International Journal of Computer Vision manuscript No.
(will be inserted by the editor)

A Deep Structured Model with Radius-Margin Bound for 3D Human Activity Recognition

Liang Lin · Keze Wang · Wangmeng Zuo · Meng Wang · Jiebo Luo · Lei Zhang

Received: date / Accepted: date

Abstract Understanding human activity is very challenging even with the recently developed 3D/depth sensors. To solve this problem, this work investigates a novel deep structured model, which adaptively decomposes an activity instance into temporal parts using the convolutional neural networks (CNNs). Our model advances the traditional deep learning approaches in two aspects. First, we incorporate latent temporal structure into the deep model, accounting for large temporal variations of diverse human activities. In particular, we utilize the latent variables to decompose the input activity into a number of temporally segmented sub-activities, and accordingly feed them into the parts (i.e. sub-networks) of the deep architecture. Second, we incorporate a radius-margin bound as a regularization term into our deep model, which effectively improves the generalization performance for classification. For model training, we propose a principled learning algorithm that iteratively (i) discovers the optimal latent variables (i.e. the ways of activity decomposition) for all training instances, (ii) updates the classifiers based on the generated features, and (iii) updates the parameters of multi-layer neural networks. In the experiments, our approach is validated on several complex scenarios for human activity recognition and demonstrates superior performances over other state-of-the-art approaches.

Keywords Human Action and Activity · RGB-Depth Analysis · Structured Model · Deep Learning

L. Lin and K. Wang are with the Sun Yat-sen Unviersity and The Hong Kong Polytechnic University, China (e-mail: linliang@ieee.org).
W. Zuo is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (e-mail: cswmzuo@gmail.com).
M. Wang is with Hefei University of Technology, China. (e-mail: eric.mengwang@gmail.com).
J. Luo is with Department of Computer Science, University of Rochester, U. S. (e-mail: jluo@cs.rochester.edu).
L. Zhang are with Department of Computing, The Hong Kong Polytechnic University. (e-mail: cslzhang@comp.polyu.edu.hk).

1 Introduction

In computer vision, it has received increasing attention in human activity understanding to determine what people are doing given an observed video in different application domains, e.g. intelligent surveillance, robotics, and human - computer interaction. Recently developed 3D/depth sensors have opened up new opportunities with enormous commercial values, which provide more rich information (e.g. extra depth data of scenes and objects) compared with the traditional cameras. Built upon the enriched information, human poses can be estimated more easily. However, modeling complicated human activities still remains challenging, mainly due to the following difficulties.

– (**a**) The complexity of representing high-level activities with the rich appearance and motion information from video. The actors may appear in diverse views or poses under different motions, and the surrounding objects and environments can also vary within the same activity category. Moreover, the depth maps provided by the 3D sensors are often unavoidably contaminated [30] due to the noise or the self-occlusion of the body parts.
– (**b**) The ambiguity in the temporal segmentation of the sub-activities which constitute an activity. An activity can be considered as a sequence of actions (i.e. sub-activities) occurred over time [5]. For instance, the activity of "microwaving food" can be temporally decomposed into several parts such as picking up food, walking and operating microwave. However, the activity composition may vary for a category of activity instances. Figure 1 shows two activities belonging to the same category, where the temporal lengths of decomposed actions are different for different subjects. It is therefore difficult to capture the temporal variation of activities during the category recognition.

Most of previous methods recognize 3D human activities by training discriminative/generative classifiers based on carefully designed features [50, 30, 49, 44]. These approaches often require sufficient domain knowledge and heavy feature engineering because of the difficulty (**a**), which could limit their applications. To improve the discriminative performance, some compositional methods [46, 5] model complex activities by segmenting the videos into temporal segments of fixed length. But because of the difficulty (**b**), they may have problems handling complex activities composed of actions of diverse temporal durations, e.g. the examples in Figure 1.

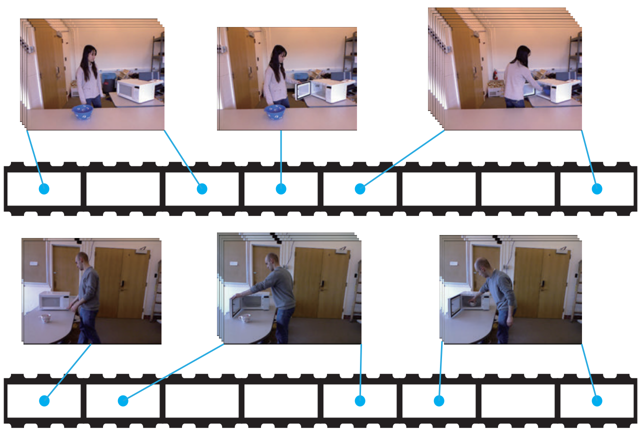


Fig. 1 Two activities of the same category. We consider one activity as a sequence of actions occurred over time, and the temporal composition of an action may differ for different subjects.

In this work, we develop a deep structured human activity model to address the above mentioned challenges, and demonstrate superior performance over other state-of-the-art approaches on the task of recognizing human activities from Grayscale-Depth videos which are captured by a RGB-D camera (i.e. Microsoft Kinect). Our model adaptively represents the input activity instance as a sequence of temporally separated sub-activities, and each one is associated with a cubic-like video segment of a flexible length. Our model is inspired by the effectiveness of two widely successful techniques: deep learning [21, 14, 20, 16, 48, 26, 45] and the latent structured models [56, 11, 1, 32, 24]. One example of the former is Convolutional Neural Networks (CNNs), which was recently applied to generate powerful features for video classification [16, 17]. On the other hand, the latent structured models (such as Deformable Part-based Model [11]) have been demonstrated as an effective class of models for handling large object variations for recognition and detection. One of the key components in these models is the reconfigurable flexibility of model structure, which often implemented by estimating latent variables during inference.

We adopt the deep CNN architecture [21, 16] to layer-wisely extract features from the input video data, and the architecture are vertically decomposed into several sub-networks corresponding to the video segments, as Figure 2 illustrates. In particular, our model searches for the optimal composition for each activity instance during the recognition, which is the key to handle the temporal variation of human activities. Moreover, we introduce relaxed radius-margin bound into our deep model, which effectively improves the generalization performance for classification. In the following, we briefly overview the main components of our model and summarize the advantages.

First, the configuration of our deep model can be flexibly adjusted to adapt to different input videos, and the significance of this property has been justified for human action recognition [22, 38, 45]. In our approach, we make our model adaptively capture temporal structure by using the latent variables. This motivation finely accords with a batch of existing part-based structured models in visual recognition [24, 27]. More specifically, we utilize the latent variables to explicitly represent the temporal composition of the human activities, i.e. the input video is partitioned into several segments of alterable lengths (each segment indicating a sub-activity). The different temporal compositions actually correspond to the different temporal durations of the separated sub-activities. And the frames of different video segments are extracted to feed to the corresponding sub-networks.

During the inference of activity recognition, we aggregate the responses from sub-networks while searching for the optimal temporal activity segmentation. This inference will inevitably cause extra computation cost just like traditional latent structured models [24]. It is worth mentioning that we can implement the inference in a parallel manner using GPU (Graphic Processing Unit) programming, in order to counter-balance the extra computational demand.

Second, we integrate the radius-margin regularization with the deep feature learning, effectively conducting the classification with good generalization performance. Collecting 3D data of human activities is relatively expensive in practice, while the large amount of training data plays a critical role in recent successful deep learning approaches [20, 27, 17]. On the other hand, the max-margin methods (e.g. Support Vector Machines) have shown very impressive generalization power and thus been widely applied for small scale training data. According to [9, 7], their performance (i.e. the error rate) for classification depends on not only the margin of positive/negative samples but also the radius of the enclosing ball of all samples, and this is more critical for joint learning of feature representation and classifier. Inspired by these works, we incorporate a radius-margin bound as a regularizer into our deep model, and demonstrate better generalization performance compared to the softmax or SVM classifier. More detailed discussion will be presented Section 3.3.

Training our deep structured model is nontrivial, as it needs to jointly optimize three components: (i) the activity decomposition, (ii) the classifier upon the generated features, and (iii) the neural networks. Seeking the global optimum for such a model is extremely intractable due to the non-convexity, and we consider an approximate solution by iteratively optimizing these components for a local convergence. In each iteration, the learning algorithm performs the following three steps.

1. We compute the optimal latent variables (i.e. sub-activity decompositions) for all training activities, and their feature vectors are then specified.
2. Based on the generated features, we optimize the classification margin of all training examples under the fixed radius bound.
3. We learn the parameters of the CNNs using the traditional backward propagation, which will lead to the decrease of the radius.

The main contributions of this work are several folds. First, we present a novel deep neural network model to handle various challenges in 3D human activity recognition, and demonstrate superior performance over state-of-the-art approaches under several challenging scenarios. Second, our deep model incorporates latent temporal structure to account for large temporal variations of diverse human activities. To the best of our knowledge, this is a novel contribution to the literature of deep learning. Third, we unify the radius-margin method with the feature learning in a principled way, providing a very general framework for many classification tasks. In addition, we construct a new database of RGB-D data, which includes 1180 instances of human activities in 20 categories.

The remainder of the paper is organized as follows. Section 2 presents a review of related work. Then we present our deep model in Section 3 and 4, followed by a description of model learning algorithm in Section 5. Section 6 discusses the procedure of activity recognition using our model. The experimental results, comparisons and component analysis are exhibited in Section 6. Section 7 concludes this paper.

2 Related Work

Many works on human action/activity recognition mainly focus on designing robust and descriptive features [49, 13, 30, 29, 55, 51, 34]. For example, Xia and Aggarwal [49] extracted spatio-temporal interest points from depth videos (DSTIP) and developed a depth cuboid similarity feature (DCSF) to model human activities. Oreifej and Liu [30] proposed to capture spatio-temporal changes of activities by using a histogram of oriented 4D surface normals (HON4D). Most of these methods, however, overlooked detailed spatio-temporal structure information, and limited in periodic activities.

Several compositional part-based approaches have been studied for complex scenarios and achieved substantial progresses [46, 42, 54, 31, 33, 44, 6], and they represent an activity with the deformable parts and contextual relations. For instance, Wang et al. [46] recognized human activities in common videos by training the hidden conditional random fields in a max-margin framework. For activity recognition in RGB-D data, Packer et al. [31] employed the latent structural SVM to train the model with part-based pose trajectories and object manipulations. An ensemble model of actionlets were studied in [44] to represent 3D human activities with a new feature called local occupancy pattern (LOP). To handle more complicated activities with large temporal variations, some improved models [38, 43, 3] discovered temporal structures of activities by localizing sequential actions. For example, Wang and Wu [43] proposed to solve the temporal alignment of actions by maximum margin temporal warping. Tang et al. [38] captured the latent temporal structures of 2D activities based on the variable-duration hidden Markov model. Koppula and Saxena [18] applied the Conditional Random Fields to model the sub-activities and affordances of the objects for 3D activity recognition.

Recently, the And-Or graph representations are introduced as extensions of the part-based models [56, 32, 22, 39, 23], and produce very competitive performance to deal with large data variations. These models incorporate not only the hierarchical decompositions, but also the explicit structural alternatives (e.g. the different ways of compositions). Zhu and Mumford [56] first explored the And-Or graph models for image parsing. Pei et al. [32] then introduced the models for video event understanding, but their approach required elaborate annotations. Liang et al. [22] proposed to train the spatio-temporal And-Or graph model using a non-convex formulation, which is discriminatively trained from weakly annotated training data. However, the above mentioned models rely on the hand-crafted features, and their discriminative capacities are not optimized for 3D human activity recognition.

In the mean time, the past few years have seen a resurgence of research in the design of deep neutral networks, and impressive progresses have been made on learning image features from raw data [14, 20, 25]. To address human action recognition from videos, Ji et al. [16] developed a novel deep architecture of convolutional networks, where they extracted features from both spatial and temporal dimensions. Luo et al. [27] proposed to incorporate a new Switchable Restricted Boltzmann Machine (SRBM) to explicitly model the complex mixture of visual appearance for pedestrian detection, and train their model using an EM-type interative algorithm. Amer and Todorovic [1] applied Sum Product Net-

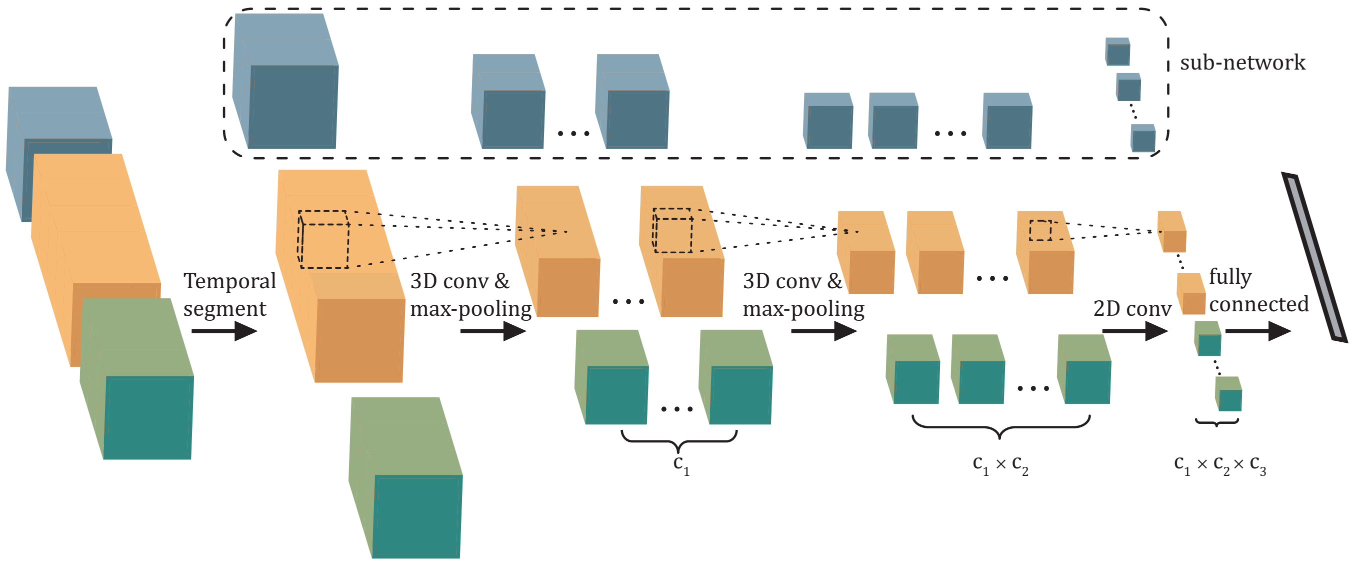


Fig. 2 The architecture of spatio-temporal convolutional neural networks. The neural networks are stacked up by convolutional layers, max-pooling operators and a fully connected layer, where the raw segmented videos are treated as the input. A sub-network is referred to a vertically-decomposed subpart stacked up by several layers, which extracts features for one segmented video section (i.e. one sub-activity). Moreover, by using the latent variables, our architecture is capable of explicitly handling diverse temporal compositions of complex activities.

works (SPNs) to model human activities based on variable primitive actions. Our deep model is partially motivated by these works, and we target on an more flexible and powerful solution by jointly considering the latent structure embedding, feature learning, and radius-margin classification.

Recently, recurrent neural networks (RNN) has been used for activity recognition due to its capability in modeling complex temporal dynamics. Donahue et al. [10] presented a long-term recurrent convolutional network (LRCN) architecture to integrates CNN and RNN into an unified model, and achieved promising results in a number of vision tasks. Rohrbach et al. [41] further improved LRCN by adding a pooling layer and had shown its potentials in video description. The main difference between RNN models and ours is that their models exploit several types of neural gates and memory cells to learn temporal dynamics implicitly, while our deep structured model explicitly accounts for temporal variations of human activities by inferring latent variables. Speficially, compared with these RNN models, our model has the following advantages. First, the temporal composition is explicitly captured by our model, giving rise to a better interpretability, i.e. the semantic correspondence of video segments and sub-activities. Second, as some recent works report [2], the RNN models may have problems on using common dropout tricks and this limitation would influence the performances. Moreover, the integration with explicit regularization approaches (e.g. the radius-margin bound) is also an important superiority of our model.

3 Deep Structured Model

In this section, we introduce the main components of our deep structured model, including the spatio-temporal CNNs, the latent structure of activity decomposition, and the radius-margin bound for classification.

3.1 Spatio-temporal CNNs

We propose an architecture of spatio-temporal convolutional neural networks (CNNs), as Figure 2 illustrates. In the input layer, the activity video is decomposed into M video segments, where each segment associates to one separated sub-activity. Accordingly, the proposed architecture consists of M sub-networks to extract features from the corresponding decomposed video segments, respectively. Our spatio-temporal CNNs involve both 3D and 2D convolutional layers. The 3D convolutional layer extracts spatio-temporal features for jointly capturing appearance and motion information, and is followed by a max-pooling operator to improve the robustness against local deformations and noise. As shown in Figure 2, each sub-network (highlighted by the dashed box) is stacked up by two 3D convolutional layers and one 2D convolutional layer. For the input to each sub-network, the number of frames is very small (e.g. 9). After two layers of 3D convolution followed with max-pooling, the temporal dimension for each set of feature maps is too small to perform 3D convolution. Thus, we stack a 2D convolutional layer upon the two 3D convolutional layers. The outputs from different sub-networks are merged to be fed to one fully connected layer that generates the final feature vector of the input video.

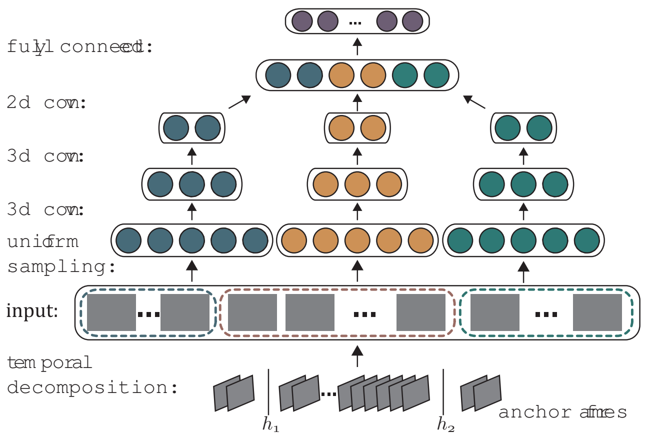


Fig. 3 Illustration of incorporating latent structure into the deep model. Different sub-networks are denoted by different colors.

3.2 Latent Temporal Structure

Unlike the traditional deep learning methods with the fixed architectures, we incorporate latent structure into the deep model to flexibly adapt to the input video during inference and learning. To address the large temporal variation of human activities, we assume the input video is temporally divided into a number M of segments, corresponding to the sub-activities. We associate the CNNs with the video segmentation by feeding each segmented part into a sub-network as Figure 2 illustrates. Next, according to the way of video segmentation (i.e. decomposition of sub-activities), we manipulate the CNNs by inputting sampled video frames.

Specifically, we index each video segment by its starting anchor frame s_j and its temporal length (i.e. the number of frames) t_j for each sub-network, which must take m video frames as the input. Note that when $t_j \neq m$, a uniform sampling is performed to extract m key frames. Thus, for all video segments, we denote the indexes of starting anchor frames as $(s_1,...,s_M)$ and their temporal lengths as $(t_1,...,t_M)$, which are regarded as the latent variables in our model, $h=(s_1,...,s_M,t_1,...,t_M)$. These latent variables specifying the segmentation will be adaptively estimated for different input videos. Figure 3 shows an intuitive example of our structured deep model, where the input video are segmented into three sections corresponding to the three sub-networks in our deep architecture. In this way, the configuration of the CNNs are dynamically adjusted together with searching for the appropriate latent variables of input videos.

Given the parameters of CNNs ω and the input video x_i with its latent variables h_i, the generated feature of x_i can be represented as $\phi(x_i;\omega,h_i)$.

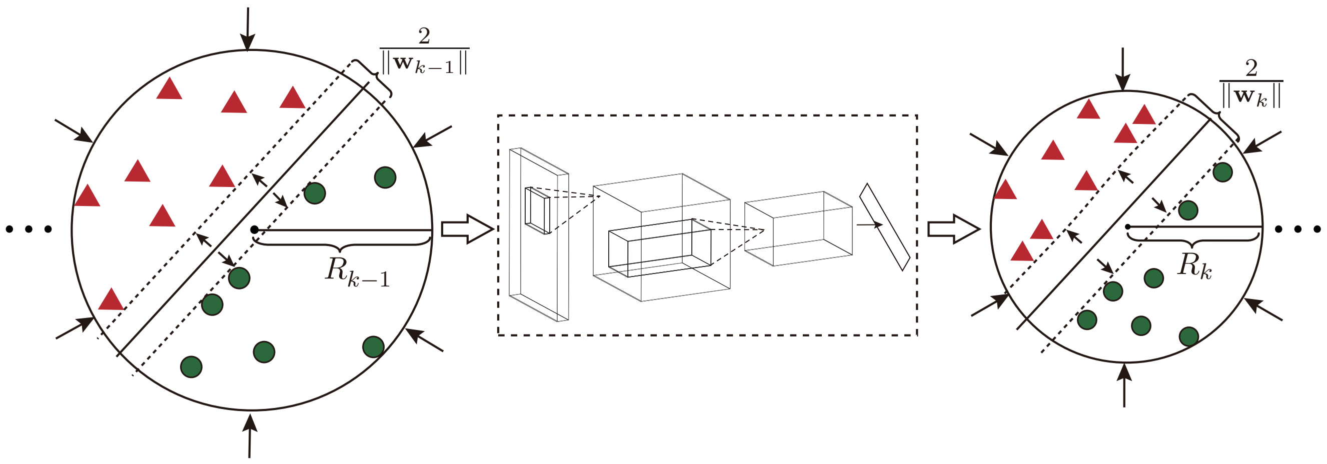
3.3 Deep Model with Relaxed Radius-Margin Bound

The large amount of training data is crucial for the success of many deep learning models. Given sufficient training data, the effectiveness of applying the softmax classifier with CNNs has been validated for image classification [20]. However, for 3D human activity recognition, the available training data are usually less what we expected. For example, the CAD-120 dataset [19] consists of only 120 RGB-D sequences of 10 categories. Under this scenario, though parameter pre-training and dropout are available, the model training often suffers from the over-fitting issue. Hence, we consider introducing a more effective classifier together with regularizer to improve the generalization performance of the deep model.

In supervised learning, Support Vector Machine (SVM), also known as the max-margin classifier, is theoretically sound and generally can achieve promising performance compared with the alternative linear classifiers. In the deep learning research, the combination of SVM and CNNs has been exploited [15] and obtained excellent results in object detection [12]. Motivated by these approaches, we impose a max-margin classifier $(\mathbf{w},b)$ upon the feature generated by the spatio-temporal CNNs for human activity recognition.

As a max-margin classifier, standard SVM adopts $\|\mathbf{w}\|^2$, the reciprocal of the squared margin γ^2, as the regularizer. However, the generalization error bound of SVM depends on the radius-margin ratio R^2/γ^2, where R is the radius of the minimum enclosing ball (MEB) of the training data [40]. When the feature space is fixed, the radius R is constant and can thus be ignored. However, in our approach, the radius R is determined by the MEB of the training data in the feature space generate by the CNNs. Under this scenario, the model has the risk that the margin can be increased by simply expanding the MEB of the training data in the feature space. For example, simply multiplying a constant to the feature vector can enlarge the margin between the positive and negative samples, but obviously it will not really work for better classification. To overcome this problem, we incorporate the radius-margin bound together with the feature learning, as Figure 4 illustrates. In particular, we impose a max-margin classifier with radius information upon the feature generated by the fully connected layer of the spatio-temporal CNNs. The optimization tends to maximize the margin while shrinking the MEB of the training data in the feature space, and we thus obtain a tighter error bound.

Suppose there are a set of N training samples $(X,Y) = \{(x_1,y_1), ... , (x_N,y_N)\}$, where x_i is the video, $y \in \{1,...,C\}$



$\frac{2}{\|\mathbf{w}_{k-1}\|}$
R_{k-1}
$\frac{2}{\|\mathbf{w}_{k}\|}$
R_k

Fig. 4 Illustration of our deep model with the radius-margin bound. To improve the generalization performance for classification, we propose to integrate the radius-margin bound as a regularizer with the feature learning. Intuitively, together with optimizing the max-margin parameters $(\mathbf{w}, b)$, we shrink the radius R of the minimum enclosing ball (MEB) of the training data that distribute in the feature space generated by the CNNs. The resulting classifier with the regularizer shows better generalization performance compared to the traditional softmax output.

represents the category labels and C is the number of activity categories. We extract the feature for each x_i by the spatio-temporal CNNs, $\phi(x_i;\omega,h_i)$, where h_i refers to the latent variables. By adopting the squared hinge loss and the radius-margin bound, we define the following loss function L_0 of our model:

$$L_0 = \overbrace{\frac{1}{2}\|\mathbf{w}\|^2 R_\phi^2}^{Radius-margin\ Ratio} + \lambda \sum_{i=1}^{N} \max\left(0, 1-\left(\mathbf{w}^T\phi(x_i;\omega,h_i)+b\right)y_i\right)^2, \tag{1}$$

where λ is the trade-off parameter, $1/\|\mathbf{w}\|$ denotes the margin of the separating hyperplane, b denotes the bias, and R_ϕ denotes the radius of the MEB of the training data $\phi(X,\omega,H) = \{\phi(x_1;\omega,h_1),\ldots,\phi(x_N;\omega,h_N)\}$ in the CNNs' feature space. Formally, the radius R_ϕ is defined as [4, 40],

$$R_\phi^2 = \min_{R,\phi_0} R^2, s.t. \|\phi(x_i;\omega,h_i)-\phi_0\|^2 \le R^2, \forall i. \tag{2}$$

The radius R_ϕ is implicitly defined by both the training data and the model parameters, making that: (i) the model in Eq. (1) is highly nonconvex, (ii) the derivative of R_ϕ with respect to ω is hard to compute, and (iii) the problem is difficult to solve using the stochastic gradient descent (SGD) method. Motivated by the radius-margin based SVM [9, 8], we investigate the relaxed form to replace the original definition of R_ϕ in Eq. (2). In particular, we introduce the maximum pairwise distance $\tilde{R}_\phi$ over all the training samples in the feature space, as

$$\tilde{R}_\phi^2 = \max_{i,j} \|\phi(x_i;\omega,h_i)-\phi(x_j;\omega,h_j)\|^2. \tag{3}$$

Do and Kalousis [8] proved that R_ϕ could be well bounded by $\tilde{R}_\phi$ with the following lemma,

Lemma 1

$$\tilde{R}_\phi \le R_\phi \le \frac{1+\sqrt{3}}{2}\tilde{R}_\phi.$$

This above Lemma guarantees that the true radius R_ϕ can be well approximated by $\tilde{R}_\phi$. With the proper parameter η, the optimal solution for minimizing the radius-margin ratio $\|\mathbf{w}\|^2 R_\phi^2$ is the same with that for minimizing the radius-margin sum $\|\mathbf{w}\|^2 + \eta R_\phi^2$ [8]. Thus, by approximating R_ϕ^2 with $\tilde{R}_\phi^2$ and replacing the radius-margin ratio with the radius-margin sum, we suggest the following deep model with the relaxed radius-margin bound,

$$L_1 = \frac{1}{2}\|\mathbf{w}\|^2 + \max_{i,j}\|\phi(x_i;\omega,h_i)-\phi(x_j;\omega,h_j)\|^2 + \lambda \sum_{i=1}^{N} \max\left(0, 1-\left(\mathbf{w}^T\phi(x_i;\omega,h_i)+b\right)y_i\right)^2. \tag{4}$$

However, the first max operator in Eq. (4) is non-smooth and defined over all pairs of training samples, and it is thus unsuitable for using the mini-batch-based SGD optimization method. In the following, we first use the softmax function to avoid the non-smoothness of the max operator, and then further relax the radius to avoid the definition over all pairs of training samples. More specifically, we first transfer the max operator into a softmax form, resulting the following model,

$$L_2 = \frac{1}{2}\|\mathbf{w}\|^2 + \eta \sum_{i,j} \kappa_{ij}\|\phi(x_i;\omega,h_i)-\phi(x_j;\omega,h_j)\|^2 + \lambda \sum_{i=1}^{N} \max\left(0, 1-\left(\mathbf{w}^T\phi(x_i;\omega,h_i)+b\right)y_i\right)^2. \tag{5}$$

with

$$\kappa_{ij} = \frac{\exp(\alpha\|\phi(x_i;\omega,h_i) - \phi(x_j;\omega,h_j)\|^2)}{\sum_{ij}\exp(\alpha\|\phi(x_i;\omega,h_i) - \phi(x_j;\omega,h_j)\|^2)}, \tag{6}$$

where κ_{ij} is a coefficient measuring the correlation of the two samples and $\alpha \geq 0$ is the parameter to control the approximation degree to the hard max operator. When α is infinite, the approximation in Eq. (5) becomes the model in Eq. (4). Specifically, when $\alpha = 0$, there is $\kappa_{ij} = 1/N^2$, and the relaxed loss function can be reformulated as:

$$\begin{aligned} L_3 =& \frac{1}{2}\|\mathbf{w}\|^2 + 2\eta\sum_i \|\phi(x_i;\omega,h_i) - \bar{\phi}_\omega\|^2 \\ &+ \lambda\sum_{i=1}^{N}\max\left(0, 1 - \left(\mathbf{w}^T\phi(x_i;\omega,h_i) + b\right)y_i\right)^2. \end{aligned} \tag{7}$$

with

$$\bar{\phi}_\omega = \frac{1}{N}\sum_i \phi(x_i;\omega,h_i). \tag{8}$$

The optimization objectives in Eq. (5) and (7) are two relaxed losses of our deep model with the strict radius-margin bound in Eq. (1). In this work, we focus on the objective in Eq. (7) for the model training. The learning algorithm will be discussed in Section 5.

4 Implementation

In this section, we first explain the implementation that makes our model adaptive to alterable temporal structure, and then describe the detailed setting of our deep architecture.

4.1 Latent Temporal Structure

During our learning and inference procedures, we search for the appropriate latent variables that determine the temporal decomposition of the input video (i.e. the decomposition of activities). There are two parameters related to the latent variables in our model: the number M of video segments and the temporal length m of each segment. Note that the sub-activities decomposed by our model have no precise definition given a complex activity, i.e. actions can be ambiguous depending on the considering temporal scale.

To incorporate the latent temporal structure, we associate the latent variables with the neurons (i.e. convolutional responses) in the bottom layer in the spatio-temporal CNNs.

The choice of the number of segments M is important to the performance of 3D human activity recognition. The model with a small M could be less expressive to handle temporal variations, while a large M could lead to overfitting due to high complexity. Furthermore, when $M = 1$, the model latent structure would be disabled, and our architecture degenerates to the conventional 3D-CNNs [16]. By referring to the setting of the number of parts for the deformable part-based model [11] in object detection, the value M can be set by the cross validation on a small set. In all our experiments, we set $M = 4$.

Considering that the number of frames of the input videos are diverse, we develop a process to normalize the inputs by two-step sampling in the learning and inference procedure. First, we sample 30 anchor frames uniformly from the input video. Based on these anchor frames, we search for all of the possible non-overlapped temporal segmentations, and the anchor frame segmentation corresponds to the segmentation of the input video. Then, from each video segment (indicating a sub-activity) we uniformly sample m frames to feed the neural networks, and in our experiments we set $m = 9$. In addition, we reject the possible segmentations that cannot offer m frames for any video segment.

For an input video, the possibility of temporal structure variations is 115 in our experiments (i.e. the possible enumeration numbers of anchor frame segmentation).

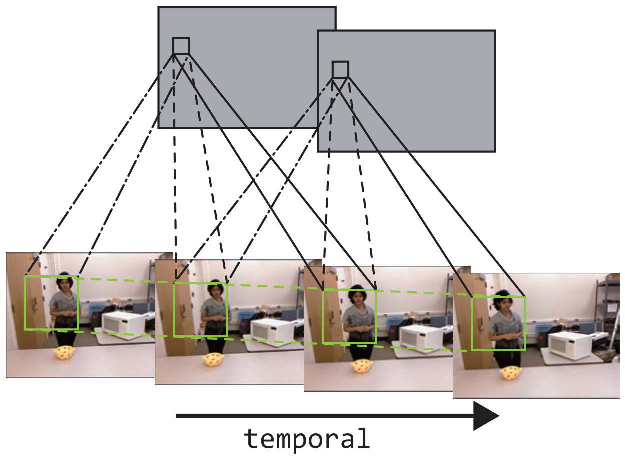


Fig. 5 Illustration of the 3D convolution across both spatial and temporal domains. In this example, the temporal dimension of the 3D kernel is 3, and each feature map is thus obtained by performing 3D convolutions across 3 adjacent frames.

4.2 Architecture of Deep Neural Networks

The proposed spatio-temporal CNN architecture is constructed by stacking up two 3D convolution layers, one 2D convolution layer and one fully connected layer, and the max-pooling operator is deployed after each 3D convolutional layer. In the following, we introduce the definitions and implementations of these components in our model.

3D Convolutional Layer. The 3D convolution operation is adopted to perform convolutions spanning over both spatial and temporal dimensions for the characterization of both appearance and motion features [16]. Suppose $\mathbf{p}$ is the input video segment with the width w, the height h, and the number of frames m, ω is the 3D convolutional kernel with the the width w', height h', and temporal length m'. As shown in Figure 5, a feature map $\mathbf{v}$ can be obtained by performing 3D convolutions from the sth to the $(s+m'-1)$th frames, where the response for the position (x,y,s) in the feature map is defined as,

$$v_{xys} = \tanh(b + \sum_{i=0}^{k'-1}\sum_{j=0}^{h'-1}\sum_{k=0}^{m'-1} \omega_{ijk} \cdot p_{(x+i)(y+j)(s+k)}), \tag{9}$$

where $p_{(x+i)(y+j)(s+k)}$ denotes the pixel value of the input video $\mathbf{p}$ at position $(x+i, y+j)$ in the $(s+k)$th frame, ω_{ijk} denotes the value of the convolutional kernel ω at the position (i,j,k), b stands for the bias, and tanh denotes the hyperbolic tangent function. Thus, given $\mathbf{p}$ and ω, $m-m'+1$ feature maps can be obtained, each with size of $(w-w'+1, h-h'+1)$.

Based on the 3D convolution operation, 3D convolution layer is designed for spatio-temporal feature extraction by considering three issues:

- *Number of convolutional kernels.* The feature maps generated by one convolutional kernel are limited in capturing appearance and motion information. To generate more types of features, several kernels are employed in each convolutional layer. We define the number of 3D convolutional kernels in the first layer as c_1. After the first 3D convolutions, we obtain c_1 sets of $m-m'+1$ feature maps. Then we use 3D convolutional kernels on the c_1 sets of feature maps, and obtain $c_1 \times c_2$ sets of feature maps after the second 3D convolution layer.
- *Decompositional convolutional networks.* Our deep model consists of M sub-networks, and the input video segment to each sub-network involves m frames (the later frames might be unavailable). In the proposed architecture, all of the sub-networks use the same structure but each one has its own convolutional kernels, as we assume that each temporally decomposed sub-activity has its distinct features in terms of discriminative classification. For example, the kernels belonging to the first sub-network are only deployed to perform convolutions on the first temporal video segment. .
- *Application to gray-depth video.* The RGB images are first converted to the gray-level images, and the gray-depth video is then adopted as the input to the neural networks. The 3D convolutional kernels in the first layer are respectively applied for both the gray channel and the depth channel in the video, and the convolution results from these two channels are further aggregated to produce the feature maps. Note that the dimensions of the features remain the same as from only one channel.

In our implementation, the input frame is scaled with the height $h = 80$ and width $w = 60$. In the first 3D convolution layer, the number of 3D convolutional kernels is $c_1 = 7$, and the size of the kernel is $w' \times h' \times m' = 9 \times 7 \times 3$. In the second layer, the number of 3D convolutional kernels is $c_2 = 5$, and the size of the kernel is $w' \times h' \times m' = 7 \times 7 \times 3$. Thus, we have 7 sets of feature maps after the first 3D convolution layer, and obtain 7×5 sets of feature maps after the second 3D convolution layer.

Max-pooling Operator. After each 3D convolution, the max-pooling operation is introduced to enhance the deformation and shift invariance [20, 52]. Given a feature map with the size of $a_1 \times a_2$, a $d_1 \times d_2$ max-pooling operator is performed by taking the maximum of every non-overlapping $d_1 \times d_2$ sub-regions of the feature map, resulting in an $a_1/d_1 \times a_2/d_2$ pooled feature map. In our implementation, 3×3 max-pooling operator was applied after every 3D convolution layers. After two layers of 3D convolution and max-pooling, for each sub-network, we have 7×5 sets of $6 \times 4 \times 5$ feature maps.

2D Convolutional Layer. After two layers of 3D convolution followed with max-pooling, 2D convolution is employed to further extract higher-level complex features. The 2D convolution can be viewed as a special case of 3D convolution with $m' = 1$, which is defined as

$$v_{xy} = \tanh(b + \sum_{i=0}^{k'-1}\sum_{j=0}^{h'-1} \omega_{ij} \cdot p_{(x+i)(y+j)}), \tag{10}$$

where $p_{(x+i)(y+j)}$ denotes the pixel value of the feature map $\mathbf{p}$ at position $(x+i, y+j)$, ω_{ij} denotes the value of the convolutional kernel ω at the position (i,j), and b denotes the bias. In the 2D convolution layer, suppose the number of 2D convolutional kernels is c_3, $c_1 \times c_2 \times c_3$ sets of new feature maps are obtained by performing 2D convolutions on $c_1 \times c_2$ sets of feature maps generated by the the second 3D convolution layer.

In our implementation, the number of 2D convolutional kernels is set as $c_3 = 4$ with the kernel size 6×4. Hence for each sub-network we can obtain 700 feature maps with size 1×1.

Fully Connected Layer. There is only one fully connected layer with 64 neurons in our architecture. All these neurons connect to a vector of $700 \times 4 = 2800$ dimensions, which is generated by concatenating the feature maps from all of the sub-networks. The margin-based classifier is defined based on the output of the fully connected layer, where we adopt the squared hinge loss to predict the activity categories as

$$\theta(z) = \arg\max_i (w_i^T z + b_i) \tag{11}$$

where z is the 64-dimensional vector from the fully connected layer, and $\{w_i, b_i\}$ denotes the weight and bias connected to the i-th activity category.

Dropout trick. Since our deep architecture contains a large number of parameters (i.e. 179200 weighs at the fully-connected layer), we apply the standard dropout approach during the model training to alleviating the over-fitting problem. According to the recent reports [36], the dropout method is capable of effectively improving the generalization power of neural network models by randomly turning off the neurons in the learning. Specifically, we set turning-off probability rate is 0.6 for each neuron at the fully-connected layer in each learning iteration, and this dropout approach is applied by default in every experiment.

5 Learning Algorithm

The proposed deep structured model involves three components to be optimized: (i) the latent variables H that manipulate the activity decomposition, (ii) the margin-based classifier $\{\mathbf{w}, b\}$, and (iii) the CNNs' parameters ω. The latent variables are not continuous and need to be estimated adaptively for different input videos, making the standard back propagation algorithm [21] unsuitable for our deep model. In this section, we present a joint component learning algorithm that iteratively optimizes the three components. Moreover, to overcome the problem of insufficient 3D data, we propose to borrow the large amount of 2D videos to pretrain the CNNs' parameters in advance.

5.1 Joint Component Learning

Denote $(X,Y) = \{(x_1,y_1), \ldots, (x_N,y_N)\}$ as the training set with N examples, where x_i is the video, $y_i \in \{1,\ldots,C\}$ denotes the activity category. Denote $H = \{h_1,\ldots,h_N\}$ as the set of latent variables for all training examples. The model parameters to be optimized can be divided into three groups, i.e. H, $\{\mathbf{w}, b\}$, and ω. Fortunately, given any two groups of parameters, the other group of parameters can be efficiently learned using either the stochastic gradient descent (SGD) algorithm (e.g. for $\{\mathbf{w}, b\}$ and ω) or enumeration (e.g. for H).

Therefore, we adopt a principled coordinate type algorithm to optimize the our deep structured model in Eq. (5) and (7). This learning algorithm actually is a general expectation maximization (GEM) method [47], which iteratively performs the E-step and the M-step: the former discovering the optimal latent variables by global searching and the latter optimizing the CNN and classifier parameters for a sub-optimal solution. As shown in [47], such a GEM procedure can converge monotonically to a stationary point. More specifically, our learning algorithm iterates with the three steps: (i) Given the model parameters $\{\mathbf{w}, b\}$ and ω, we estimate the latent variables h_i for each video and update the corresponding feature $\phi(x_i;\omega,h_i)$ (Figure 6 (a)); (ii) Given the updated features $\phi(X;\omega,H)$, we update the max-margin classifier $\{\mathbf{w}, b\}$ (Figure 6 (b)); (iii) Given the model parameters $\{\mathbf{w}, b\}$ and H, we update the CNN parameters ω, which will lead to both the increase of the margin and the decrease of the radius (Figure 6 (c)). It is worth mentioning that the two steps (ii) and (iii) can be performed in the same procedure of SGD, i.e. their parameters are jointly updated in an end-to-end way.

In the following, we explain in detail the three steps for minimizing the loss in Eq. (7), which are derived from our deep model.

(i) Given the model parameters ω and $\{\mathbf{w}, b\}$, for each sample (x_i, y_i), the most appropriate latent variables h_i can be determined by exhaustive searching over all the possible choices,

$$h_i^* = \arg\min_{h_i} 1 - \big(\mathbf{w}\phi(x_i;\omega,h_i) + b\big)y_i. \tag{12}$$

GPU programming is employed to accelerate the searching process. With the updated latent variables, we further obtain the feature set $\phi(X;\omega,H)$ of all the training data.

(ii) Given $\phi(X;\omega,H)$ and the CNNs' parameters ω, batch stochastic gradient descent (SGD) is adopted for updating model parameters in Eq. (7). In iteration t, a batch $B_t \subset (X,Y,H)$ of k samples is chosen. We can obtain the gradients of the max-margin classifier with respect to parameters $\{\mathbf{w}, b\}$,

$$\frac{\partial L_3}{\partial \mathbf{w}} = \mathbf{w} - \lambda \sum_{(x_i,y_i,h_i)\in B_t} y_i \phi(x_i;\omega,h_i) \max\Big(0, 1 - \big(\mathbf{w}^T\phi(x_i;\omega,h_i) + b\big)y_i\Big), \tag{13}$$

$$\frac{\partial L_3}{\partial b} = -2\lambda \sum_{(x_i,y_i,h_i)\in B_t} y_i \max\Big(0, 1 - \big(\mathbf{w}^T\phi(x_i;\omega,h_i) + b\big)y_i\Big), \tag{14}$$

(iii) Given the latent variables H and the max-margin classifier $\{\mathbf{w}, b\}$, based on the gradients with respect to ω, the back propagation algorithm can be adopted to learn CNNs' parameters ω. More specifically, we first update the mean $\bar{\phi}_\omega$ in Eq. (8) based on $\phi(X;\omega,H)$, and then compute the derivative of the relaxed loss in Eq. (7) as

$$\begin{aligned}\frac{\partial L_3}{\partial \omega} &= 4\eta \sum_{(x_i,y_i,h_i)\in B_t} \big(\phi(x_i;\omega,h_i) - \bar{\phi}_\omega\big)^T \frac{\partial \phi(x_i;\omega,h_i)}{\partial \omega} \\ &- 2\lambda \sum \mathbf{w}^T y_i \frac{\partial \phi(x_i;\omega,h_i)}{\partial \omega} \max\Big(0, 1 - \big(\mathbf{w}^T\phi(x_i;\omega,h_i) + b\big)y_i\Big).\end{aligned} \tag{15}$$

By performing the proposed back propagation algorithm, we can further decrease the relaxed loss and optimize the model parameters. During the back propagation, batch SGD

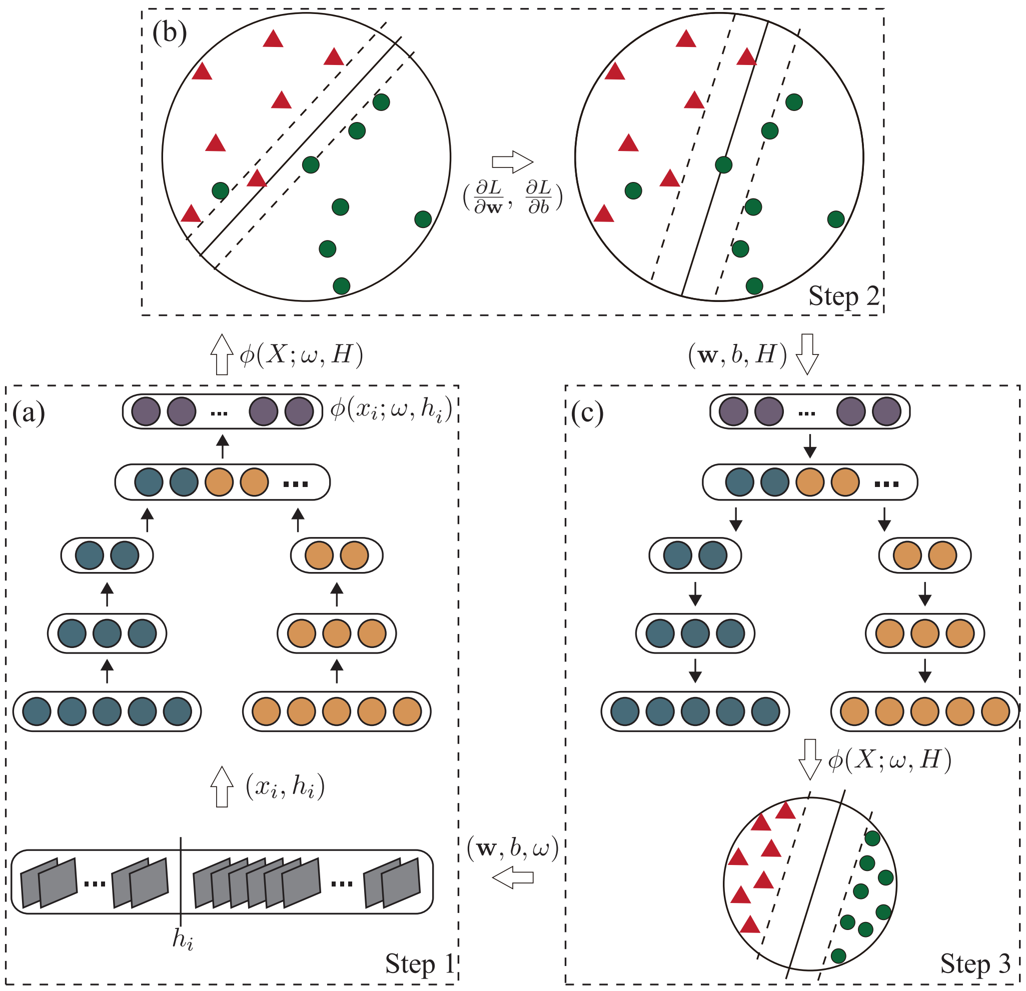


Fig. 6 Illustration for our joint component learning algorithm. It iteratively performs with the three steps: (a) Given the classification parameters $\{\mathbf{w},b\}$ and the CNNs' parameters ω, we estimate the latent variables h_i for each video and generate the corresponding feature $\phi(x_i;\omega,h_i)$; (b) Given the updated features $\phi(X;\omega,H)$ for all training examples, the classifier $\{\mathbf{w},b\}$ is updated via SGD ; (c) Given $\{\mathbf{w},b\}$ and H, back propagation updates the CNNs' parameters ω.

is adopted to simultaneously update the parameters of both step (**ii**) and (**iii**). The optimization algorithm iterates between these three steps until convergence.

5.2 Model Pre-training

Parameter pre-training followed by fine-tuning is an effective method to boost the performance in deep learning, especially when the training data is scarce. In the literature, there are two popular solutions, i.e. unsupervised pre-training on unlabeled data [35] and supervised pre-training for an auxiliary task [12]. The latter usually requires the data formate (e.g. image) for parameter pre-training is exactly the same as that (e.g. image) for fine-tuning.

In our approach, we suggest an alternative solution for 3D human activity recognition. Although collecting RGB-D videos of human activities is expensive, a large amount of 2D activity videos can be easily obtained. Consequently, we first apply the supervised pre-training using a large number of 2D activity videos, and then fine-tune the CNNs' parameters for training the 3D human activity models.

In the step of pre-training, the CNNs' parameters are randomly initialized at the beginning. For each input 2D video, we equally segment it into M parts without estimating its latent variables. Here we simply employ the softmax classifier to pre-train the parameters of CNN, since the softmax loss unbiasedly treat all samples and it is suitable for learning a general feature representation [12].

The 3D and 2D convolutional kernels obtained in pre-training are only for gray channel. Thus, after pre-training, we duplicate the dimension of the 3D convolutional kernels in the first layer and initialize the parameters for the depth channel by the parameters for the gray channel, which allows us to borrow the features learned from the 2D video while directly learning the higher level information from the

Algorithm 1 Learning Algorithm

```
Input:
    The labeled 2D, 3D activity dataset and learning rate $\alpha_{\mathbf{w},b}$, $\alpha_\omega$.
Output:
    Model parameters $\{\omega, \mathbf{w}, b\}$.
Initialization:
    Pre-train the spatio-temporal CNNs using the 2D videos.
Learning on 3D video dataset:
repeat
    1. Estimate the latent variables $H$ for all samples by fixing model parameters $\{\omega, \mathbf{w}, b\}$.
    2. Optimize $\{\mathbf{w}, b\}$ given the CNN model parameters $\omega$ and the input sample segments indicated by $H$:
        2.1 Calculate $\phi(X;\omega,H)$ by forwarding the neural network with $\omega$.
        2.2 Optimize $\{\mathbf{w}, b\}$ via:
            $\mathbf{w} := \mathbf{w} - \alpha_{\mathbf{w},b} * \frac{\partial L}{\partial \mathbf{w}}$ by Eq. (13);
            $b := b - \alpha_{\mathbf{w},\mathbf{b}} * \frac{\partial L}{\partial b}$ by Eq. (14);
    3. Optimize $\omega$ given $\{\mathbf{w}, b\}$ and $H$:
        3.1 Calculate $\kappa_{ij}$, $\kappa_i$ and $\phi_i$ for $L_2$, or calculate $\bar{\phi}_\omega$ for $L_3$.
        3.2 Optimize the parameters $\omega$ of the spatio-temporal CNNs:
            $\omega := \omega - \alpha_\omega * \frac{\partial L}{\partial \omega}$ by Eq. (15).
until $L$ in (5) or (7) converges.
```

specific 3D activity dataset. For the fully connected layer, we set its parameters as random values.

We summarize the overall learning procedure in Algorithm 1.

6 Inference

Given an input video x_i, the inference task aims to recognize its category of the activity, which can be formulated as the minimization of $F_y(x_i, \omega, h)$ with respect to the activity label y and the latent variables h,

$$(y^*, h^*) = \arg\max_{(y,h)} \{F_y(x_i, \omega, h) = \mathbf{w}_y^T \phi(x_i; \omega, h) + b_y\}. \quad (16)$$

where $\{\mathbf{w}_y, b_y\}$ denotes the parameters of the max-margin classifier for the activity category y. Note the possible values for y and h are discrete. Thus the problem above can be solved by searching across all of the labels $y(1 \leq y \leq C)$ and calculate the maximum $F_y(x_i, \omega, h)$ by optimizing h. To find the maximum of $F_y(x_i, \omega, h)$, we enumerate all the possible values of h, and calculate the corresponding $F_y(x_i, \omega, h)$ via forward propagations. Since the forward propagations decided by different h are independent, we can parallelize the computation via GPU to accelerate the inference process.

7 Experiments

To validate the advantages of our model, experiments are conducted on several challenging public datasets, i.e. *CAD-120 Dataset* [19], *SBU Kinect Interaction Dataset* [53], and a larger dataset newly created by us, namely *Office Activity (OA) Dataset*. Moreover, we introduce a more comprehensive dataset in our experiments by combining five existing datasets of RGB-D human activity. In addition to demonstrating the superior performance of the proposed model over other state-of-the-arts, we extensively evaluate the main components of our framework.

7.1 Datasets and Setting

The *CAD-120* dataset comprises of 120 RGB-D video sequences of humans performing long daily activities of 10 categories, and has been widely used for testing 3D human activity recognition methods. These activities recorded via the Microsoft Kinect sensor were performed by four different subjects, and each activity was repeated three times by the same actor. These activities have a long sequence of sub-activities, which vary from subject to subject significantly in terms of length of the sub-activities, order of the sub-activities as well as in the way they executed the task. Moreover, the challenges on this dataset also lie in the large variance in object appearance, human pose, and viewpoint. Several sampled frames and depth maps from this databases of these 10 categories are exhibited in Figure 7 (a).

The *SBU* dataset consists of 8 categories of two-person interaction activities, including a total of about 300 RGB-D video sequences, i.e. about 40 sequences for each interaction category. Even though most interactions in this dataset are simple, it is still challenging for modeling two-person interactions by considering the following difficulties: i) one person is acting and the other person is reacting in most cases, ii) the average frame length of these interaction is short (ranging from 20 to 40), iii) the depth maps have noises. Figure 7 (b) shows several sampled frames and depth maps of these 8 categories.

The proposed *OA* dataset is more comprehensive and challenging compared with the existing datasets, and it covers the regular daily activities taken place in an office. To the best of our knowledge, it is the largest activity dataset of RGB-D videos consisting of 1180 sequences. The *OA* database is publicly accessible [1]. Three RGB-D sensors (i.e. Microsoft Kinect cameras) are utilized to capture data from different viewpoints, and more than 10 actors are involved. The activities are captured in two different offices to increase the variability, where each actor performs the same activity twice. The activities performed by two subjects with interactions are also included. Specifically, it is divided into two subsets, each of which contains 10 categories of activities: *OA1* (complex activities by a single subject) and *OA2* (complex interactions by two subjects). Several sampled frames

[1] `http://vision.sysu.edu.cn/projects/3d-activity/`

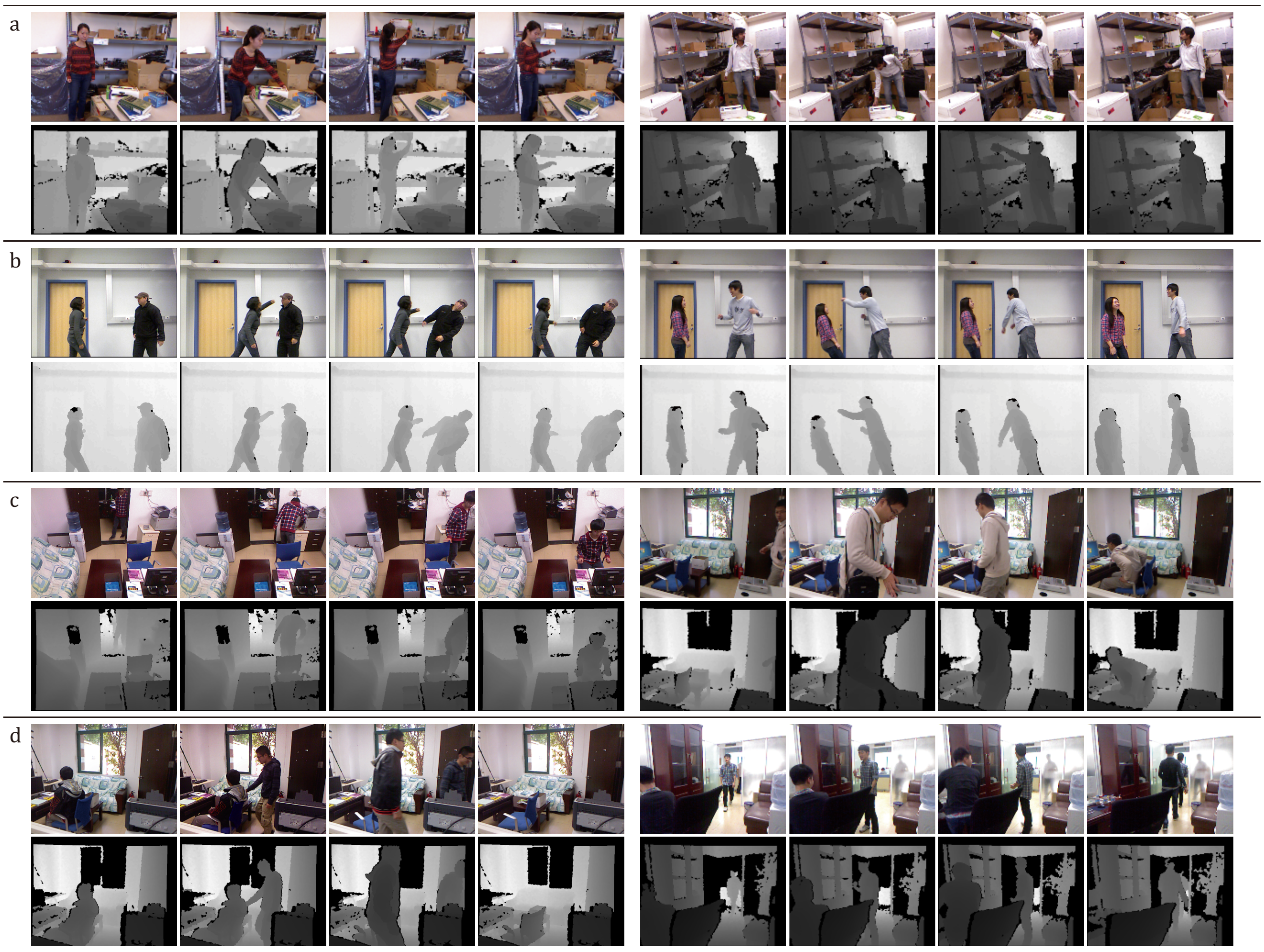


Fig. 7 Activity examples from the testing databases. Several sampled frames and depth maps are presented. (a) *CAD-120*, (b) *SBU*, (c) *OA1*, (d) *OA2*, respectively, show two activities of the same category selected from the three databases.

and depth maps are exhibited in Figure 7 (c) and Figure 7 (d) from *OA1* and *OA2*, respectively.

To evaluate our model under a larger scale scenario, we collect an extra dataset by combining existing RGB-D human activity datasets: *RGBD-HuDaAct* [28], *CAD120*, *SBU*, *UTKinect-Action* [50] and *OA*. This dataset contains 2989 video sequences with 5, 500, 000 frames (approximately 50 hours long) belonging to 50 activity categories, and we name it as *Merged_50* Dataset. Note that we merge very similar activity categories from the different datasets. In addition, we create a coarse-level variant of this dataset by merging the 50 categories into only 4, that is, all of the 2989 activity instances are roughly divided into 4 types: {a person interacting small objects (e.g. answering-phones, having-meal), a person interacting large objects (e.g. sleeping-in-bed, cleaning-objects), physical contacting of persons (e.g. departing, asking-and-way), non - physical contacting of persons (e.g. exchanging objects, hugging objects together)}. And we name this coarse-level dataset as *Merged_4*.

All the experiments are executed on a desktop PC with an Intel i7 4.0GHz CPU, 8GB RAM and GTX 980 GPU. For model learning, Algorithm 1 is employed to learn the CNN ω and the classifier $\{\mathbf{w}, b\}$. The main time-consuming part is the model pre-training, which can take several days based on our desktop PC. Afterwards, the training time of our model on 3D activity datasets is acceptable: 3 hours on CAD-120 (including 120 long videos). Each iteration of training costs similar time, and the convergence of our model over iterations is shown in Figure 8. For inference, with the GPU-based parallel implementation, it only takes around 0.4 seconds to complete recognition on a given video with about 200 frames. For *CAD-120* and *SBU*, we follow the same training/test split adopted in the comparison methods. For *OA1* and *OA2*, we adopt the 5-fold cross validation by ensuring that the subjects in training set are different with those in testing set. Since *Merged_50* and *Merged_4* datasets contain different subjects in different environments, we randomly select 70%, 10%, 20% video sequences from the two datasets for training, validating and testing, respectively.

	[16]	Softmax + CNN	SVM + CNN	R-SVM + CNN	Softmax + LCNN	SVM + LCNN	R-SVM + LCNN
without pre-training	56.4%	61.8%	57.5%	62.5%	70.8%	66.7%	**74.7%**
with pre-training	63.1%	68.3%	78.3%	77.7%	82.7%	89.4%	**90.1%**

Table 1 Average accuracy with/without incorporating the latent structure on *CAD 120* dataset with different top classifiers: Softmax, linear SVM, radius-margin SVM.

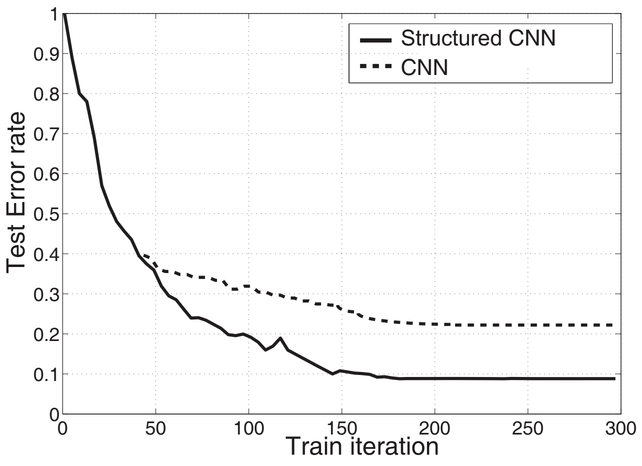


Fig. 8 Test error rates with/without incorporating the latent structure in the deep model. The solid curve represents the deep model trained by the proposed joint component learning method, and the dashed curve represents the traditional training way (i.e. using standard back-propagation).

7.2 Empirical Analysis

Empirical analysis are given to assess the main components of the proposed deep structured model, including latent structure, relaxed radius-margin bound, model pre-training, and depth/grayscale channel. Several variants of our method are suggested by enabling/disabling some components. Specifically, we denote the conventional 3D convolutional neural network with the softmax classifier as Softmax + CNN, denote the 3D CNN with the SVM classifier as SVM + CNN, denote the 3D CNN with the relaxed radius-margin bound classifier as R-SVM + CNN. Analogously, we denote our deep model as LCNN, and then define Softmax + LCNN, SVM + LCNN, and R-SVM + LCNN accordingly.

Latent Model Structure. In this experiment, we implement a simplified version of our model by removing the latent structure and compare it with our full model. The simplified model is actually a spatio-temporal CNN model including both 3D and 2D convolutional layers, and this model uniformly segments the input video into M sub-activities. Without the latent variables to be estimated, the standard back propagation algorithm is employed for model training. We execute this experiment on *CAD120* dataset. Figure 8 shows the test error rates with different iterations of the simplified model (i.e. CNN) and the full version (i.e. structured CNN) in the same CNN initialization. Based on the results, we observe that our full model outperforms the simplified model in both error rate and training efficiency. Furthermore, one can see that the structured models with model pre-training, i.e. Softmax + LCNN, SVM + LCNN, R-SVM + LCNN, achieve 14.4%/11.1%/12.4% better performance than the traditional CNN models, i.e. Softmax + CNN, SVM + CNN, R-SVM + CNN, respectively. The results clearly demonstrate the significance of incorporating latent temporal structure in dealing with the large temporal variations of human activities.

Pre-training. To justify the effectiveness of pre-training, we discard the parameters trained on 2D videos, and learn the model directly on the grayscale-depth data. We compare the performance with/without pre-training using SVM + LCNN and R-SVM + LCNN, as listed in Table 1. One can see that pre-training is effective in reducing the test error rate. Actually, the test error rate with pre-training is about 15% less than that without pre-training.

Relaxed Radius-margin Bound. As described above, the training data for grayscale-depth human activity recognition are scarce. Thus, for the last fully connected layer, we adopt the SVM classifier by incorporating with the relaxed radius-margin bound, resulting in the R-SVM + LCNN model. To justify the role of the relaxed radius-margin bound, Table 3 compares the accuracies of Softmax + LCNN, SVM + LCNN, and R-SVM + LCNN on all datasets with the same experimental settings. It is observed that the max-margin based classifiers (SVM and R-SVM) are particularly effective on small scale datasets (CAD120, SBU, OA1, OA2, Merged_50). On average, the accuracy of R-SVM + LCNN is average 6.5% higher than that of Softmax + LCNN, and is about 1% higher than that of SVM + LCNN. On Merged_4 dataset, the improvement of R-SVM + LCNN is incrementally evident, 1.8% higher than Softmax + LCNN. These results finely accord with our motivation of incorporating the radius-margin bound into our deep learning framework. Moreover, when the model is learned without pre-training, R-SVM + LCNN gains about 4% and 8% improvements over Softmax + LCNN and SVM + LCNN by accuracy, respectively, as Table 1 reports.

Channel Analysis. To evaluate the contribution of the grayscale and depth data, we execute the following experiment on the *OA* datasets: keeping only one channel data as input. Specifically, we first disable the depth channel and input the grayscale data to perform the training/testing, and then disable the grayscale channel and employ the depth channel for training/testing. Table 4 proves that depth data

	[37]	best result from [19]	[49]	[16]	best result from [18]	R-SVM + LCNN
arranging-objects	-	33.0%	75.0%	68.3%	50.0%	**91.7**%
cleaning-objects	-	67.0%	68.3%	60.0%	67.0%	**83.3**%
having-meal	-	**100.0**%	41.7%	60.0%	**100.0**%	91.7%
making-cereal	-	**100.0**%	76.7%	77.6%	**100.0**%	**100.0**%
microwaving-food	-	**100.0**%	36.7%	71.7%	67.0%	**100.0**%
picking-objects	-	75.0%	75.0%	58.3%	67.0%	**91.7**%
stacking-objects	-	**92.0**%	75.0%	48.3%	**92.0**%	91.7%
taking-food	-	75.0%	83.3%	73.3%	67.0%	**91.7**%
taking-medicine	-	**100.0**%	58.3%	76.7%	92.0%	84.6%
unstacking-objects	-	**100.0**%	33.3%	36.7%	92.0%	75.0%
Average accuracy	59.7%	84.2%	62.3%	63.1%	83.1%	**90.1**%

Table 2 Accuracy of all categories on *CAD120* dataset. Accuracy per activity category and average accuracy of all categories are reported.

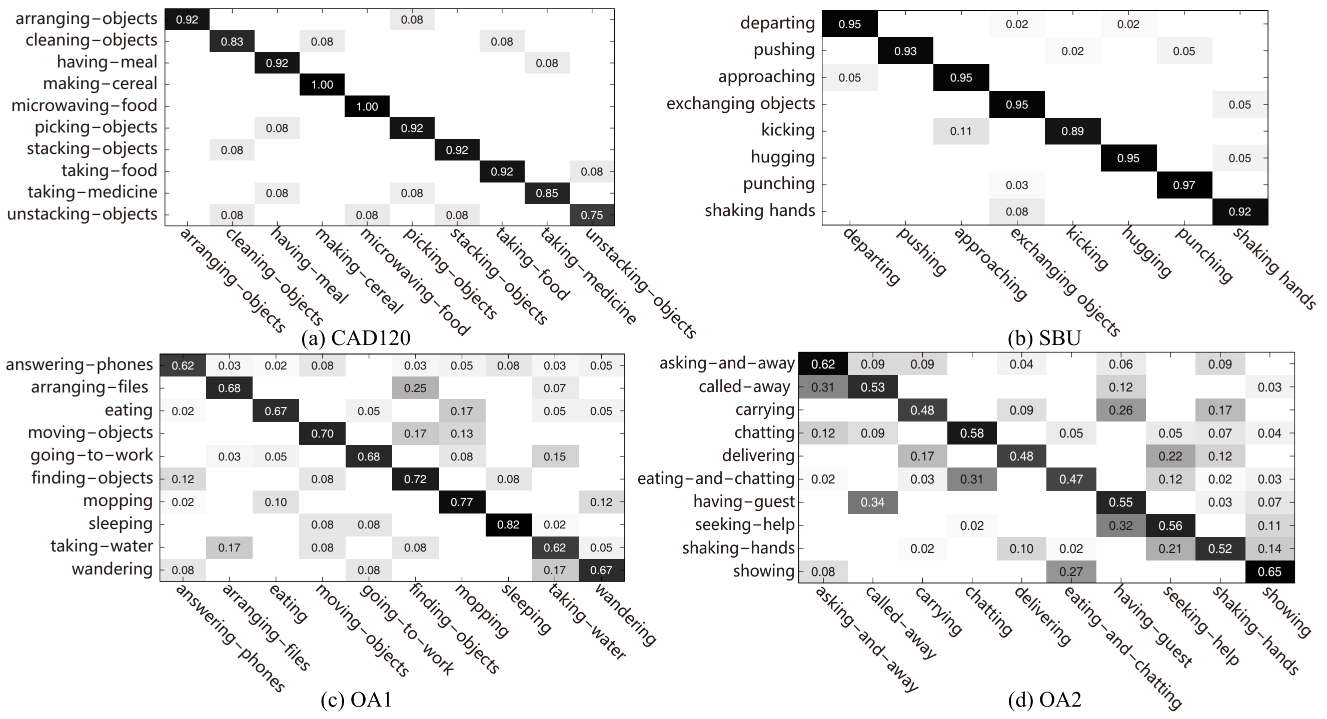


Fig. 9 Confusion Matrices of our proposed deep structured model on (a) *CAD120*, (b) *SBU*, (c) *OA1*, (d) *OA2* datasets. It is evident that these confusion matrices all have a strong diagonal with few errors.

	Softmax + LCNN	SVM + LCNN	R-SVM + LCNN
CAD120	82.7%	89.4%	**90.1**%
SBU	92.4%	92.8%	**94.0**%
OA1	60.7%	68.5%	**69.3**%
OA2	47.0%	53.7%	**54.5**%
Merged_50	30.3%	36.4%	**37.3**%
Merged_4	87.1%	88.5%	**88.9**%

Table 3 Average accuracy of all categories on four datasets with different classifiers.

	grayscale	depth	grayscale + depth
OA1	60.4%	65.2%	**69.3**%
OA2	46.3%	51.1%	**54.5**%
Merged_50	27.8%	33.4%	**37.3**%
Merged_4	81.7%	85.5%	**88.9**%

Table 4 Channel analysis on the three datasets. Average accuracy of all categories are reported.

	Linear SVM	MILBoost	ours
Average accuracy	87.3%	91.1%	**93.4**%

Table 5 Average accuracy on *SBU* dataset.

can boost the performance by large margins, especially in *OA1* and *Merged_50*. This is due to the fact that large appearance variances existed in grayscale data. In particular, our testing is performed on the new subjects and this would further increase the appearance variance. On the contrary, the depth data are more reliable and have much smaller variances, which is helpful in capturing the salient motion information.

7.3 Experimental Results and Comparisons

CAD-120 dataset. On this dataset, we adopt five state-of-the-art methods for comparison. Note that for different methods we train the models using the same data annotation, which only includes the activity labels on videos. As shown in Table 2, our method obtains the average accuracy of 90.1%, which is significantly superior to the results generated by other five competing methods, i.e. 59.7% [37], 84.2% [19], 62.3% [49], 63.1% [16] and 83.1% [18]. Table 2 reports the accuracies per activity category of our method and the method based on hand-crafted feature engineering [49], the deep architecture of convolutional neutral networks [16] [2], and the rich spatio-temporal relations modeling [18]. Our method achieves the highest accuracies on 6 of the 10 activity categories.

SBU dataset. As shown in Table 5, our method obtains the average accuracy of 93.4% and performs better than the methods based on body-pose features [53], which indicates that our method is effective in learning discriminative features directly from raw data for modeling person-to-person interaction.

[2] We implement the 3D-CNN model [16]. For fair comparison, parameter pre-training and dropout have been also employed in our implementation, and the configuration of 3D-CNN is the same with that of our model except that we set $M = 1$ for 3D-CNN.

	[49]	[16]	ours
answering-phones	12.5%	40.0%	**61.7%**
arranging-files	59.7%	53.3%	**68.3%**
eating	40.3%	41.7%	**66.7%**
moving-objects	48.6%	51.7%	**70.0%**
going-to-work	34.7%	41.7%	**68.3%**
finding-objects	65.3%	36.7%	**71.7%**
mopping	63.9%	66.7%	**76.7%**
sleeping	25.0%	45%	**81.7%**
taking-water	58.3%	40.0%	**61.7%**
wandering	56.9%	50.0%	**66.7%**
Accuracy	46.5%	46.7%	**69.3%**

Table 6 Quantitative results on *OA1* dataset. Accuracy per activity category and average accuracy of all categories are reported.

	[49]	[16]	ours
asking-and-away	12.5%	39.6%	**62.3%**
called-away	45.8%	44.8%	**53.5%**
carrying	**66.7%**	56.8%	48.3%
chatting	37.5%	17.2%	**57.9%**
delivering	20.1%	34.5%	**48.3%**
eating-and-chatting	**50.0%**	35.8%	46.6%
having-guest	37.5%	34.1%	**55.2%**
seeking-help	16.7%	44.8%	**56.1%**
shaking-hands	41.7%	32.8%	**51.7%**
showing	37.5%	29.3%	**64.6%**
Accuracy	36.6%	37.0%	**54.5%**

Table 7 Quantitative results on *OA2* dataset. Accuracy per activity category and average accuracy of all categories are reported.

	[49]	[16]	ours
Merged_50	21.1%	24.1%	**37.3%**
Merged_4	79.1%	81.2%	**88.9%**

Table 8 Average accuracy on *Merged_50* and *Merged_4* datasets.

OA dataset. In this experiment, we apply our method on the two *OA* subsets. Tables. 6 and 7 list the accuracies per category and average accuracy of the competing methods, and our method outperforms the state-of-the-art methods in terms of the average accuracy. On the *OA1* set, our method achieves the best accuracies on all categories and obtains the highest average accuracy of 69.3%, as shown in Table. 6. On the *OA2* set, our method achieves the best accuracies on 8 out of 10 activity categories and obtains the highest average accuracy of 54.5%, as shown in Table 7. By checking the results, we find that the failure cases are mainly caused by the lack of contextualized scene understanding. For example, understanding the activities of *having-guest* and *eating-and-chatting* actually requires extra higher level information, and we will consider this issue in the future work.

Merged datasets. Table 8 reports the average accuracy of the competing methods, and our method outperforms the state-of-the-art methods [49, 16] in terms of the average accuracy.

In summary, our method consistently achieves better results than the competing methods on the three 3D activity datasets. Figure 9 shows the confusion matrices of our model for all datasets. One can see that, the confusion matrices are strongly diagonal with few errors, which indicates that our deep structured model is effective in handling various challenges in 3D human activity recognition.

8 Conclusion

In this paper, we have introduced, first, a deep and latent-structured model using the convolutional neural networks. Second, a unified formulation integrating the radius-margin regularization with the feature learning. Third, an effective learning algorithm that iteratively optimizes the sub-activity decomposition, the margin-based classifier, and the neural networks. We have demonstrated the practical applicability of our model by effectively recognizing human activities using a depth camera. Experiments on the public datasets suggest that our model convincingly outperforms other state-of-the-art methods under several very challenging scenarios.

One main drawback of our current solution is the scalability of model inference. The brute-force enumeration over all settings of the latent variables will cause extra computation cost and this issue may become much more serious when the number (e.g., 1000) of human activity categories is large. Apart from the scalability issue, we intend to extend our work in the following directions. The first is to generalize our model with compositional grammar rules (e.g. the And-Or grammars), and thus deal with more complicated event understanding (e.g. the causality inference). The second is to revise our neural network for recognizing human action / activity from 2D videos. Note that there are distinct differences between 2D videos and 3D videos. For example, these mentioned 2D datasets basically include diverse environments (e.g., indoor / outdoor) with the camera moving, and the 3D depth data are all captured indoor with a fixed sensor (i.e. Microsoft Kinects). In addition, the 2D videos are usually in higher resolution than the data (i.e. 320×240) captured by the depth sensor.

Acknowledgment

This work was supported in part by the Hong Kong Scholar Program, and in part by the HK PolyU's Joint Supervision Scheme with the Chinese Mainland, Taiwan and Macao Universities (Grant no. G-SB20), in part by Guangdong Natural Science Foundation (Grant no. s2013010013432), and in part by Guangdong Science and Technology Program (Grant no. 2013B010406005).

References

1. Amer MR, Todorovic S (2012) Sum-product networks for modeling activities with stochastic structure. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1314–1321
2. Bayer J, Osendorfer C, Korhammer D, Chen N, Urban S, van der Smagt P (2014) On fast dropout and its applicability to recurrent networks. In: *International Conference on Learning Representations (ICLR)*
3. Brendel W, Todorovic S (2011) Learning spatiotemporal graphs of human activities. In: *IEEE Conference on Computer Vision (ICCV)*, pp 778–785
4. Chapelle O, Vapnik V, Bousquet O, Mukherjee S (2002) Choosing multiple parameters for support vector machines. *Machine Learning* 46(1-3):131–159
5. Chaquet JM, Carmona EJ, Fernndez-Caballero A (2013) A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding* 117(6):633–659
6. Cheng Z, Qin L, Huang Q, Jiang S, Yan S, Tian Q (2011) Human group activity analysis with fusion of motion and appearance information. In: *ACM international conference on Multimedia (ACM MM)*, pp 1401–1404
7. Chung KM, Kao WC, Sun CL, Wang LL, Lin CJ (2003) Radius margin bounds for support vector machines with the rbf kernel. *Neural Computation* 15(11):2643–2681
8. Do H, Kalousis A (2013) Convex formulations of radius-margin based support vector machines. In: *International Conference on Machine Learning (ICML)*
9. Do H, Kalousis A, Hilario M (2009) Feature weighting using margin and radius based error bound optimization in svms. In: *European Conference on Machine Learning and Knowledge Discovery in Databases: Part I*, pp 315–329
10. Donahue J, Hendricks LA, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*
11. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32(9):1627–1645
12. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 580–587
13. Gupta R, Chia AYS, Rajan D (2013) Human activities recognition using depth images. In: *ACM international conference on Multimedia (ACM MM)*, pp 283–292
14. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507
15. Huang FJ, LeCun Y (2006) Large-scale learning with svm and convolutional for generic object categorization. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 284–291
16. Ji S, Xu W, Yang M, Yu K (2013) 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 35(1):221–231
17. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*
18. Koppula HS, Saxena A (2013) Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation. In: *International Conference on Machine Learning (ICML)*, pp 792–800
19. Koppula HS, Gupta R, Saxena A (2013) Learning human activities and object affordances from rgb-d videos. *International Journal of Robotics Research (IJRR)* 32(8):951–970
20. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*,

pp 1097–1105
21. LeCun Y, Boser B, Denker J, Henderson D, Howard R, Hubbard W, Jackel L, Henderson D (1990) Handwritten digit recognition with a back-propagation network. In: *Advances in neural information processing systems*
22. Liang X, Lin L, Cao L (2013) Learning latent spatio-temporal compositional model for human action recognition. In: *ACM international conference on Multimedia (ACM MM)*, pp 263–272
23. Lin L, Wu T, Porway J, Xu Z (2009) A stochastic graph grammar for compositional object representation and recognition. *Pattern Recognition* 42(7):1297–1307
24. Lin L, Wang X, Yang W, Lai JH (2015) Discriminatively trained and-or graph models for object shape detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 37(5):959–972
25. Luo P, Wang X, Tang X (2013) A deep sum-product architecture for robust facial attributes analysis. In: *IEEE Conference on Computer Vision (ICCV)*, pp 2864–2871
26. Luo P, Wang X, Tang X (2013) Pedestrian parsing via deep decompositional neural network. In: *IEEE Conference on Computer Vision (ICCV)*, pp 2648–2655
27. Luo P, Tian Y, Wang X, Tang X (2014) Switchable deep network for pedestrian detection. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*
28. Ni B, Wang G, Moulin P (2013) Rgbd-hudaact: A color-depth video database for human daily activity recognition. In: *Consumer Depth Cameras for Computer Vision*, Springer London, pp 193–208
29. Ni B, Y Pei ZL, Lin L, Moulin P (2013) Integrating multi-stage depth-induced contextual information for human action recognition and localization. In: *International Conference and Workshops on Automatic Face and Gesture Recognition*, pp 1–8
30. Oreifej O, Liu Z (2013) Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 716–723
31. Packer B, Saenko K, Koller D (2012) A combined pose, object, and feature model for action understanding. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1378–1385
32. Pei M, Jia Y, Zhu S (2011) Parsing video events with goal inference and intent prediction. In: *IEEE Conference on Computer Vision (ICCV)*, pp 487–494
33. Sadanand S, Corso JJ (2012) Action bank: A high-level representation of activity in video. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1234–1241
34. Scovanner P, Ali S, Shah M (2007) A 3-dimensional sift descriptor and its application to action recognition. In: *ACM international conference on Multimedia (ACM MM)*, pp 357–360
35. Sermanet P, Kavukcuoglu K, Chintala S, LeCun Y (2013) Pedestrian detection with unsupervised multi-stage feature learning. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 3626–3633
36. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: A simple way to prevent neural networks from overtting. *The Journal of Machine Learning Research* 15(1):1929–1958
37. Sung J, Ponce C, Selman B, Saxena A (2012) Unstructured human activity detection from rgbd images. In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp 842–849
38. Tang K, Fei-Fei L, Koller D (2012) Learning latent temporal structure for complex event detection. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1250–1257
39. Tu K, Meng M, Lee MW, Choi T, Zhu S (2014) Joint video and text parsing for understanding events and answering queries. *IEEE Transactions on Multimedia* 21(2):42–70
40. Vapnik V (1998) Statistical learning theory. *New York: John Wiley and Sons*
41. Venugopalan S, Xu H, Donahue J, Rohrbach M, Mooney R, Saenko K (2015) Translating videos to natural language using deep recurrent neural networks. In: *North American Chapter of the Association for Computational Linguistics*
42. Wang C, Wang Y, Yuille AL (2013) An approach to pose-based action recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 915–922
43. Wang J, Wu Y (2013) Learning maximum margin temporal warping for action recognition. In: *IEEE Conference on Computer Vision (ICCV)*, pp 2688–2695
44. Wang J, Liu Z, Wu Y, Yuan J (2012) Mining actionlet ensemble for action recognition with depth cameras. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1290–1297
45. Wang K, Wang X, Lin L, Wang M, Zuo W (2014) 3d human activity recognition with reconfigurable convolutional neural networks. In: *ACM international conference on Multimedia (ACM MM)*
46. Wang Y, Mori G (2011) Hidden part models for human action recognition: Probabilistic vs. max-margin. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 33(7):1310–1323
47. Wu CFJ (1983) On the convergence properties of the em algorithm. *Annals of Statistics* 11(1):95–103
48. Wu P, Hoi S, Xia H, Zhao P, Wang D, Miao C (2013) Online multimodal deep similarity learning with application to image retrieval. In: *ACM international conference on Multimedia (ACM MM)*, pp 153–162

49. Xia L, Aggarwal J (2013) Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 2834–2841
50. Xia L, Chen CC, Aggarwal JK (2012) View invariant human action recognition using histograms of 3d joints. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp 20–27
51. Yang X, Zhang C, Tian Y (2012) Recognizing actions using depth motion maps-based histograms of oriented gradients. In: *ACM international conference on Multimedia (ACM MM)*, pp 1057–1060
52. Yu K, Lin Y, Lafferty J (2011) Learning image representations from the pixel level via hierarchical sparse coding. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1713–1720
53. Yun K, Honorio J, Chattopadhyay D, Berg TL, Samaras D (2012) Two-person interaction detection using body-pose features and multiple instance learning. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*
54. Zhao X, Liu Y, Fu Y (2013) Exploring discriminative pose sub-patterns for effective action classification. In: *ACM international conference on Multimedia (ACM MM)*, pp 273–282
55. Zhou X, Zhuang X, Yan S, Chang SF, Johnson MH, Huang TS (2009) Sift-bag kernel for video event analysis. In: *ACM international conference on Multimedia (ACM MM)*, pp 229–238
56. Zhu S, Mumford D (2006) A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision* 2(4):259–362