi(x_i, y)||^2 \\ \text{s.t.} \quad & \sum_y \alpha_{i,y} = C, \quad \forall i \\ & \alpha_{i,y} \ge 0, \quad \forall i, \quad \forall y \end{aligned} \tag{10}$$

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

$$w = -\sum_i \sum_y \alpha_{i,y}\Psi(x_i, y)$$

Note that (10) is quite similar to the dual form of standard multi-class SVMs. In fact, if $h_{i,y}$ is a deterministic function of x_i, (10) 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:

$$K(i, y; j, y') = \Psi(x_i, y)^\top \Psi(x_j, y')$$

Let us define α as the concatenation of $\{\alpha_{i,y} : \forall i \ \ \forall y\}$, so the length of α is $N \times |\mathcal{Y}|$. Define Δ as a vector of the same length. The (i, y)-th entry of Δ is 1 if $y \ne y_i$, and 0 otherwise. Then (10) can be written as:

$$\begin{aligned} \max_\alpha \quad & \alpha^\top \Delta - \frac{1}{2}\alpha^\top K \alpha \\ \text{s.t.} \quad & \sum_y \alpha_{i,y} = C, \qquad \forall i \\ & \alpha_{i,y} \ge 0, \qquad \forall i, \ \ \forall y \end{aligned} \tag{11}$$

Note the matrix K in (11) only depends on the dot-product between feature vectors of different $\langle x_i, y\rangle$ 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 (10) 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 (10) and solve a serie of smaller QPs. This is similar to the sequential minimal optimization (SMO) used in SVM [7, 23] and M^{3}N [28]. The basic idea of this algorithm is to choose all the $\{\alpha_{i,y} : \forall y \in \mathcal{Y}\}$ for a particular training example x_i and fix all the other variables $\{\alpha_{k,y'} : \forall k : k \ne i, \forall y' \in \mathcal{Y}\}$ that do not involve x_i. Then instead of solving a QP involving all the variables $\{\alpha_{i,y} : \forall i, \forall y\}$, we can solve a much smaller QP only involving $\{\alpha_{i,y} : \forall y\}$. 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 (10) in terms of $\{\alpha_{i,y} : \forall y\}$ as follows:

$$\begin{aligned} & \mathcal{L}(\{\alpha_{i,y} : \forall y\}) \\ = & \sum_y \alpha_{i,y}\delta(y, y_i) - \frac{1}{2}\Big[||\sum_y \alpha_{i,y}\Psi(x_i, y)||^2 \\ & + 2\Big(\sum_y \alpha_{i,y}\Psi(x_i, y)\Big)^\top \Big(\sum_{k:k\ne i}\sum_{y'} \alpha_{k,y'}\Psi(x_k, y')\Big)\Big] \\ & + \text{other terms not involving } \{\alpha_{i,y} : \forall y\} \end{aligned}$$

The smaller QP corresponding to $\langle x_i, y_i\rangle$ can be written as follows:

$$\begin{aligned} \max_{\alpha_{i,y}:\forall y} \quad & \mathcal{L}(\{\alpha_{i,y} : \forall y\}) \\ \text{s.t.} \quad & \sum_y \alpha_{i,y} = C \\ & \alpha_{i,y} \ge 0, \quad \forall y \end{aligned} \tag{12}$$

Note $\sum_{k:k\ne i}\sum_{y'} \alpha_{k,y'}\Psi(x_k, y')$ can be written as:

$$-w - \sum_y \alpha_{i,y}\Psi(x_i, y)$$

So as long as we maintain (and keep updating) the global parameter w and keep track of $\alpha_{i,y}$ and $\Psi(x_i, y)$ for each example $\langle x_i, y_i \rangle$, we do not need to actually do the summation $\sum_{k:k\neq i} \sum_{y'}$ when optimizing (12). In addition, when we solve the QP involving $\alpha_{i,y}$ for a fixed i, all the other constraints involving $\alpha_{k,y}$ where $k \neq i$ are not affected. This is not the case if we try to optimize the primal problem in (9). If we try to optimize the primal variable w by only considering the constraints involving the i-th examples, it is possible that the new w obtained from the optimization might violate the constraints imposed by other examples. There is also work [6] showing that the dual optimization has a better convergence rate.

4.4. Finding the Optimal h

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

$$h^* = \arg\max_h w^\top \Phi(x, h, y) \tag{13}$$

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 $h = (z_1, z_2, ..., z_m)$ forms a tree-structured graphical model, the inference problem in (13) 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, 31]. We introduce variables $\mu_{j\mathfrak{a}}$ to denote the indicator $\mathbb{1}(z_j = \mathfrak{a})$ 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 $\mathbb{1}(z_j = \mathfrak{a}, z_k = \mathfrak{b})$ 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 $\psi_j(z_j)$ to collectively represent the summation of all the unary potential functions in (1) that involve the node $j \in \mathcal{V}$. We use $\psi_{jk}(z_j, z_k)$ to collectively represent the summation of all the pairwise potential functions in (1) that involve the edge $jk \in \mathcal{E}$. The problem of finding of optimal h^* can be formulated into the following linear programming (LP) problem:

$$\begin{aligned}
&\max_{0\leq\mu\leq 1} \sum_{j\in\mathcal{V}} \sum_{\mathfrak{a}\in\mathcal{H}} \mu_{j\mathfrak{a}} \psi_j(\mathfrak{a}) + \sum_{jk\in\mathcal{E}} \sum_{\mathfrak{a}\in\mathcal{H}} \sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} \psi_{jk}(\mathfrak{a}, \mathfrak{b}) \\
&\text{s.t.} \quad \sum_{\mathfrak{a}\in\mathcal{H}} \mu_{j\mathfrak{a}} = 1, \quad \forall j \in \mathcal{V} \\
&\qquad \sum_{\mathfrak{a}\in\mathcal{H}} \sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = 1, \quad \forall jk \in \mathcal{E} \\
&\qquad \sum_{\mathfrak{a}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = \mu_{k\mathfrak{b}}, \quad \forall jk \in \mathcal{E}, \quad \forall \mathfrak{b} \in \mathcal{H} \\
&\qquad \sum_{\mathfrak{b}\in\mathcal{H}} \mu_{jk\mathfrak{ab}} = \mu_{j\mathfrak{a}}, \quad \forall jk \in \mathcal{E}, \quad \forall \mathfrak{a} \in \mathcal{H}
\end{aligned} \tag{14}$$

If the optimal solution of this LP is integral, we can recover 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, 31]. For general graph topology, the optimal solution of this LP can be fractional, which is not surprising, since the problem in (13) 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]).

4.5. Comparison with HCRF

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 h, versus summing over h.

From the computational perspective, if 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 [15]. MMHCRFs can use LP [29] or graph-cut [15] 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. The inference of this structure can be solved by MMHCRFs using LP [25, 29]. 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 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 [12] can be applied in MMHCRFs to solve the inference problem. This is essentially what has been done in [11]. This inference problem for HCRFs can be solved using convolution. But distance transform ($O(|\mathcal{H}|)$) is more efficient than convolution ($O(|\mathcal{H}| \log |\mathcal{H}|)$).

Ideas similar to MMHCRFs have also been applied to model complex hidden structures arising in other fields, e.g., parse tree structures in natural language processing [5], and motif positions in bioinformatics [33].

From the modeling perspective, we believe MMHCRFs are better suited for classification than HCRFs. This is because HCRFs require summing over exponentially many 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" labelings of 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 than that of HCRFs. A more rigorous analysis of these two different learning criteria will be an interesting and important future work.

5. Experiments

We test our algorithm on two widely used benchmark datasets: the Weizmann human action dataset [3] and the KTH human motion dataset [26]. Performance on these benchmarks is saturating – state-of-the-art approaches achieve near-perfect results. We show our method outperforms the HCRF model [32] and is comparable to other state-of-the-art approaches. In order to do a fair comparison with [32], we use the same split of training/testing data. We also use the same patch initialization technique in [32] to find ten salient patches in each image. We set $C = 1$ for MMHCRF in all the experiments.

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. Similar to [32], we learn our model with different sizes of possible part labels, $|\mathcal{H}| = 6, 10, 20$. During testing, our model performs per-frame classification, then obtains the class label for the video by majority voting.

The comparison with the HCRF model [32] is shown in Table 1. Our model outperforms HCRF in all three cases. The comparison with other approaches is summarized in Table 2. We should emphasize that we do not attempt a direct comparison, since Niebles & Fei-Fei [21] does not require stabilization, and Blank et.al [3] uses a different measurement (per-cube accuracy). We show the confusion matrix of our approach with $|\mathcal{H}| = 10$ for per-frame classification in Fig. 2(a). The confusion matrix for per-video classification is omitted, since it is just a perfect diagonal matrix.

KTH dataset: The KTH human motion dataset contains six types of human actions: walking, jogging, running, boxing, hand waving and hand clapping. Again, we learn the model with $|\mathcal{H}| = 6, 10, 20$.

method	$\|\mathcal{H}\|$=6	$\|\mathcal{H}\|$=10	$\|\mathcal{H}\|$=20
HCRF	0.8682 0.9167	0.9029 0.9722	0.8557 0.9444
Our approach	**0.8996** **0.9722**	**0.9311** **1.0000**	**0.8891** **0.9722**

Table 1. Comparison of our approach with the HCRF model [32] on the Weizmann dataset. The first number in each cell is the accuracy of per-frame classification. The second number is the accuracy of per-video classification.

method	per-frame	per-video	per-cube
Our method	**0.9311**	**1.0000**	N/A
Wang & Mori [32]	0.9029	0.9722	N/A
Jhuang et al. [13]	N/A	0.9880	N/A
Niebles & Fei-Fei [21]	0.5500	0.7280	N/A
Blank et al. [3]	N/A	N/A	0.9964

Table 2. Comparison of classification accuracy with previous work on the Weizmann dataset.

method	$\|\mathcal{H}\|$=6	$\|\mathcal{H}\|$=10	$\|\mathcal{H}\|$=20
HCRF	0.6633 0.7855	0.6698 0.8760	0.6444 0.7512
Our approach	**0.7064** **0.8475**	**0.7853** **0.9251**	**0.7486** **0.8966**

Table 3. Comparison of our approach with the HCRF model on the KTH dataset. The first number in each cell is the accuracy of per-frame classification. The second number is the accuracy of per-video classification.

method	**accuracy**
Our method	**0.9251**
Liu & Shah [19]	**0.9416**
Jhuang et al. [13]	0.9170
Wang & Mori [32]	0.8760
Niebles et al. [22]	0.8150
Dollár et al. [9]	0.8117
Schuldt et al. [26]	0.7172

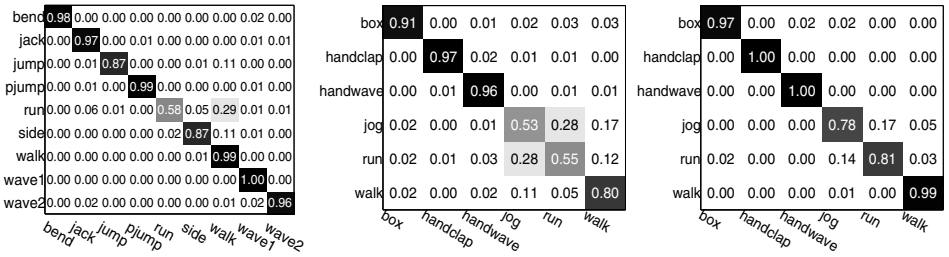
Table 4. Comparison of per-video classification accuracy with previous approaches on the KTH dataset.

The confusion matrices of our approach (with $|\mathcal{H}| = 10$) for both per-frame and per-video classification are shown in Fig. 2(b,c). The comparison with the HCRF model is summarized in Table 3. Again, our approach outperforms the HCRF model.

The comparison with other approaches is summarized in Table 4. Again, we would like to emphasize that this is not a direct comparison, due to all sorts of variations of the experiment setups in different methods listed in Table 4. We only show the results to demonstrate that our approach is comparable to other state-of-the-art methods.

6. Conclusion

We have presented the max-margin hidden conditional random field – a learning framework for training multi-



(a) Weizmann: per-frame (b) KTH: per-frame (c) KTH: per-video

Figure 2. Confusion matrices of classification results on Weizmann and KTH dataset. The confusion matrix of per-video classification on the Weizmann dataset is not shown, since it is simply a perfect diagonal matrix.

class classifiers with structured latent variables. We propose an efficient learning algorithm based on the cutting plane method and decomposed dual optimization. Our proposed model is quite general and can be applied in a wide range of latent structures, as long as we have an algorithm for solving the inference in Sec. 4.4. For a lot of vision applications, this inference is easier than computing the summation over all the latent variables, which is required by HCRF training. Our experimental results show that the max-margin learning achieves better performance than the conditional likelihood learning, which is consistent with similar findings in the structured output learning literature [1, 28, 29].

As future work, we would like to apply our model in various vision problems that involve latent structures, and to derive efficient approximation algorithms for general graph structures, of which exact inference is intractable. We also plan to further investigate the theoretical differences between the max-margin and the conditional likelihood learning criteria.

References

[1] Y. Altun, T. Hofmann, and I. Tsochantaridis. SVM learning for interdependent and structured output spaces. In *Machine Learning with Structured Outputs*. MIT Press, 2006.

[2] S. Belongie, G. Mori, and J. Malik. Matching with shape contexts. In *Analysis and Statistics of Shapes*. Birkhäuser, 2005.

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

[4] S. Boyd and L. Vanderghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] C. Cherry and C. Quirk. Discriminative, syntactic language modeling through latent SVMs. In *AMTA*, 2008.

[6] M. Collins, A. Globerson, T. Koo, X. Carrera, and P. Bartlett Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *JMLR*, 9:1775–1822, 2008.

[7] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2:265–292, 2001.

[8] N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. In *CVPR*, 2005.

[9] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, 2005.

[10] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

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

[12] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005.

[13] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, 2007.

[14] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *ICCV*, 2007.

[15] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts. *IEEE PAMI*, 26(2):147–159, 2004.

[16] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[17] I. Laptev and P. Pérez. Retrieving actions in movies. In *ICCV*, 2007.

[18] Y. LeCun, S. Schopra, R. Radsell, R. Marc'Aurelio, and F. Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

[19] J. Liu and M. Shah. Learning human actions via information maximization. In *CVPR*, 2008.

[20] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, 1981.

[21] J. C. Niebles and L. Fei-Fei. A hierarchical model of shape and appearance for human action classification. In *CVPR*, 2007.

[22] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *BMVC*, 2006.

[23] J. C. Platt. Using analytic QP and sparseness to speed training of support vector machines. In *NIPS 11*, 1999.

[24] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell. Hidden conditional random fields. *IEEE PAMI*, 29(10):1848–1852, 2007.

[25] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Berlin, 2003.

[26] C. Schuldt, L. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, 2004.

[27] M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008.

[28] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS 16*, 2004.

[29] B. Taskar, S. Lacoste-Julien, and M. I. Jordan. Structured prediction, dual extragradient and Bregman projections. *JMLR*, 7:1627–1653, 2006.

[30] D. Tran and D. Forsyth. Configuration estimates improve pedestrian finding. In *NIPS 20*, 2007.

[31] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on trees: Message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005.

[32] Y. Wang and G. Mori. Learning a discriminative hidden part model for human action recognition. In *NIPS 21*, 2008.

[33] C. N. Yu and T. Joachims. Learning structural SVMs with latent variables. *NIPS Structured Input-Structured Output Workshop*, 2008.