IET Computer Vision

Research Article

IET Journals
The Institution of Engineering and Technology

Continuous action segmentation and recognition using hybrid convolutional neural network-hidden Markov model model

ISSN 1751-9632
Received on 21st October 2015
Revised 21st December 2015
Accepted on 13th January 2016
E-First on 19th February 2016
doi: 10.1049/iet-cvi.2015.0408
www.ietdl.org

Jun Lei[1] ✉, *Guohui Li*[1], *Jun Zhang*[1], *Qiang Guo*[1], *Dan Tu*[1]

[1]*College of Information Systems and Management, National University of Defense Technology, Changsha, People's Republic of China*
✉ *E-mail: leijun1987@gmail.com*

Abstract: Continuous action recognition in video is more complicated compared with traditional isolated action recognition. Besides the high variability of postures and appearances of each action, the complex temporal dynamics of continuous action makes this problem challenging. In this study, the authors propose a hierarchical framework combining convolutional neural network (CNN) and hidden Markov model (HMM), which recognises and segments continuous actions simultaneously. The authors utilise the CNN's powerful capacity of learning high level features directly from raw data, and use it to extract effective and robust action features. The HMM is used to model the statistical dependences over adjacent sub-actions and infer the action sequences. In order to combine the advantages of these two models, the hybrid architecture of CNN-HMM is built. The Gaussian mixture model is replaced by CNN to model the emission distribution of HMM. The CNN-HMM model is trained using embedded Viterbi algorithm, and the data used to train CNN are labelled by forced alignment. The authors test their method on two public action dataset Weizmann and KTH. Experimental results show that the authors' method achieves improved recognition and segmentation accuracy compared with several other methods. The superior property of features learnt by CNN is also illustrated.

1 Introduction

Recognising human actions from videos are a crucial problem in a variety of computer visual applications, such as camera surveillance, content-based video retrieval and human-computer interaction. As far as we can see, a large majority of existing approaches focus on solving isolated action recognition problem, where they assume that the boundaries of these individual actions are given in advance. Segmentation is pre-processed to partition video into parts, and the action recognition turns to a classification problem which assigns each part an action label. While in realistic situations, the boundaries of individual actions in video are unknown. In this paper, we investigate a continuous action recognition problem, which is to recognise a sequence of actions. This kind of problem is more challenging due to the following reasons. First, the high variability of articulated motion, movement combination, viewpoint and possible occlusions make it very difficult to represent the actions. Second, what is more complicated is that we have to simultaneously segment and recognise a sequence of actions, such as walking, boxing and running which are performed by one person in a video.

A lot of works have developed many novel handcrafted features for action representation. Compared with global features, such as motion energy image [1], histogram of orient gradient (HOG) [2] and spatio-temporal volume [3], the local features are more invariant to changes in viewpoint, partial occlusion and person appearance. The local features are usually constructed by extracting sets of space–time interest points, which are then combined into a fixed-sized description. The three-dimensional (3D) Harris space–time point feature with local HOG and histogram of flow (HOF) descriptors [4] is likely to be the most used local feature. These handcrafted features are especially designed for special task. However, it is difficult to know which feature is more proper in practical application because the choice of feature is highly problem dependent. Recently, deep models [5] have achieved great success in many applications of computer vision. Convolutional neural network (CNN) [6, 7], one type of deep models, has been demonstrated as an effective model to learn high-level features directly from raw data. CNN was initially applied to process 2D images. Afterwards, Ji *et al.* [8] extended 2D CNN to 3D CNN for the application of isolated action recognition in video, and achieved superior performance without relying on handcrafted features.

The other critical issue is the modelling of continuous action. Action detection approaches [9, 10] find the occurrences of defined actions by searching for the subvolume of video that has high score, without explicitly segmenting the video. Though action detection approaches can be used to track continuous action recognition problem to some extent, they detect each action independently and detected actions may overlap each other. Most current approaches, including these based on generative models (e.g. Markov chain model [11, 12], dynamic time warping [13]) or discriminative models [e.g. support vector machine (SVM) [14]], treat continuous action recognition as sequence problem, addressing joint segmentation and recognition of continuous actions. These approaches consider characteristics within segments and interactions between them, and use dynamic programming algorithms to infer the segmentation and recognition result. Hidden Markov model (HMM) which can model action's temporal dynamics is widely used for analysing human action. HMM was early successfully applied in continuous speech recognition which is much similar to our continuous action recognition problem.

In this paper, we propose a novel joint segmentation and recognition framework for continuous action recognition combining HMM and CNN. The motivation for combining CNN and HMM is the complementary modelling capacities of them on the problem of continuous action recognition. The CNN is typically used as a static model. In order to apply CNN to sequence problem, we introduce a temporal dynamical model – HMM. The CNN is used as a feature extractor. Efficient and robust action features are learnt through CNN directly from raw video data. In this way, all the knowledge of the feature extraction model is from the data, and consequently this model is more generalised. The HMM is used to deal with the temporal variations of continuous action, and it simultaneously performs the segmentation and recognition. We train our CNN-HMM model in alternative way. In order to train CNN, it is unrealistic to segment the actions manually when the amount of data is huge. Instead, states of clips are inferred by forced HMM alignment and these states are used to label the clips to train CNN.

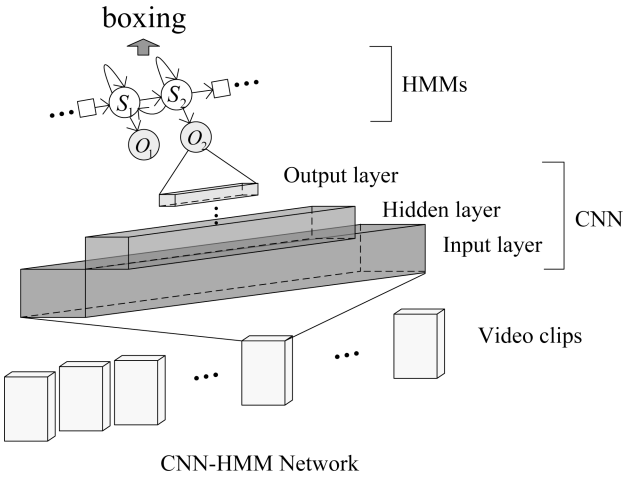

Fig. 1 *Architecture of our hybrid CNN-HMM model. This figure illustrates the recognition process of continuous action. The video is partitioned to clips by slicing window. The CNN is used to extract each clip's features and output the posterior probability of each state. After the posterior probability is scaled, it is used as the emission probability of HMM. The HMM is used to model continuous action and infer the segmentation and recognition result*

Fig. 1 illustrates the architecture of our model. The CNN and HMM are combined in a hierarchical structure. A window slides over the entire video, generating a sequence of overlapping small video clips. We model the video clips as observations generated by the states of HMM. A CNN is applied on each video clip to extract features and output the posterior probability of each state. After the posterior probability is rescaled, it is then used as the emission probability of HMM. The HMM finds the most likely action labels of the video clips sequence, thus obtains the segmentation and recognition result.

We organise the rest of the paper as follows. In Section 2, we give a brief survey of related works in this field. We give the formulation of continuous action recognition problem under HMM framework in Section 3. The proposed CNN-HMM model is described in Section 4. Experiments are presented and analysed in Section 5. We conclude the paper in Section 6.

2 Related works

Deep learning models have been applied in action recognition in recent years. Ji *et al.* [8] developed a 3D CNN to extract spatial and temporal features from video data for action recognition. They connected auxiliary handcrafted features at the last hidden layer of CNN to regularise the model. A linear classifier was applied in the end. Karpathy *et al.* [15] investigated four connectivity ways of CNN in time domain to fuse time information, and proposed architecture for processing multi-resolution streams to speed up the training. As its complexity, the 3D CNN is limited to a small number of continuous frames, which are sub-sequence of an action. The action's evolution over time is ignored. To solve this problem, Baccouche *et al.* [16] used 3D CNN to extract features of each sub-sequence and trained a recurrent neural network to classify the entire sequence. Ng *et al.* [17] proposed two classes of CNN architectures: convolutional temporal feature pooling and long short-term memory architectures for video classification, which can also be used to classify action video. The architectures learn features and combine them across a video over longer time periods. Besides CNN model, other deep models based on independent subspace analysis [18] and sparse auto-encoder [19] were also introduced in action recognition. However, those works are specific to isolated action.

Continuous action recognition approaches take sequential information into consideration. Template-based approach proposed by Kulkarni *et al.* [13] defined the representation of an action category by an action template, and the continuous action was simultaneously segmented and recognised by one-pass or two-pass dynamic frame warping. Large margin-based discriminative models are extended to solve sequence problem. Hoai *et al.* [14] proposed a multi-class SVM-based method to find the globally optimal temporal segmentation and class labels. Shi *et al.* [20] proposed an SVM-based approach under semi-Markov model framework. Most general approaches are based on state-space model. Wang *et al.* [21] proposed an improved switching linear dynamic system with substructure transition model and discriminative boundary model embedded. Ning *et al.*[12] added a latent pose estimator layer to CRF, which converted the high dimensional observations into more compact and informative representations. The HMM is a class of important dynamic state-space model for continuous action recognition. Lv and Nevatia [22] proposed an HMM-based framework for continuous action recognition. They set emission probability as the matching score of silhouette and 3D action template, and assumed a uniform transitional probability. Guenterberg *et al.*[23] used HMM in body sensor networks to recognise continuous action.

Our idea of combining CNN and HMM has been already applied in street view house number recognition in our earlier work [24]. The closest work to ours was presented in [25]. The authors also combined HMM with a deep model. However, they handled 3D skeletal data and they used Gaussian restricted Boltzmann machine to learn high level features.

3 Problem formulation

Let a video be represented as a sequence of clips $\boldsymbol{V} = (v_1, \ldots, v_t, \ldots, v_T)$, which is cropped by slicing window with the width of d and stride of s. v_t is the tth video clip and has d frames. T is the total number of clips in the video. Action features are extracted from clips, which are denoted by $\boldsymbol{O} = (\boldsymbol{o}_1, \ldots, \boldsymbol{o}_t, \ldots, \boldsymbol{o}_T)$, where $\boldsymbol{o}_t$ is the feature vector extracted from the tth video clip. We suppose there are K possible action categories, and the action label set is denoted as Ω. In our setting, for clips sequence $\boldsymbol{V}$ where the unknown action labels are in an unknown order, the continuous action recognition problem is to partition it into J sub-sequences as $\boldsymbol{SQ} = (\mathrm{sq}_{t_0:t_1}, \ldots, \mathrm{sq}_{t_{j-1}:t_j}, \ldots, \mathrm{sq}_{t_{J-1}:t_J})$ and recognise the action labels sequence $\boldsymbol{L} = (l_{t_1}, \ldots, l_{t_j}, \ldots, l_{t_J})$, where $\mathrm{sq}_{t_{j-1}:t_j}$ is the sub-sequence $(v_{t_{j-1}}, \ldots, v_{t_j})$ whose action label is denoted by $l_{t_j} \in \Omega$. t_{j-1}, t_j are start and end time of this sub-sequence where $t_0 = 1$, $t_J = T$.

We use continuous observation HMM to model this sequence problem. Each category of action is modelled by an HMM. Different phases in an action correspond to the hidden states of HMM. The progress of an action can be viewed as transition between hidden states. The modelling of continuous action is to connect these HMMs together in sequence. The transition from one action to the other is actually the transition from one HMM to another. The action features of video clips are treated as the observations generated by the hidden states during the Markov process. Note that the combined HMMs can be considered as one composite HMM.

Define K HMMs for actions as M_l, corresponding to action $l \in \Omega$. For each HMM M_l, the model changing state from s_i to s_j is governed by the discrete transition probability $a_{ij} = p(s_j|s_i)$. The observation $\boldsymbol{o}_t$ is generated from the emission probability distribution $b_j(\boldsymbol{o}_t) = p(\boldsymbol{o}_t|s_j)$. After these probability parameters are trained, continuous action recognition is to find a sequence of labels $\widehat{\boldsymbol{L}}$ that makes the highest log probability based on maximum a posterior rule, given as

$$\widehat{\boldsymbol{L}} = \underset{\boldsymbol{L}, 1 \le J \le T}{\arg\max} \log P(M_{l_{t_1}} M_{l_{t_2}}, \ldots, M_{l_{t_J}} | \boldsymbol{O}) \tag{1}$$

This is done by finding a path through the composite HMM which has the highest probability using the token passing algorithm [26]. Segmentation is implicitly achieved within this inference process.

We build 3D CNN to extract features from the video clips, which is introduced in Section 4.1. The detailed process of modelling continuous actions is given in Section 4.2.

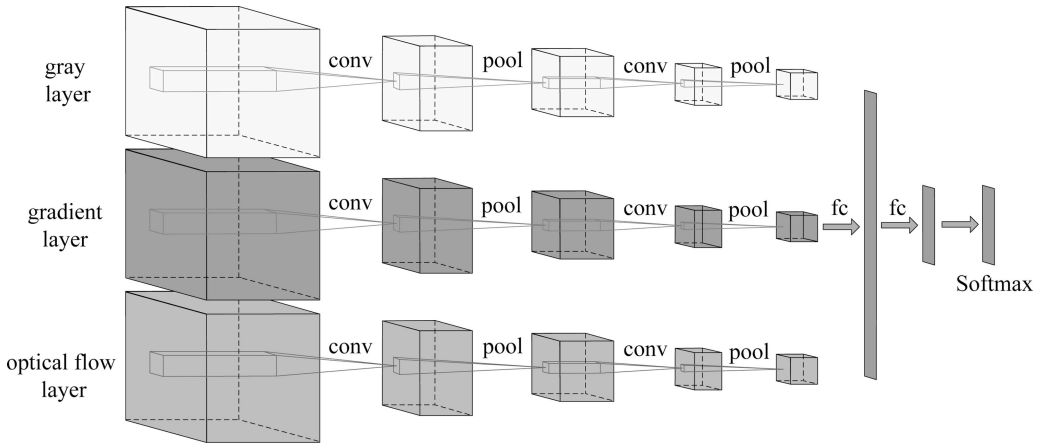

Fig. 2 *Architecture of our 3D CNN*

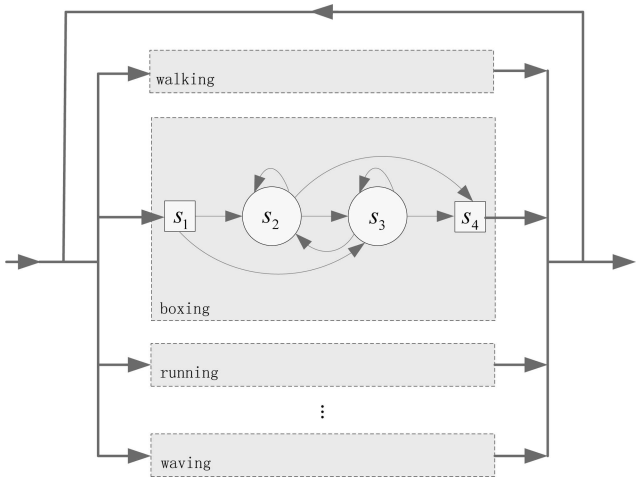

Fig. 3 *Composite HMM for continuous action. The circle nodes denote emitting hidden states, and the square nodes denote entry and exit states. The arrows between nodes indicate the possible transition between states*

4 Proposed hybrid CNN-HMM model

4.1 3D CNN

When extending CNN to apply on video analysis application, we have to consider a proper connection structure to capture motion information from continuous frames. The architecture of our 3D CNN is shown in Fig. 2. For a video clip v_t, its frames number is d. We design an $n \times n \times d$ 3D kernel which connects $n \times n$ size local neighbourhood in spatial dimension and d continuous frames in temporal dimension. The convolutional layer is achieved by convolving this 3D kernel to the clip, and the weight of 3D kernel is shared across the entire clip. The 3D kernel can be viewed as a feature extractor that computes one type of feature. We use multiple distinct 3D kernels to generate different types of features. Then, we stack these different feature maps together to a new cube. A sub-sampling layer follows and the size of max pooling window is $s \times s$. Multiple groups of convolutional layer and sub-sampling layer are stacked.

Following the thought of adding hardwired layers to encode prior knowledge on features in [8], we input two channels as gradient and optical flow to our network as well, besides raw grey channel. The gradient layer is obtained by computing the magnitude of gradients on each frame, and the frames number is d. The optical flow layer is obtained by computing the magnitude of optical flow fields from adjacent frames, and the frames number is $d-1$. Gradient, optical flow and raw grey layers are processed through several groups of convolutional layer and sub-sampling layer, respectively. The final outputs of them are combined to a vector, which is fed to two fully connection layers. The second fully connection layer has the same number of nodes as the number of action categories. The top layer is a softmax layer that outputs the classification probability. The output of the first connection layer is the feature vector $\boldsymbol{o}_t$ learnt by CNN.

Given the action labels of the training video clips, the 3D CNN can be trained by back-propagation (BP) algorithm [6].

4.2 Composite HMM for continuous actions

The modelling of continuous action is to construct a composite HMM which connects HMMs together in sequence, and each HMM corresponds to one category of action. For an N_l-state HMM M_i, in order to connect HMMs, two non-emitting states are added as the entry state and exit state, which are denoted by s_1 and s_{N_l+2}, respectively. In addition, in order to connect HMMs, two non-emitting states are added as the entry state and exit state. Compared with emitting hidden states which cost one unit time to pass through, the entry and exit states do not cost time.

Fig. 3 shows the composite HMM for modelling continuous action. We use two-state HMM to model each action in the experiment. The two-state HMM in Fig. 3 illustrates our design details for an action. An action begins at entry state and ends at exit state. The transition between actions is achieved by moving from one action's exit state to another one's entry state. Transition from state to itself is added to handle different action performance speed. The action behaviour can either move to other state or stay at current state, leading to different performance speed. We add transition from the last emitting state s_3 to the first emitting state s_2, thus the repetitive performance of an action can be modelled. We also add transitions from s_1 to s_3 and from s_2 to s_4, thus an action can start and end at arbitrary states. This design can deal with the situation where the video of an action is incomplete. The parameters of the composite HMM can be trained by applying the Baum–Welch algorithm [27]. In this algorithm, all HMM models are trained in parallel. The transition probabilities between HMMs are set uniform, thus parameters between different HMMs are neglected. We use this two-state HMM for Gaussian mixture model (GMM)-HMM and CNN-HMM in our experiment.

4.3 Hybrid CNN-HMM model

After the CNN and composite HMM are built, the next problem is how to combine them together. Traditional HMM often uses GMM to estimate emission probability $p(\boldsymbol{o}_t|s_j)$. We replace the GMM by CNN to estimate the emission probability in our work. A generative model as GMM is difficult at modelling high dimensional data. However, the CNN, as discriminative model, does not need to know the distribution of data and is better at discriminating between classes.

Noted that the output of the softmax layer is the posterior probability $p(s_j|\boldsymbol{o}_t)$. Using Bayes' rule, we have

$$p(\boldsymbol{o}_t|s_j) = \frac{p(s_j|\boldsymbol{o}_t)P(\boldsymbol{o}_t)}{p(s_j)} \tag{2}$$

Algorithm 1

1 CNN and HMM initialization: use a small number of labelled video lips (v_t, l) to initialize CNN, where v_t is labelled to sc_l. The transition probabilities are initialized randomly, and the emission probabilities are computed using the scaled likelihood $p(sc_l \mid o_t) / p(sc_l)$, where $p(sc_l \mid o_t)$ is computed by CNN and $p(sc_l)$ is initialized to be uniform.

2 Train the composite HMM on the training dataset.

3 Recognize the actions using Token Passing algorithm and evaluate the recognition accuracy A_0.

4 **while** $| A_t - A_{t-1} | > \Delta$ **do**

5 Given the action labels and their order of training video data, assign each video clip a tied state by forced alignment, and label them with corresponding action class.

6 Fine-tune CNN using BP algorithm. Denote the CNN as cnn_t

7 Estimate the prior probability $p(sc_l) = n(sc_l) / n$, where $n(sc_l)$ is the number of video clips assigned with tied state sc_l and n is the total number of video clips.

8 Use CNN to get each tied state's posterior probability $p(sc_l \mid o_t)$ and compute the scaled likelihood $p(sc_l \mid o_t) / p(sc_l)$.

9 Re-estimate the transition probability HMM by Baum-Welch algorithm. Denote the updated composite HMM as hmm_t.

10 Recognize the actions using Token Passing algorithm and evaluate the recognition accuracy A_t.

11 $t \leftarrow t + 1$

12 **end**

13 Output the final CNN-HMM model.

Fig. 4 *Embedded Viterbi algorithm to train CNN-HMM*

where $P(s_j)$ is the prior probability of each state that we can estimate from the training set. $p(\boldsymbol{o}_t)$ is independent of action label and thus can be ignored. Finally, the emission probability used in our CNN-HMM model is approximated by $p(s_j|\boldsymbol{o}_t)/p(s_j)$, which is called scaled likelihood [28].

When using a generative model as GMM, diving feature space into finer categories can make GMM easier to represent the feature space. However, this is not true for a discriminative model as CNN. In order to make CNN more invariant to intra variations of action while keeping discriminative to extra action variations, it is better to treat variations of an action as one category. Therefore, we represent the hidden states of each action by one class. This is implemented by tying the N_l-state HMM M_l to 1-state HMM. These N_l states are tied to state sc_l and share the same parameters. Video clips assigned with tied state labels are used to fine-tune the CNN. Actually, the two-state HMM described in Section 4.2 can also be replaced by a simple one-state HMM. The one-state HMM is almost identical to two-state HMM, as we tie these states to one-state.

The CNN-HMM model is trained using embedded Viterbi algorithm [29]. The algorithm is summarised in Algorithm 1 (see Fig. 4). The CNN and HMM are updated alternatively until required convergence is achieved. Given the action labels and their order in the video data, the forced alignment labels each clip with a tied state label, avoiding the need to manually segment and label the video clips. After the CNN is trained, the transition parameters are re-estimated. This process is repeated until there is no improvement of the accuracy. The initialisation of CNN and HMM has important effect on the convergence speed and performance. In order to get good initialisation values, we use a small number of labelled video clips to train the CNN.

4.4 Computational complexity

At the training phase, learning CNN using BP algorithm is time-consuming. It needs repeated iteration of forward and backward propagations through multiple network layers. Graphics processing unit (GPU) is usually used to accelerate this process. While at the inference phase, the recognition speed is faster than real-time. As a result, the computational complexities of CNN's one forward propagation and Viterbi algorithm are $O(T)$ and $O(c^2T)$, respectively, where c is the number of hidden states and T is the number of video clips.

5 Experiments

In this section, we test our method on two datasets: Weizmann [3] and KTH [30]. Following [14], we manually generate long continuous action sequences by concatenating the isolated actions. We also test isolated action recognition result on KTH dataset. The persons in videos are detected using deformable parts model (DPM) detector [31] and tracked using convolutional restricted Boltzmann machine (CRBM) tracker [32]. The person image is rescaled to uniform size. For all actions in our experiment, we use the two-state HMM described in Section 4.1 to model them. Note that we tie the two-state HMM to one-state HMM in our CNN-HMM method. Our experiment is made on a PC with Intel Xeon E3-1230 (3.3 GHz 8 core CPU, 16 G RAM) and NVIDIA GPU GTX780.

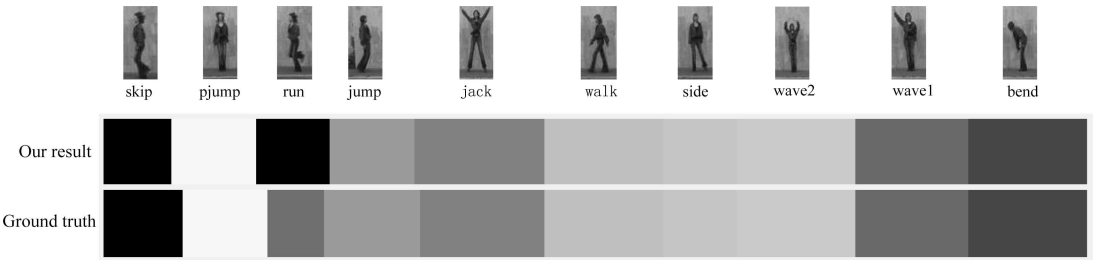

Fig. 5 *Presentation of segmentation and recognition result compared with ground truth. Typical images of actions are shown in the top row. Our segmentation and recognition result and the ground truth are clearly shown by the colour bars in the second and third rows, respectively. Each colour corresponds to a category of action*

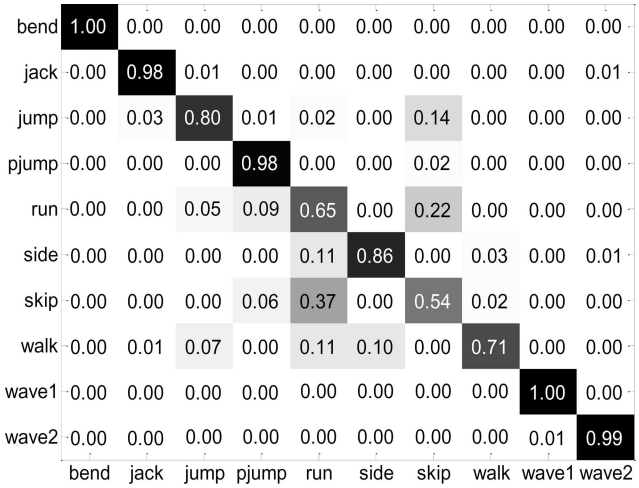

Fig. 6 *Confusion matrix of recognition accuracies on Weizmann dataset*

We evaluate the performance of continuous action recognition considering both the accuracy of recognition and segmentation. We define the evaluation metric of recognition accuracy as the ratio between the number of correct classified frames over the total number of frames. This metric is computed at frames level and the action label of the frame is that of the video clip it belongs to. When recognising isolated action, the label is assigned to the entire video. The recognition accuracy is defined as the rate of right classified videos.

5.1 Weizmann dataset

The Weizmann dataset consists of 90 video sequences of 10 persons. Each person performs ten classes of isolated actions: bend, jumping jack (jack), jump forward on two legs (jump), jump in place on two legs (pjump), run, gallop sideways (side), skip, walk, wave one hand (wave1) and wave two hands (wave2). We concatenate ten actions of each person in an arbitrary order, thus generating nine long sequences. We randomly select eight sequences for training and one sequence for testing. We compare our method with other methods: multi-class SVM [14] and semi-Markov model (SMM) [20].

The configurations of our CNN-HMM model for Weizmann dataset are as follows. The size of rescaled person images is 90×50. The width of sliding window is $d = 5$ and its stride is $s = 4$. The sliding window operation totally produces 1359 video clips. The size of input blocks to grey layer, gradient layer and optical flow layer are $90 \times 50 \times 5$, $90 \times 50 \times 5$, $90 \times 50 \times 4$, respectively. Two convolutional and sub-sampling layers are applied on each of the three channels separately. For the first convolutional layer, we use 16 different 3D kernels of size $8 \times 8 \times 5$ on each input layer ($8 \times 8 \times 4$ for optical flow layer), generating 16 feature maps. The size of max pooling window for the first sub-sampling layer is 2×2. For the second convolutional layer, eight different 3D kernels of size $4 \times 4 \times 5$ on each input layer ($4 \times 4 \times 4$ for optical flow layer) are used. The size of max pooling window for the second sub-sampling layer is 3×3. After convolutions and sub-samplings, the total number of output nodes of all channels is 2400. They are combined and connected to 50 nodes in the first fully connection layer. The second fully connection layer has ten output nodes. The setting of these configurations is determined by searching the parameter values which achieve the best performance in the process of experiments.

Table 1 Comparison of continuous action recognition results on Weizmann dataset

Methods	Average recognition accuracy, %
CNN-HMM	89.2
multi-class SVM [14]	87.7
SMM [20]	69.7

We run our model 20 times to get average recognition accuracy. In every run time, we randomly select one person's action data from training set to initialise the CNN, and give the action labels and labels order of the remaining training data. It takes 7 h 12 min to train this model. The average recognition speed is 75.75 frames per second.

Fig. 5 displays our recognition and segmentation result of one continuous action sequence and the ground truth is compared with this result. In this result, the recognition accuracy is 91.95%. We can see that the segmentation and recognition of actions is satisfying except the action run is wrongly recognised as skip in this example.

The average recognition accuracies are shown in Table 1. The results of multi-class SVM and SMM were reported in [14]. Our CNN-HMM model achieves the highest average recognition accuracy of 89.2%, better than that of multi-class SVM and SMM. Note that multi-class SVM method use provided binary masks which precisely locate the persons, our method do not use this information.

The confusion matrix for recognition accuracies of all actions is shown in Fig. 6. We find that the recognition accuracies of most actions are high, except for that the action run and skip are often confused by our method. The reasons we conclude are: (i) the image presentations of action run and skip are similar; (ii) the amount of data in Weizmann is small, and this makes the training of CNN-HMM model inadequate to classify them.

5.2 KTH dataset

The KTH dataset consists of 25 persons performing 6 classes of actions: boxing, hand clapping, hand waving, jogging, running and walking, which are performed in four different sceneries. It contains 599 videos and we use the same sliding window as above producing 46,509 video clips. The amount of data is much larger than that of Weizmann, and it is beneficial for our CNN-HMM model to achieve good result. In order to illustrate the advantage of action feature learnt by CNN, we implement a Harris GMM-HMM method which extracts 3D Harris space–time point features. The Harris GMM-HMM method uses GMM to model the emission probability of states. For fair comparison, we also use GMM to model the emission probability of CNN feature, and we denote this method as CNN GMM-HMM. We will see that CNN feature works better than 3D Harris feature, and the CNN-HMM method using CNN to model the emission probability instead of GMM improves the performance further.

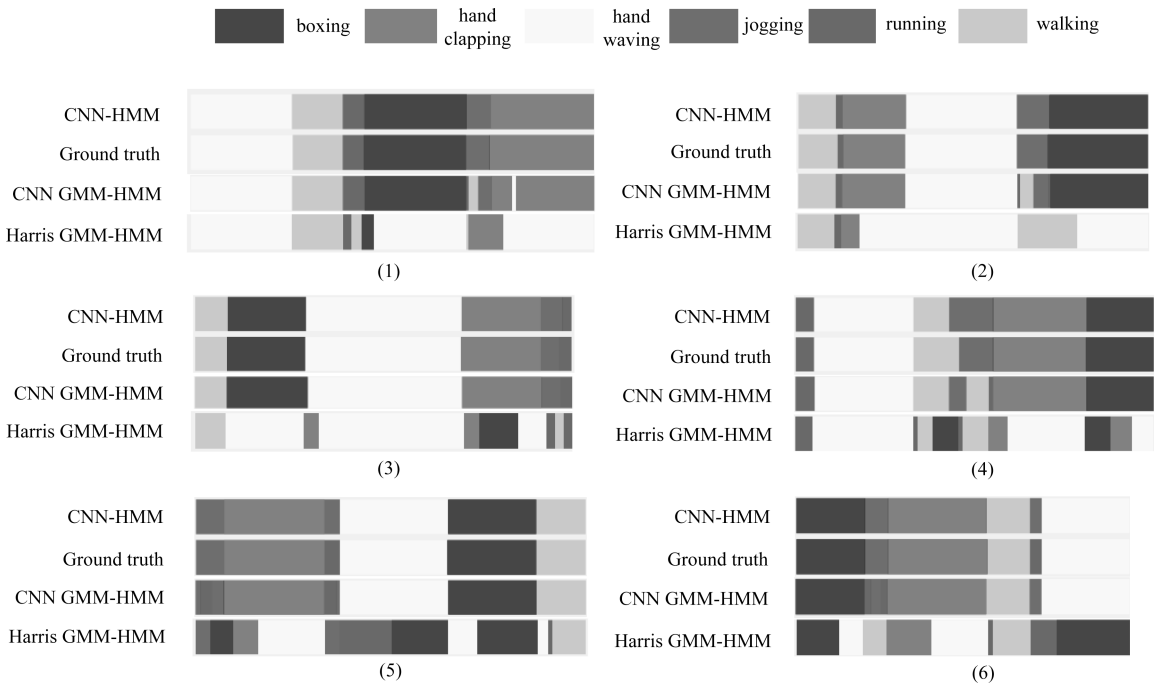

Fig. 7 *Comparison of recognition and segmentation results are represented in colour bar form. The results of CNN-HMM, Harris GMM-HMM and CNN GMM-HMM methods together with the ground truth are compared on several long video sequences. The numbers of total frames from (1) to (6) are 2006, 1578, 1810, 1730, 1930 and 1474, respectively*

The architecture of the CNN used for KTH dataset is the same as that used for Weizmann dataset, while we increase the number of network nodes as the training data is larger. 32 and 16 3D kernels are used in the first and second convolutional layer, respectively, in each of the three channels. After convolutions and sub-samplings, the total number of output nodes of all channels is 6400. The number of nodes in the first fully connection layer is set to be 100, and the second fully connection layer has six output nodes. Other parameters remain the same.

For the Harris GMM-HMM method, we extract 3D Harris features in each video clip, and describe them by HOG and HOF descriptors. The dimensionality of HOG and HOF descriptors is 162 (HOG: 72, HOF: 90). We construct a 100 words dictionary using *k*-means clustering, and the features are quantised on this dictionary. Therefore, the dimensionality of the quantised 3D Harris feature is 100. For the CNN GMM-HMM method, we use the same trained CNN in our CNN-HMM method to extract features, which are the outputs of the first fully connection layer. The dimensionality of the CNN feature is 100. As GMM is difficult at modelling high dimensional data, the principal component analysis (PCA) is performed to reduce the dimensionality of feature. We select the amount of principal components to preserve 98% percentage of the data information. After PCA, the dimensionality of CNN and 3D Harris features reduce to 5 and 75, repetitively. The number of mixture components of GMM is varying from 1 to 100 with interval of 5. The mixture components number is set to be the one that obtains the best result in the experiment.

We generate 100 long video sequences by concatenating these videos, and each long video sequence contains six classes of actions in arbitrary order. In all experiments, we use 80 long video sequences for training, and the remaining data is used for testing. In our CNN-HMM method, we use ten long video sequences to initialise the CNN. The process is repeated for five times in each method. It takes 21 h 3 min to train our model. The average recognition speed is 72.85 frames per second.

The average recognition accuracies of the three methods are shown in Table 2. The CNN-HMM and CNN GMM-HMM methods perform well, achieving an average recognition accuracy of 94.43 and 93.72%. However, the Harris GMM-HMM model does not solve this problem well, and gets a low average recognition accuracy of 42.69%.

We show recognition and segmentation results on several long sequences in Fig. 7. We can see that the segmentations of adjacent actions by the CNN-HMM method are relatively accurate. It recognises all actions in long sequences from (1) to (4), while the action running is wrongly recognised as jogging in long sequences (5) and (6). The CNN GMM-HMM method is inferior to CNN-HMM method for some small fragments in actions are misclassified. The recognition and segmentation results of Harris GMM-HMM are bad. The segmentations are fragmented. Fig. 8 shows the confusion matrixes for recognition accuracies of all actions of the three methods. The recognition accuracies of jogging and running are low in both CNN-HMM and CNN GMM-HMM methods. The two actions are difficult to be distinguished as the reason that the posture gestures of jogging and running are similar.

The experimental results demonstrate the advantage of CNN feature over 3D Harris feature. The CNN feature has a more compact and informative representation, and only small amount of principal components can preserve most of the data information. Meanwhile, the CNN feature can still describe the actions with sufficient accuracy. On the contrary, as its representation of high dimensionality, the 3D Harris feature cannot be well modelled by GMM, which causes the poor performance of Harris GMM-HMM method.

Our CNN-HMM method can also be used to recognise isolated action. We follow the training/test split setting as [8]. 16 randomly selected persons for training and 9 persons for test. The recognition accuracy is averaged across five random trials. Results of ours and other state-of-art methods using deep model are shown in Table 3. Our method outperforms 3D CNN [8] and ICA [18].

Table 2 Comparison of continuous action recognition results on KTH dataset

Methods	Average recognition accuracy, %
CNN-HMM	94.43
CNN GMM-HMM	93.72
Harris GMM-HMM	42.69

6 Conclusion

In this paper, we introduce a hybrid CNN-HMM model to recognise and segment continuous action simultaneously. We make full use of the respective advantages of CNN and HMM: CNN's

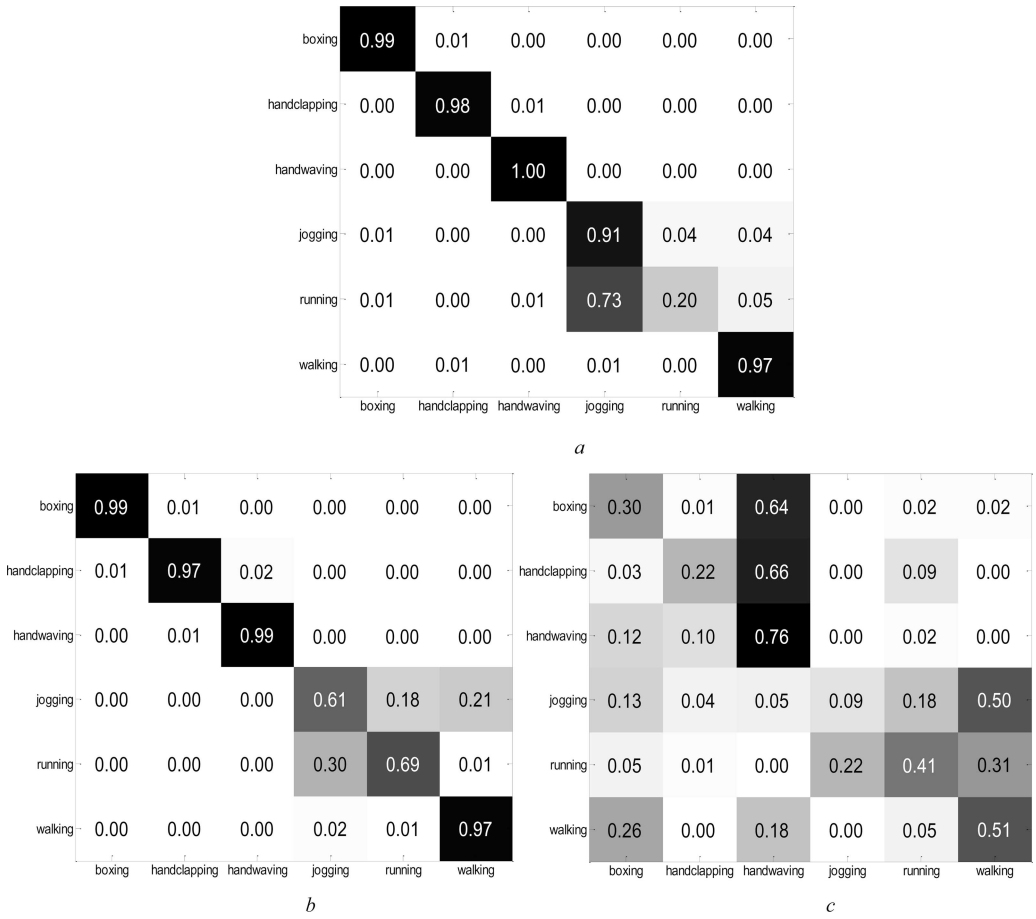

Fig. 8 *Confusion matrixes of recognition accuracies on KTH dataset*
a Confusion matrix for CNN-HMM method
b Confusion matrix for CNN GMM-HMM method
c Confusion matrix for Harris GMM-HMM method

Table 3 Comparison of isolated action recognition results on KTH dataset

Methods	Average recognition accuracy, %
ICA [18]	93.90
3D CNN [8]	90.20
our CNN-HMM	93.97

powerful ability of automatically learning efficient action features without prior knowledge of the application scenery, and HMM's strong ability of modelling time sequence problem. The CNN and HMM are combined by using CNN instead of GMM to model the emission probability in HMM. The CNN-HMM model is trained by embedded Viterbi algorithm. The data used to train CNN are labelled by forced alignment, and we do not need to manually segment and label large amount of continuous action videos. Experimental results demonstrate the practicability and accuracy of this method. As shown in compared experiments, the feature learnt by CNN has superior performance to 3D Harris feature on KTH dataset, and the CNN-HMM model achieves better recognition accuracy both on continuous and isolated action than other compared methods.

Though the framework of combining CNN and HMM efficiently solves the continuous action recognition problem, some improvements can be made in the future. The end-to-end integration way should be exploited to make the combining of CNN and HMM more seamless, and the HMM could also be replaced by more reasonable and effective time sequential model.

7 Acknowledgments

This work was partially supported by the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing grant 2015A04.

8 References

[1] Bobick, A.F., Davis, J.W.: 'The recognition of human movement using temporal templates', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2001, **23**, (3), pp. 257–267
[2] Thurau, C., Hlavac, V.: 'Pose primitive based human action recognition in videos or still images'. CVPR, Anchorage, AK, June 2008, pp. 1–6
[3] Blank, M., Gorelick, L., Shechtman, E., *et al.*: 'Actions as space-time shapes', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2007, **29**, (12), pp. 1395–1402
[4] Laptev, I., Marszalek, M., Schmid, C., *et al.*: 'Learning realistic human actions from movies'. CVPR, Anchorage, AK, June 2008, pp. 1–8
[5] Hinton, G.E., Osindero, S., Teh, Y.: 'A fast learning algorithm for deep belief nets', *Neural Comput.*, 2006, **18**, (7), pp. 1527–1554
[6] LeCun, Y., Bottou, L., Bengio, Y., *et al.*: 'Gradient based learning applied to document recognition', *Proceedings of the IEEE*, 1998, **86**, (11), pp. 2278–2324
[7] LeCun, Y., Kavukcuoglu, K., Farabet, C.: 'Convolutional networks and applications in vision'. Proc. 2010 IEEE Int. Symp. on Circuits and Systems, Paris, France, May 2010, pp. 253–256
[8] Ji, S., Yang, M., Yu, K.: '3D convolutional neural networks for human action recognition', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013, **35**, (1), pp. 221–231
[9] Duchenne, O., Laptev, I., Sivic, J., *et al.*: 'Automatic annotation of human actions in video'. ICCV, Kyoto, Japan, September 2009, pp. 1491–1498
[10] Yuan, J., Liu, Z., Wu, Y.: 'Discriminative subvolume search for efficient action detection'. CVPR, Miami, FL, June 2009, pp. 2442–2449
[11] Lv, F., Nevatia, R.: 'Recognition and segmentation of 3-d human action using HMM and multi-class AdaBoost'. ECCV, Graz, Austria, May 2006, pp. 359–372
[12] Ning, H., Xu, W., Gong, Y., *et al.*: 'Latent pose estimator for continuous action recognition'. ECCV, Marseille, France, October 2008, pp. 419–433
[13] Kulkarni, K., Evangelidis, G., Cech, J., *et al.*: 'Continuous action recognition based on sequence alignment', *Int. J. Comput. Vis.*, 2014, **112**, (1), pp. 90–114
[14] Hoai, M., Lan, Z., De la Torre, F.: 'Joint segmentation and classification of human actions in video'. CVPR, Providence, RI, June 2011, pp. 3265–3272

[15] Karpathy, A., Toderici, G., Shetty, S., *et al.*: 'Large-scale video classification with convolutional neural networks'. CVPR, Columbus, OH, June 2014, pp. 1725–1732
[16] Baccouche, M., Mamalet, F., Wolf, C., *et al.*: 'Sequential deep learning for human action recognition'. Human Behaviour Understanding, Springer Berlin Heidelberg, 2011, pp. 29–39
[17] Ng, J.Y., Hausknecht, M., Vijayanarasimhan, S.: 'Beyond short snippets: deep networks for video classification'. CVPR, Moston, MA, June 2015, pp. 4694–4702
[18] Le, Q.V., Zou, W.Y., Yeung, S.Y., *et al.*: 'Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis'. CVPR, Providence, RI, June 2011, pp. 3361–3368
[19] Hasan, M., Roy-Chowdhury, A.K.: 'Continuous learning of human activity models using deep nets'. ECCV, Zurich, Switzerland, September 2014, pp. 705–720
[20] Shi, Q., Cheng, L., Wang, L., *et al.*: 'Discriminative human action segmentation and recognition using semi-Markov model', *Int. J. Comput. Vis.*, 2011, **93**, (1), pp. 22–32
[21] Wang, Z., Wang, J., Xiao, J., *et al.*: 'Substructure and boundary modeling for continuous action recognition'. CVPR, Providence, RI, June 2012, pp. 1330–1337
[22] Lv, F., Nevatia, R.: 'Single view human action recognition using key pose matching and Viterbi path searching'. CVPR, Minneapolis, MN, June 2007, pp. 1–8
[23] Guenterberg, E., Ghasemzadeh, H., Loseu, V., *et al.*: 'Distributed continuous action recognition using a hidden Markov model in body sensor networks'. Distributed Computing in Sensor Systems, Springer Berlin Heidelberg, 2009, pp. 145–158
[24] Guo, Q., Tu, D., Lei, J., *et al.*: 'Hybrid CNN-HMM model for street view house number recognition'. ACCV Workshops, Singapore, Singapore, November 2014, pp. 303–315
[25] Wu, D., Shao, L.: 'Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition'. CVPR, Columbus, OH, June 2014, pp. 724–731
[26] Young, S.J., Russell, N.H., Thornton, J.H.S.: '*Token passing: a conceptual model for connected speech recognition systems*' (Cambridge University Engineering Department, Cambridge, UK, 1989)
[27] Baum, L.E., Petrie, T.: 'Statistical inference for probabilistic functions of finite state Markov chains', *Ann. Math. Stat.*, 1967, **37**, (6), pp. 1554–1563
[28] Bourlard, H.A., Morgan, N.: '*Connectionist speech recognition: a hybrid approach*' (Springer Science & Business Media, 2012)
[29] Morgan, N., Bourlard, H.: 'Continuous speech recognition', *IEEE Signal Process. Mag.*, 1995, **12**, (3), pp. 24–42
[30] Schüldt, C., Laptev, I., Caputo, B.: '*Recognizing human actions: a local SVM approach*' (CVPR, Washington, DC, USA, 2004), pp. 32–36
[31] Felzenszwalb, P.F., Girshick, R.B., McAllester, D., *et al.*: 'Object detection with discriminatively trained part-based models', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2010, **32**, (9), pp. 1627–1645
[32] Lei, J., Li, G., Tu, D., *et al.*: 'Convolutional restricted Boltzmann machines learning for robust visual tracking', *Neural Comput. Appl.*, 2014, **25**, (6), pp. 1383–1391