Hidden Conditional Random Fields for Action Recognition

Lifang Chen[1,2], Nico van der Aa[1,2], Robby T. Tan[1], Remco C. Veltkamp[1]
[1]*Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands*
[2]*Noldus InnovationWorks, Noldus Information Technology, Wageningen, The Neterlands*
l.chen@students.uu.nl, n.vanderaa@noldus.nl, r.t.tan@uu.nl, r.c.veltkamp@uu.nl

Keywords: Action Recognition, (Max-Margin) Hidden Conditional Random Fields, Part Labels Method.

Abstract: In the field of action recognition, the design of features has been explored extensively, but the choice of action classification methods is limited. Commonly used classification methods like k-Nearest Neighbors and Support Vector Machines assume conditional independency between features. In contrast, Hidden Conditional Random Fields (HCRFs) include the spatial or temporal dependencies of features to be better suited for rich, overlapping features. In this paper, we investigate the performance of HCRF and Max-Margin HCRF and their baseline versions, the root model and Multi-class SVM, respectively, for action recognition on the Weizmann dataset. We introduce the Part Labels method, which uses explicitly the part labels learned by HCRF as a new set of local features. We show that only modelling spatial structures in 2D space is not sufficient to justify the additional complexity of HCRF, MMHCRF or the Part Labels method for action recognition.

1 INTRODUCTION

Action recognition is split into feature selection and classificiation. Having extracted features, assigning action labels to videos becomes a classification problem. Next to conventional classifiers like k-Nearest Neighbor (Blank et al., 2005) and Support Vector Machines (SVM) (Jhuang et al., 2007), more complex models have been introduced for action classification, which are either generative or discriminative.

Generative approaches model a joint probability distribution over both the features and their part labels, implying the need of a prior model over the features. To model this prior tractably, generative approaches assume features are conditionally independent of their labels. A typical example is the Hidden Markov Model using hidden states to represent different phases in an action (Yamato et al., 1992).

Discriminative approaches do not need to model the prior on features, since they directly model a conditional distribution over action classes from the features. Therefore, the independence assumption is relaxed. Conditional Random Fields (CRFs) (Kumar and Hebert, 2003) is such a discriminative approach. However, CRF requires fully labelled data where each observation node has an intermediate level label, like ”hands up” or ”put down leg”. Since most available datasets do not provide this intermediate labelling, Quattoni et al. (Quattoni et al., 2004) propose the HCRF model, which extends CRF to incorporate these intermediate part labels as hidden variables. The assignments of these hidden variables are learned during training, not required in the dataset. HCRF was originally proposed for object recognition. Wang and Mori (Wang and Mori, 2011) extended HCRF to action recognition by modelling the spatial dependencies of patches within a frame as they model a human action as a constellation of parts conditioned on image features. They improved the classification performance by combining the flexibility of local representation and the large-scale global representation under the unified framework of HCRF.

Max-margin methods set separating hyperplanes such that the margin between the correct label and all others is maximized, ensuring the score of the correct label is much higher than the incorrect ones. Felzenszwalb et al. (Felzenszwalb et al., 2008) propose the Latent Support Vector Machine (LSVM), which learns a discriminative model with structured hidden (or latent) variables similar to HCRF with a max-margin approach. LSVM is a binary classifier which does not directly handle multi-class classification. Crammer and Singer (Crammer and Singer, 2002) introduce the multi-class SVM which extends the binary SVM to support multi-class classification. Similarly, Wang and Mori (Wang and Mori, 2011) proposed MMHCRF to extend LSVM to directly handle multi-class classification.

Our work is based on HCRF and MMHCRF. Both methods model the spatial structure of an image by structured hidden variables. However, HCRF learns the model parameter with a maximum likelihood approach, while MMHCRF adopts a max-margin approach. We propose a new method that combines the advantages of both HCRF and MMHCRF, leading to more accurate classification results.

2 CLASSIFICATION METHODS

To compare and analyze the value of HCRF and MMHCRF, the theory behind these methods is briefly explained, including their baseline method where the hidden labels are removed.

2.1 Hidden Conditional Random Fields

To classify a frame I in a video sequence, let $\mathbf{x}$ be the feature extracted from I, and y be its action label. Denote $\mathcal{Y}$ as the set of possible action classes. Assume I contains a set of patches $\{I_1, I_2, \ldots, I_m\}$, and its corresponding features can be written as $\mathbf{x} = \{x_0, x_1, \ldots, x_m\}$. x_0 is the global feature vector which is extracted from the whole frame, and $x_i\,(i = 1, \ldots, m)$ is the local feature vector extracted from patch I_i. Our training set consists of labelled frames $(\mathbf{x}_t, y_t)$ for $t = 1, \ldots, T$.

Assume we can assign each patch I_i with a hidden part label h_i from a finite set of possible part labels $\mathcal{H}$, each frame I has a vector of hidden part labels $\mathbf{h} = \{h_1, h_2, \ldots, h_m\}$. A hidden part label represents the motion pattern of a body part, such as move forward for the head. As the values of $\mathbf{h}$ are learned during training, they are the hidden variables of the model.

The hidden part labels can depend on each other. For example, in the case of walking, head and torso might tend to move forward. Assuming an undirected graph structure $G = (V, E)$ for each frame, $h_i\,(i = 1, 2, \ldots, m)$ are the vertices V and the dependence between h_j and h_k is an edge $(j,k) \in E$. Intuitively, G models the conditional dependencies between the hidden part labels. The structure of G is assumed to be a tree (Quattoni et al., 2007). Note that the graph structure can be different from image to image. Figure 1 shows the graphical model.

Given the feature $\mathbf{x}$, part labels $\mathbf{h}$, and class label y, we can define a potential function $\theta^\mathsf{T} \cdot \Phi(\mathbf{x}, \mathbf{h}, y)$ which is parametrized by the model parameter θ:

$$\theta^\mathsf{T} \cdot \Phi(\mathbf{x}, \mathbf{h}, y) = \sum_{j \in V} \alpha^\mathsf{T} \cdot \phi(x_j, h_j) + \sum_{j \in V} \beta^\mathsf{T} \cdot \varphi(y, h_j) + \sum_{(j,k) \in E} \gamma^\mathsf{T} \cdot \psi(y, h_j, h_k) + \eta^\mathsf{T} \cdot \omega(y, x_0), \quad (1)$$

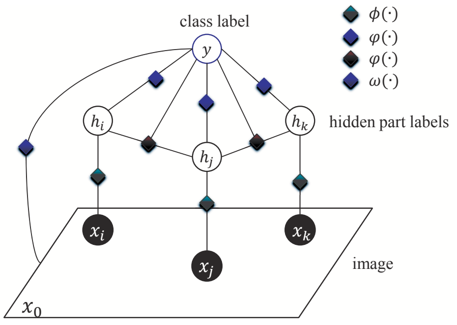


Figure 1: The undirected graph model. The circles and squares correspond to variables and factors, respectively.

where α, β, γ and η are the components of θ, i.e. $\theta = \{\alpha, \beta, \gamma, \eta\}$. Φ is linear with respect to θ. $\phi(\cdot)$, $\varphi(\cdot)$, $\psi(\cdot)$ and $\omega(\cdot)$ are functions defining the features of the model. The unary potential $\alpha^\mathsf{T} \cdot \phi(x_j, h_j)$ models how likely patch x_j is assigned with part label h_j, while the unary potential $\beta^\mathsf{T} \cdot \varphi(y, h_j)$ measures how likely an image with class label y contains a patch with part label h_j. The pairwise potential $\gamma^\mathsf{T} \cdot \psi(y, h_j, h_k)$ measures how likely an image with class label y contains a pair of part labels h_j and h_k, where $(j,k) \in E$. Finally, the root potential $\eta^\mathsf{T} \cdot \omega(y, x_0)$ measures the compatibility of class label y and the global feature of the whole image.

Given the potential function $\theta^\mathsf{T} \cdot \Phi(\mathbf{x}, \mathbf{h}, y)$, the conditional probabilistic model is given as

$$P(y, \mathbf{h} | \mathbf{x}, \theta) = \frac{\exp\left(\theta^\mathsf{T} \cdot \Phi(\mathbf{x}, \mathbf{h}, y)\right)}{\sum_{y' \in \mathcal{Y}} \sum_{\mathbf{h}} \exp\left(\theta^\mathsf{T} \cdot \Phi(\mathbf{x}, \mathbf{h}, y')\right)}. \quad (2)$$

Its denominator is a normalization term which sums over all possible class labels $y' \in \mathcal{Y}$ and all possible combinations of $\mathbf{h}$. When the feature of an image $\mathbf{x}$ and model parameter θ are known, the probability of this image having class label y is the summation of conditional probabilities $P(y, \mathbf{h}|\mathbf{x}, \theta)$ over all possible assignments of part labels $\mathbf{h}$:

$$P(y|\mathbf{x}, \theta) = \sum_{\mathbf{h}} P(y, \mathbf{h} | \mathbf{x}, \theta) \quad (3)$$

The joint conditional probability $P(y|\mathbf{x}, \theta)$ is maximized for all training samples. The objective function used for training parameters θ is defined as:

$$L(\theta) = \sum_t \log P(y_t | \mathbf{x}_t, \theta) - \frac{1}{2\sigma^2} ||\theta||^2. \quad (4)$$

The first term in Eq.(4) is the conditional log-likelihood on the training images. The second term penalizes large values of θ. The optimal θ is learned by maximizing the objective function in Eq.(4). The optimal θ^* cannot be computed analytically; instead we need to employ iterative gradient-based optimization methods such as limited-memory BFGS (Byrd et al., 1994) to search for the optimal θ. Similarly as with other hidden state models like HMMs, adding hidden states $\mathbf{h}$ makes the objective function $L(\theta)$ not convex (Quattoni et al., 2004). Therefore, this method cannot guarantee a global optimal point.

2.2 Root Model

The baseline model of HCRF, called the root model, only uses the global feature x_0 to train the root filter η and does not include the hidden part labels. We only use the last part of the potential function in Eq.(1) for modelling. The probability of class label y given the global feature x_0 and root filter parameter η is:

$$P_{root}(y|x_0;\eta) = \frac{\exp(\eta^\intercal \cdot \omega(y,x_0))}{\sum_{y' \in \mathcal{Y}} \exp(\eta^\intercal \cdot \omega(y',x_0))}. \tag{5}$$

The η^* that optimizes this probability is computed analogously as θ^* for the HCRF.

Besides being a simplified version of HCRF, the root model can also be used to initialize the root filter η of HCRF. Since the objective function of HCRF is not convex, a good starting point of the model parameters lead to a good local optimum. The trained model parameter of the root model, called root filter, can be a good estimation of the root filter η in the HCRF model. The other parameters α, β and γ are initialized randomly.

2.3 Max-Margin HCRF

Instead of using maximum likelihood, the *Max-Margin Hidden Conditional Random Fields* (MMHCRF) uses a max-margin criterion to set the model parameter to maximize the margins between the correct label and the other labels. MMHCRF uses the potential function and its parametrization in Eq.(1) as HCRF does. For a training image, its feature vector and action label pair $(\mathbf{x},y)$ are scored by the potential function with the best assignment of hidden variables:

$$f_\theta(\mathbf{x},y) = \max_{\mathbf{h}} \theta^\intercal \cdot \Phi(\mathbf{x},\mathbf{h},y). \tag{6}$$

Given the training samples $(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_T,y_T)$, we want to find θ that maximizes the margin between the score of the correct label and the score of other labels. Similar to multi-class SVM (Crammer and Singer, 2002), this training process can be formulated as an optimization problem:

$$\begin{aligned} &\min_{\theta,\xi} \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{T}\xi_t \qquad (7) \\ &\text{s.t. } \max_{\mathbf{h}} \theta^\intercal \cdot \Phi(\mathbf{x}_t,\mathbf{h},y) - \max_{\mathbf{h}'} \theta^\intercal \cdot \Phi(\mathbf{x}_t,\mathbf{h}',y_t) \\ &\qquad \leqslant \xi_t - \delta(y,y_t), \forall t, \forall y, \end{aligned}$$

where $\delta(y,y_t)$ is 1 if $y \neq y_t$ and 0 otherwise. Intuitively, we want to find θ whose L_2-norm is as small as possible, and satisfies the constraints that the score for the correct label y_t is at least one larger than the scores of the other labels for each training sample. ξ_t is the slack variable for the t-th training image to handle the soft margin when data is not fully linearly separable, and C controls the trade-off between the slack variable penalty and the size of the margin.

Note that the constraints of Eq.(7) are not convex. Therefore, this method is not guaranteed to reach the global optimum. Using a coordinate descent algorithm similar to (Felzenszwalb et al., 2008), a local optimum of Eq.(7) can be computed by iterating through these two steps:

1. Holding θ, ξ fixed, optimize the hidden part labels $\mathbf{h}'$ for the training example $(\mathbf{x}_t,y_t)$:

$$\mathbf{h}_{t,y_t} = \arg\max_{\mathbf{h}'} \theta^\intercal \cdot \Phi(\mathbf{x}_t,\mathbf{h}',y_t). \tag{8}$$

2. Holding h_{t,y_t} fixed, optimize θ, ξ by solving this optimization problem:

$$\begin{aligned} &\min_{\theta,\xi} \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{T}\xi_t \qquad (9) \\ &\text{s.t. } \max_{\mathbf{h}} \theta^\intercal \cdot \Phi(\mathbf{x}_t,\mathbf{h},y) - \theta^\intercal \cdot \Phi(\mathbf{x}_t,\mathbf{h}_{t,y_t},y_t) \\ &\qquad \leqslant \xi_t - \delta(y,y_t), \forall t, \forall y. \end{aligned}$$

These two steps are repeated until convergence.

During testing, for every new image $\mathbf{x}$, we first calculate the optimal $\mathbf{h}$ for every possible class label y: $\mathbf{h}_y = \arg\max_{\mathbf{h}} \theta^\intercal \cdot \Phi(\mathbf{x},\mathbf{h},y)$. Next, we calculate the score of each class label and pick the label with the highest score: $y^* = \arg\max_y \theta^\intercal \cdot \Phi(\mathbf{x},\mathbf{h}_y,y)$.

2.4 Multi-class SVM

In a similar way as the root model is the baseline model for HCRF, we can derive a root model for MMHCRF, which only uses the root potential $\eta^\intercal \cdot \omega(y,x_0)$ as its potential function and trains the model parameter with a max-margin approach. Setting $f_\theta(\mathbf{x},y) = \eta^\intercal \cdot \omega(y,x_0)$, we obtain:

$$\begin{aligned} &\min_{\theta,\xi} \frac{1}{2}||\theta||^2 + C\sum_{t=1}^{T}\xi_t \qquad (10) \\ &\text{s.t. } \eta^\intercal \cdot \omega(y,x_{t,0}) - \eta^\intercal \cdot \omega(y_t,x_{t,0}) \leqslant \xi_t - 1, \forall t, \forall y \neq y_t \\ &\qquad \xi_t \geq 0, \forall t. \end{aligned}$$

This quadratic program is the standard multi-class SVM (Crammer and Singer, 2002).

3 PART LABEL METHOD

Inspired by the concept of bag-of-words (Niebles and Fei-Fei, 2007), we introduce the Part Labels method to find the best part label assignment for each image using the model parameter trained by HCRF. The main idea behind this method is that if the model parameter is well trained, its learned part labels are descriptive enough for the image, and thus can improve the performance compared to the root model.

3.1 Model Formulation

When the model parameter θ is trained by the HCRF model, ideally, the trained θ should give the correct assignments of part labels higher probabilities, and the incorrect ones lower probabilities. Therefore, we could safely use this θ to find the assignment of part labels with the highest probability $\mathbf{h}^* = \arg\max_{\mathbf{h}} \theta^{\mathsf{T}} \cdot \Phi(\mathbf{x}, \mathbf{h}, y)$ as the correct part labels. Our method is different from MMHCRF since it only adopts the maximization approach to pick the best assignment of part labels once, after the HCRF training process. To obtain the best assignment of part labels $\mathbf{h}^*$ for each training sample, we can use the decoding process of Belief Propagation (Yedidia et al., 2003).

The vector of part labels could be considered as a refined set of local features for the images, like the ”words” for the bag-of-words approach. For example, the part label for the patch on the head describes the movement pattern of the head. It is an abstraction of the patch features. We use these part labels as the local features of this image and combine them with the global feature vector by concatenation: $\mathbf{x}' = (\mathbf{h}^*, x_0)$.

Next, we train the new feature vector $\mathbf{x}'$ in a similar way with the root model. For a training image $(\mathbf{x}', y)$, we define its potential function:

$$\eta'^{\mathsf{T}} \cdot \omega\left(y, \mathbf{x}'\right) = \sum_{a \in \mathcal{Y}} \eta_a'^{\mathsf{T}} \cdot \mathbf{1}_{\{y=a\}} \cdot \mathbf{x}', \qquad (11)$$

where η' is the model parameter, $\omega(\cdot)$ is the feature function, η'_a measures the compatibility between feature $\mathbf{x}'$ and class label $y = a$. η' is the concatenation of η'_a for all $a \in \mathcal{Y}$. The length of vector η' is $|\mathcal{Y}||\mathbf{x}'|$.

Using the potential function defined above, we could define the probability or likelihood of class label y given the feature vector $\mathbf{x}'$:

$$P(y|\mathbf{x}'; \eta') = \frac{\exp\left(\eta'^{\mathsf{T}} \cdot \omega\left(y, \mathbf{x}'\right)\right)}{\sum_{y' \in \mathcal{Y}} \exp\left(\eta'^{\mathsf{T}} \cdot \omega\left(y', \mathbf{x}'\right)\right)}. \qquad (12)$$

The objective function for the set of all training samples can be formulated as the summation of log-likelihood of all samples:

$$L\left(\eta'\right) = \sum_{t=1}^{T} L_t\left(\eta'\right) = \sum_{t=1}^{T} \log P\left(y_t | \mathbf{x}'_t; \eta'\right). \qquad (13)$$

The training process can be formulated as an optimization problem to find the optimal η'^* that gives the maximum of the objective function. We use gradient ascend to search for the optimal η'^*.

3.2 Testing

Given a test image $\mathbf{x}$, we cannot calculate its part labels directly because its class label is unknown. Instead, we calculate the part labels for each class label to obtain a set of $|\mathcal{Y}|$ part labels $\{\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \ldots, \mathbf{h}^{(|\mathcal{Y}|)}\}$, where each part label vector $\mathbf{h}^{(k)}$ is obtained by finding patches using class label $y = k$. Then, we concatenate them with global feature x_0 to form a new set of feature vectors $\{\mathbf{x}'^{(1)}, \mathbf{x}'^{(2)}, \ldots, \mathbf{x}'^{(|\mathcal{Y}|)}\}$. We can calculate the probabilities of all possible assignments of the part labels using η' and classify it by the class label that gives the maximum probabilities.

3.3 Analysis

This method uses the learned part labels as a new set of features and sends them to the training process again. It uses the abstract information contained in the part labels explicitly. The model parameter is learned with a method similar to the root filter learning method. Figure 2 shows the flow chart of the training and testing process. The output of the training process are two model parameters θ and η', which are learned using HCRF and gradient ascent respectively.

This method is similar to the bag-of-words approach, since the part labels can be considered as words. But the way they assign part labels and words is different. This method uses the model trained by HCRF to find the part labels, while bag-of-words first computes a word vocabulary and assigns words to patches by calculating the Euclidean distance. Another difference is that this method combines both global and local features together. The global features contain rich overall information for classification, and local part labels provide a higher level of abstraction from local patch features.

The Part Labels method can be considered as a hybrid of the root model and the HCRF model. It uses the part labels learned by HCRF and trains them using the root model. Compared with the root model, it uses more information than the global feature alone, and compared with the HCRF model, it has an extra maximization step.

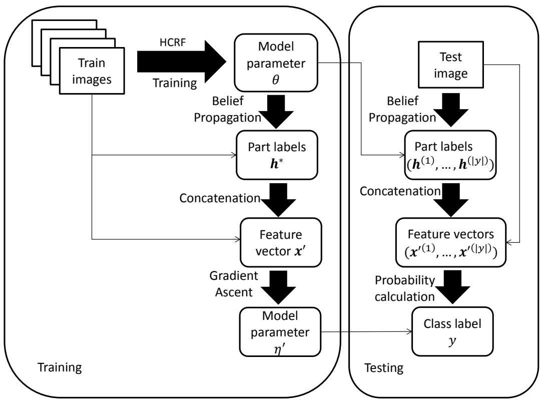


Figure 2: Flow chart of the Part Labels method.

4 EXPERIMENTATION

We test the performance of HCRF, MMHCRF, Root Model, Multiclass SVM and the Part Labels method on the popular publicly available Weizmann dataset (Blank et al., 2005). It contains 83 video sequences at 180×144 pixel resolution and 25 frames per second. The sequences contain nine different people, each performing nine different natural actions: bend, jumping jack (or shortly ”jack”), jump forward on two legs (or ”jump”), jump in place on two legs (or ”pjump”), run, gallop sideways (or ”side”), walk, wave one hand (or ”wave1”), wave two hands (or ”wave2”). The dataset is captured under laboratory settings with fixed background and camera location.

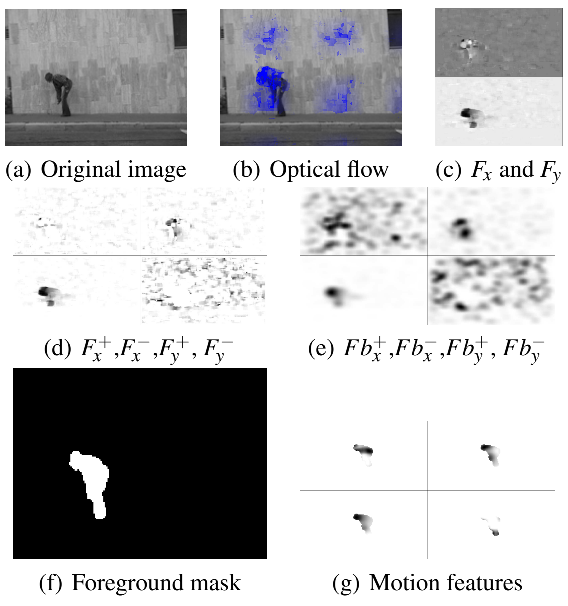
We choose the videos of five subjects as the training set, and the videos performed by the other four subjects as the testing set. All frames in the training set are randomly shuffled so that the training process converges faster. During testing, we classify frame-by-frame in a video (per-frame classification). We can obtain the action label for the whole video by majority voting of its frame labels (per-video classification).

We calculate the motion features of these video sequences in the way similar to what has been proposed in (Efros et al., 2003). This feature is based on pixel-wise optical flow to capture the motion information invariant to appearances (see Figure 3(b)). The optical flow vector F is split into two vectors corresponding to the horizontal and vertical components of the optical flow: F_x and F_y (see Figure 3(c)). F_x and F_y are further split into four non-negative channels: F_x^+, F_x^-, F_y^+ and F_y^-, so that $F_x = F_x^+ - F_x^-$ and $F_y = F_y^+ - F_y^-$ (see Figure 3(d)). To capture only the essential position information, each channel is blurred with a Gaussian kernel and normalized to obtain Fb_x^+, Fb_x^-, Fb_y^+ and Fb_y^- (see Figure 3(e)). The foreground figure is extracted using the mask provided in the dataset (see Figure 3(f)). Next, move the salient region of the person to the center of the view to obtain the final motion features (see Figure 3(g)). This last step is different from the original feature in (Efros et al., 2003), which requires to track and stabilize the video first and compute the optical flow next. With this adjustment we avoid that the correspondence of pixels gets lost due to tracking and stabilizing the person first.

The obtained motion feature vector is the global feature of a frame. We find the local patches on this frame from this global feature vector using the root model. The concatenation of the four channels $\left[Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-\right]$ within the salient region is the motion feature of this patch. To describe the location of a patch, we divide the image into a grid of $w \times h$ bins. The bin where the patch is located is set to 1, all other bins are set to 0. This length $w \times h$ vector is the location feature of this patch. The motion feature vector and location feature vector are concatenated as the feature vector of a patch. The tree structure among the local patches are built by running a minimum spanning tree algorithm over these patches, using the distances between patches as edge weights. The resulting tree structure can be different from frame to frame.

4.1 Root Model Evaluation

The root model only uses the global feature to train the root model parameter η. Since the root model does not contain hidden part labels, it does not need to solve the inference problem for parameter estimation. This makes this method very efficient. In addition, it


(a) Original image (b) Optical flow (c) F_x and F_y

(d) $F_x^+, F_x^-, F_y^+, F_y^-$ (e) $Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-$

(f) Foreground mask (g) Motion features

Figure 3: Calculation of motion features.

	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	0.9502	0.0000	0.0000	0.0000	0.0000	0.0083	0.0000	0.0415	0.0000
jack	0.0199	0.8458	0.0025	0.0771	0.0000	0.0000	0.0000	0.0025	0.0522
jump	0.1090	0.0128	0.7756	0.0000	0.0000	0.0897	0.0128	0.0000	0.0000
pjump	0.0051	0.2335	0.0051	0.7513	0.0000	0.0051	0.0000	0.0000	0.0000
run	0.0000	0.0058	0.0000	0.0000	0.8480	0.0526	0.0643	0.0000	0.0292
side	0.0000	0.0056	0.0056	0.0000	0.0167	0.9667	0.0056	0.0000	0.0000
walk	0.0000	0.0000	0.0318	0.0000	0.0058	0.1012	0.8613	0.0000	0.0000
wave1	0.1545	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.8318	0.0045
wave2	0.0080	0.0040	0.0000	0.0000	0.0161	0.0000	0.0000	0.0321	0.9398

(a) Per-frame classification

	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0
jump	0	0	1	0	0	0	0	0	0
pjump	0	0.25	0	0.75	0	0	0	0	0
run	0	0	0	0	1	0	0	0	0
side	0	0	0	0	0	1	0	0	0
walk	0	0	0	0	0	0	1	0	0
wave1	0.25	0	0	0	0	0	0	0.75	0
wave2	0	0	0	0	0	0	0	0	1

(b) Per-video classification

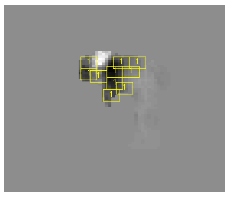
Figure 4: Confusion matrices for the root model.

gives a global optimal solution because its objective function is convex.

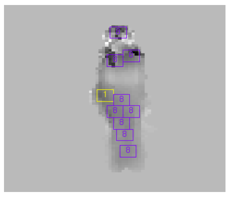
Figure 4 gives the confusion matrices of the per-frame and per-video classification on our feature. For most actions the classification result is good. One ”wave1” video is misclassified ”bend” and one ”pjump” video as ”jack”. The first error is caused by the angle between arm and body is similar for ”wave1” to the angle between upper body and lower body in ”bend”. The second error is due to moving the person to the center of the view, the information about whole body movement in vertical direction is ignored, causing the body torso movements of ”pjump” and ”jack” to be similar to each other. The root filter η for ”pjump” shows that the whole body moves up, while η for ”jack” shows limbs waving around and the torso moves up. After applying the feature on η for ”jack”, the movement of limbs is eliminated and only the torso movement remains, making it hard to distinguish ”pjump” from ”jack”.

The root model does not explicitly include temporal information, since it only uses the time information between two consecutive frames contained in the optical flow feature. As a result, movement patterns of all frames over time in an action are stacked. Movement patterns of arms and legs are projected onto the root filter. A new image is classified as the action whose movement pattern overlaps most. This characteristic of the root model causes confusion if two actions have similar frames and causes the root model to prefer actions with more variations to actions with less variations.

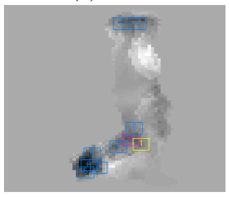
Overall, the root model is efficient and powerful with 0.8659 accuracy on per-frame classification and 0.9474 accuracy on per-video classification.



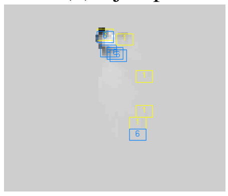
(a) Bend



(b) Pjump



(c) Walk



(d) Wave1

Figure 5: Learned part labels on the Weizmann dataset.

4.2 HCRF Evaluation

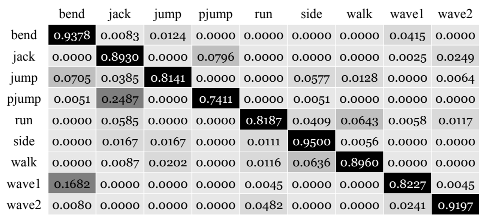
The HCRF model is evaluated with the root filter η initialized using the root filter learned in the previous root model. The parameter settings in these experiments are kept the same with (Wang and Mori, 2011). The size of possible part labels $\mathcal{H} = 10$. The number of salient patches on each frame is set to 10. The size of each patch is 5×5. The other parameters not specified in (Wang and Mori, 2011) are experimentally tuned. The grid division of each frame is set to 10×4. The model parameters α, β and γ are initialized randomly using a Gaussian distribution with mean 0 and standard deviation 0.01. All these parameters are tuned specifically for the Weizmann dataset.

Figure 5 is a visualization of the learned part labels. The patches are labeled with their most likely parts. From this visualization we can make observations about the meaning of the part labels. For example, the part label No.1 in yellow seems to represent the pattern ”moving down” which occurs in ”bend”. The part label No.8 in purple seems to present ”moving up” which happens most in ”pjump”. The part label No.6 in blue seems to represent ”rotating” which could happen in ”walk” and ”wave”.

Figure 6 shows the confusion matrix of per-frame HCRF classification results. If we compare the classification results of the root model and the HCRF model (see Table 1) and their confusion matrix, surprisingly, their outputs are not significantly different

Table 1: Comparison of the root model with HCRF.

	root model	HCRF
Per-frame	0.8659	0.8737
Per-video	0.9474	0.9474



	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	0.9378	0.0083	0.0124	0.0000	0.0000	0.0000	0.0000	0.0415	0.0000
jack	0.0000	0.8930	0.0000	0.0796	0.0000	0.0000	0.0000	0.0025	0.0249
jump	0.0705	0.0385	0.8141	0.0000	0.0000	0.0577	0.0128	0.0000	0.0064
pjump	0.0051	0.2487	0.0000	0.7411	0.0000	0.0051	0.0000	0.0000	0.0000
run	0.0000	0.0585	0.0000	0.0000	0.8187	0.0409	0.0643	0.0058	0.0117
side	0.0000	0.0167	0.0167	0.0000	0.0111	0.9500	0.0056	0.0000	0.0000
walk	0.0000	0.0087	0.0202	0.0000	0.0116	0.0636	0.8960	0.0000	0.0000
wave1	0.1682	0.0000	0.0000	0.0000	0.0045	0.0000	0.0000	0.8227	0.0045
wave2	0.0080	0.0000	0.0000	0.0000	0.0482	0.0000	0.0000	0.0241	0.9197

Figure 6: Per frame confusion matrix of HCRF model.

from each other. This implies the root filter has dominated the HCRF model and lowered the contributions of the other parts. One possible reason of this result is that the global feature and local patch features are the same type of feature. The local patch feature is simply part of the global feature. Therefore, the discriminative power of the global feature and the local patch feature is overlapping with each other. Another reason is that the local patch features in 2D space are not informative enough for action recognition, since the temporal structure of an action is not taken into account. Although this type of features work well on recognition tasks in the 2D domain, like object recognition, it is not sufficient for challenging tasks like action recognition. In Table 2 the classification results of previous work are shown which only use local patch features in 2D space. Their performance is not satisfactory, regardless of their classification results.

Overall, the performance of the HCRF model is comparable to the root model.

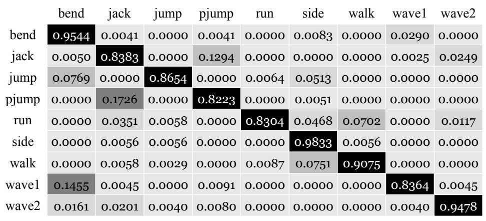
Table 2: Classification results of works using only 2D patch features on Weizmann dataset.

Method	Classification result (%)
(Scovanner et al., 2007)	30.4
(Niebles and Fei-Fei, 2007)	55.0

4.3 MMHCRF Evaluation

We have implemented the MMHCRF model for the Weizmann dataset. Unfortunately, we are not able to get a satisfactory result. To prove that the failure is not caused by the dataset itself or the max-margin approach, we evaluated the Weizmann dataset on a simpler model which only has the root potential and trains its model parameter with a max-margin approach.

Figure 7 shows the confusion matrix of the per-frame multi-class SVM classification results on the Weizmann dataset. The overall accuracy is 0.8867 for per-frame classification and 0.9737 for per-video classification. If we compare this model with the root model, we can see that the Multi-class SVM slightly outperforms the root model. This experiment proves



	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	0.9544	0.0041	0.0000	0.0041	0.0000	0.0083	0.0000	0.0290	0.0000
jack	0.0050	0.8383	0.0000	0.1294	0.0000	0.0000	0.0000	0.0025	0.0249
jump	0.0769	0.0000	0.8654	0.0000	0.0064	0.0513	0.0000	0.0000	0.0000
pjump	0.0000	0.1726	0.0000	0.8223	0.0000	0.0051	0.0000	0.0000	0.0000
run	0.0000	0.0351	0.0058	0.0000	0.8304	0.0468	0.0702	0.0000	0.0117
side	0.0000	0.0056	0.0056	0.0000	0.0000	0.9833	0.0056	0.0000	0.0000
walk	0.0000	0.0058	0.0029	0.0000	0.0087	0.0751	0.9075	0.0000	0.0000
wave1	0.1455	0.0045	0.0000	0.0091	0.0000	0.0000	0.0000	0.8364	0.0045
wave2	0.0161	0.0201	0.0040	0.0080	0.0000	0.0000	0.0000	0.0040	0.9478

Figure 7: Per frame confusion matrix of multi-class SVM.

the strength of the max-margin approach, because the Multi-class SVM trains its model parameter with a max-margin approach and it proves that it is not the dataset why MMHCRF fails.

MMHCRF trains the model parameter in a similar way as the multi-class SVM, but since MMHCRF introduces the hidden part labels, the optimization problem becomes not convex and only a local optimum can be obtained. Different ways of model parameter initialization lead to different local optimal solutions. Hence, the performance of MMHCRF is heavily dependent on its model parameter initialization.

Additionally, MMHCRF is very sensitive to its trade-off parameter C, which controls the trade off between margin size and training error. The bigger C, the less tolerable the system is to the training error.

Finally, the computational complexity of MMHCRF is much higher compared to HCRF, because it needs to solve both inference problem and a quadratic program for every training sample, whereas HCRF only needs to do the inference.

4.4 Part Labels Evaluation

Our novel Part Labels method utilizes the model parameter trained by HCRF to find the most likely part labels for each frame, which are concatenated with the global feature for training. Table 3 shows the comparison of the root model, HCRF, multi-class SVM and this Part Labels method. Figure 8 shows the confusion matrices of the per-frame and per-video classification results of the Part Labels method.

The per-frame classification results of these four models are not really significantly different from each other, most likely since they essentially use the same information. The part labels are learned from the local

Table 3: Comparison of the root model, HCRF, multi-class SVM and Part Labels.

	root model	HCRF	multi-class SVM	Part Labels
Per-frame	0.8659	0.8737	0.8867	0.8705
Per-video	0.9474	0.9474	0.9737	0.9737

	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	0.9046	0.0124	0.0124	0.0041	0.0000	0.0000	0.0000	0.0415	0.0249
jack	0.0075	0.9030	0.0000	0.0746	0.0000	0.0025	0.0000	0.0025	0.0100
jump	0.0449	0.0064	0.8269	0.0000	0.0256	0.0833	0.0128	0.0000	0.0000
pjump	0.0000	0.2487	0.0000	0.7360	0.0000	0.0000	0.0000	0.0051	0.0102
run	0.0000	0.0000	0.0000	0.0000	0.8363	0.0643	0.0994	0.0000	0.0000
side	0.0000	0.0111	0.0167	0.0000	0.0056	0.9611	0.0056	0.0000	0.0000
walk	0.0000	0.0000	0.0318	0.0000	0.0116	0.0925	0.8642	0.0000	0.0000
wave1	0.1045	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8318	0.0636
wave2	0.0080	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0723	0.9197

(a) Per-frame classification

	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0
jump	0	0	1	0	0	0	0	0	0
pjump	0	0.25	0	0.75	0	0	0	0	0
run	0	0	0	0	1	0	0	0	0
side	0	0	0	0	0	1	0	0	0
walk	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	1	0
wave2	0	0	0	0	0	0	0	0	1

(b) Per-video classification

Figure 8: Confusion matrices of the Part Labels method.

patches which are included in the global feature.

The performance of the Part Labels method is still slightly better than the performance of the root model. This is because the Part Labels method use the part labels in addition to the global feature. In the confusion matrix of per-video Part Labels classification, ”wave1” is not misclassified as ”bend”. Even though the global features of these two actions are similar, their part labels are different, as we can see from Figure 5. Using this information in the Part Labels model helps to distinguish them from each other.

5 CONCLUDING REMARKS

This paper introduces a new method for action recognition called the Part Labels method which finds the best assignment of part labels for each image using the model parameters trained by HCRF. By analysing the root model, HCRF, Multi-class SVM, MMHCRF and the newly proposed Part Labels method on a benchmark dataset for human actions, we noticed that the performance of simpler models (the root model and the multi-class SVM) is comparable to the more complex models (HCRF and Part Labels). This is because both HCRF and Part Labels only model the spatial structure, and neglects the temporal structure over frames. For challenging tasks such as action recognition, the spatial structure changes over time and becomes too complex to model.

A natural extension of our work is to include the temporal information. This could be done by including the temporal information in spatio-temporal features or by directly modelling the temporal structure among frames.

ACKNOWLEDGEMENTS

This work was supported by the EU FP7 Marie Curie Network iCareNet under grant number 264738 and the Dutch national program COMMIT.

REFERENCES

Blank, M., Gorelick, L., Shechtman, E., Irani, M., and Basri, R. (2005). Actions as space-time shapes. In *ICCV’05*.

Byrd, R., Nocedal, J., and Schnabel, R. (1994). Representations of quasi-newton matrices and their use in limited memory methods. *Mathematical Programming*, 63:129–156.

Crammer, K. and Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292.

Efros, A., Berg, A., Mori, G., and Malik, J. (2003). Recognizing action at a distance. In *ICCV’03*.

Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *CVPR’08*.

Jhuang, H., Serre, T., Wolf, L., and Poggio, T. (2007). A biologically inspired system for action recognition. In *ICCV’07*.

Kumar, S. and Hebert, M. (2003). Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV’03*.

Niebles, J. and Fei-Fei, L. (2007). A hierarchical model of shape and appearance for human action classification. In *CVPR’07*.

Quattoni, A., Collins, M., and Darrell, T. (2004). Conditional random fields for object recognition. In *NIPS’04*.

Quattoni, A., Wang, S., Morency, L.-P., Collinsl, M., and Darrell, T. (2007). Hidden conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1848–1852.

Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional SIFT descriptor and its application to action recognition. In *Proc. of the 15th international conference on Multimedia*.

Wang, Y. and Mori, G. (2011). Hidden part models for human action recognition: Probabilistic versus max-margin. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1310–1323.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *CVPR’92*.

Yedidia, J., Freeman, W., and Weiss, Y. (2003). Understanding belief propagation and its generalizations. In *Exploring artificial intelligence in the new millennium*, pages 239–269.