Egocentric Gesture Recognition Using Recurrent 3D Convolutional Neural Networks with Spatiotemporal Transformer Modules

Congqi Cao[1,2], Yifan Zhang[1,2] *, Yi Wu[3,4], Hanqing Lu[1,2] and Jian Cheng[1,2,5]
[1]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]School of Technology, Nanjing Audit University
[4]Department of Medicine, Indiana University, USA
[5]CAS Center for Excellence in Brain Science and Intelligence Technology
{congqi.cao, yfzhang, luhq, jcheng}@nlpr.ia.ac.cn, ywu.china@gmail.com

Abstract

Gesture is a natural interface in interacting with wearable devices such as VR/AR helmet and glasses. The main challenge of gesture recognition in egocentric vision arises from the global camera motion caused by the spontaneous head movement of the device wearer. In this paper, we address the problem by a novel recurrent 3D convolutional neural network for end-to-end learning. We specially design a spatiotemporal transformer module with recurrent connections between neighboring time slices which can actively transform a 3D feature map into a canonical view in both spatial and temporal dimensions. To validate our method, we introduce a new dataset with sufficient size, variation and reality, which contains 83 gestures designed for interaction with wearable devices, and more than 24,000 RGB-D gesture samples from 50 subjects captured in 6 scenes. On this dataset, we show that the proposed network outperforms competing state-of-the-art algorithms. Moreover, our method can achieve state-of-the-art performance on the challenging GTEA egocentric action dataset.

1. Introduction

With the development and popularity of the wearable devices such as VR/AR helmet and glasses, there is a demand to manipulate these devices intelligently. Since gesture is a common form for human communication and hands can be conveniently captured by cameras mounted on the devices from the first person view, hand gesture is a natural way to interact with wearable devices. It motivates the demand to recognize meaningful gestures from egocentric videos automatically.

In the traditional gesture recognition implementations, hand-crafted features are commonly adopted [16, 30, 20]. With the development of deep neural networks, end-to-end learning frameworks based on convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are applied to gesture recognition achieving state-of-the-art performance [18, 14, 19]. First-person vision provides a new perspective of the visual world that is inherently human-centric, and thus brings its unique characteristics to gesture recognition: 1) Egocentric motion: since the camera is worn on the user's head, camera motion can be significant due to the head movement, in particular when the user interacts while walking. 2) Hands in close range: due to the short distance from the camera to the hands and the narrow field-of-view of the egocentric camera, hands are prominent in the frame but meanwhile could be partly or even totally out of the field-of-view. The frameworks proposed for the second and third person view gesture recognition can not deal with these challenges very well.

Another notable issue in the field of egocentric gesture recognition is a lack of large scale training data for developing models especially the deep networks. With limited available datasets such as American sign language dataset [27] (which defines 40 American sign language gestures for deaf captured with only 1 subject) and Interactive museum database [2] (which contains 7 gesture classes performed by 5 subjects), a few methods have been developed. Starner *et al*. [27] use Hidden Markov model to recognize the sentence-level sign language with hand-crafted features extracted from hand blobs. Baraldi *et al*. [2] extract dense trajectory features inside and around hand regions which need to perform hand detection in advance and remove the camera motion by estimating the homography between two consecutive frames. However, hand detection brings addition-

*Corresponding author

al computational cost. Another disadvantage of the multi-stage framework is that the recognition performance heavily relies on the accuracy of the hand detection and camera motion estimation algorithms, which could be the bottleneck in case with large data variation.

We aim to design an end-to-end learnable egocentric gesture recognition model without detecting hand and estimating head motion explicitly and independently. Since 3D CNNs and RNNs have been proved to be effective at video analysis [28, 7, 19], we connect a RNN after a 3D CNN to process video sequences, constituting a framework for egocentric gesture recognition. Inspired by the spatial transformer [11] which is introduced to spatially transform images, we propose a novel spatiotemporal transformer module (STTM) to actively transform 3D feature maps into a canonical view in both spatial and temporal dimensions. The STTM mainly consists of three parts: a localization network for transformation parameters prediction, a grid generator for sampling grid generation, and a sampler for feature map interpolation. In order to handle the spatial and temporal variations in video simultaneously and universally, we use 3D homography transformation to warp the spatiotemporal feature maps of 3D CNNs. For better learning ability, we include recurrence between STTMs to embed long-term information. The recurrent STTM (RSTTM) can be inserted into a 3D CNN between any two convolutional layers. The whole framework as shown in Figure 1 is end-to-end learnable.

To validate our method, we introduce up-to-date the largest egocentric gesture dataset, named EgoGesture, with sufficient size, variation and reality, to be able to train deep networks. This dataset contains over 24,000 gesture samples and 3,000,000 frames for both color and depth modalities from 50 distinct subjects. We design 83 gestures focusing on interaction with wearable devices and collect them from 6 diverse indoor and outdoor scenes. We also consider the scenario when people perform gestures while walking. This dataset provides the test-bed not only for gesture classification in segmented data but also for gesture detection in continuous data.

The main contributions of our work include:

- We extend spatial transformer in temporal dimension, obtaining a 3D spatiotemporal transformer which can be directly applied to 3D CNNs for video processing.
- We utilize homography transformations to deal with head motion in egocentric videos which can rectify the warp caused by head movements.
- We estimate the transformation parameters at current time based on the previous ones on video sequences by introducing recurrent connections.

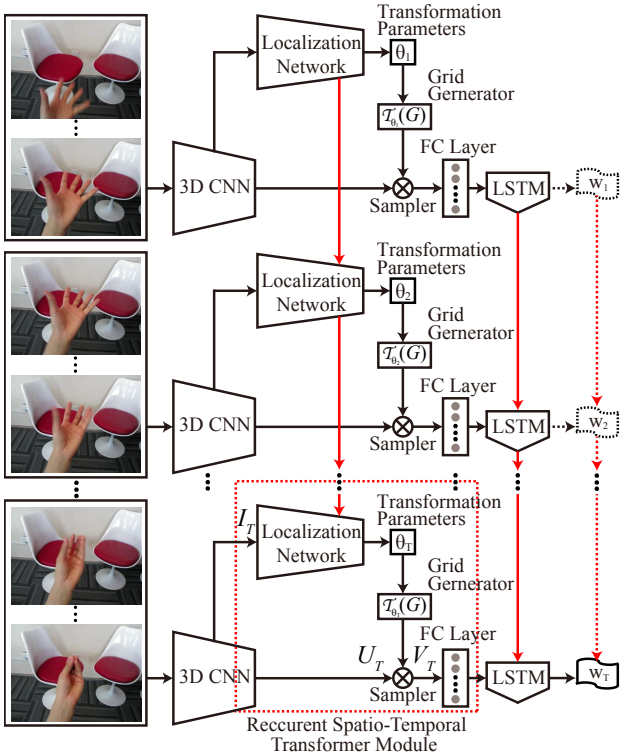

Figure 1. Illustration of the recurrent 3D CNN with RSTTM. Video sequence is split to clips and then input to the 3D CNN with RSTTM for feature extraction. LSTM is used to model the temporal transition and dependency between video clips, outputting the class label for each clip or only returning the final sequential label.

- We propose a benchmark dataset to help the community to move steps forward in egocentric gesture recognition, make it possible to apply data-hungry methods.

2. Related Work

There are several branches in egocentric computer vision related to hands, such as hand detection and segmentation [1], fingertip detection [10], pose estimation [21], gesture recognition [2], human-object interaction [8, 17] and cooking action recognition [24]. Besides the hand detection tasks, most of the existing recognition methods [2, 17, 24] also need to detect hand regions explicitly. Baraldi *et al*. [2] and Singh *et al*. [24] employ hand segmentation algorithms based on detecting skin pixels before feature extraction. Ma *et al*. [17] pre-train a hand segmentation CNN to help find objects of interest for activity recognition. These hand detection based methods bring additional computational cost and are heavily relies on the accuracy of the hand detection algorithms. Most of the hand detection algorithms are based on skin model which is easy to be affected by the other irrelevant skin areas or wearing gloves. To model the motion caused by head movement, 2D affine [25] and 2D homography [24] transformations are used as a pre-processing step for image warping independently from the model learning.

However, we argue that it is better to design a transformer module integrated within the classification model which can directly enhance the discrimination of the representation and still maintain the end-to-end learning capability.

Jaderberg *et al*. [11] introduce a differentiable module, the Spatial Transformer, which can be trained to learn the optimum transformation parameters conditionally on the input only with the class label for supervision, to improve the robustness of CNNs towards translation, scaling, rotation and even more generic image warping variances. Sønderby *et al*. [26] extend the spatial transformer network (STN) with RNN for digit sequence recognition in spatial dimension. This recurrent STN model performs better than the feed-forward STN model [11] by attending to an individual digit at a time. Zhong *et al*. [35] apply the spatial transformer module to alignment learning in face recognition. Comparing to similarity and affine transformations, the homography transformation is proved to be more suitable for face recognition in [35]. However, the above researches are all based on images. For video analysis, van Amersfoort *et al*. [29] utilize a 2D affine transformer to predict the next frame in video. A CNN is used to predict a series of affine transformations applied on the current frame patches to generate the next frame. Although transformations of a few previous frames are given to the CNN, there are no temporal connections to model the sequential information. We extend 2D spatial transformers to 3D spatiotemporal transformers, and choose the most suitable transformation type, i.e. homography, for egocentric gesture recognition. Moreover, we include recurrence in our model not only for label prediction but also for transformation estimation, which takes full advantage of the temporal information in videos.

3. Method

In this section, we describe the architecture of the proposed recurrent 3D convolutional neural network with recurrent spatiotemporal transformer module.

3.1. Recurrent 3D Convolutional Neural Network

3D CNNs and RNNs have been testified [28, 19, 4, 7, 15] to be good at video representation and sequence modeling respectively. We propose a framework of recurrent 3D CNN in an end-to-end learning paradigm, which can not only capture short-term spatiotemporal features, but also model long-term dependencies.

C3D [28] and long short term memory network (LSTM) [9] are chosen as the basic brick to construct our framework. C3D [28] is a 3D convolutional neural network with eight 3D convolutional layers, one 2D pooling layers, four 3D pooling layers and three fully-connected layers. The 3D layers take a volume as input and output a volume which can preserve the spatiotemporal information of the input. LSTM [9] is employed to model the temporal evolution of

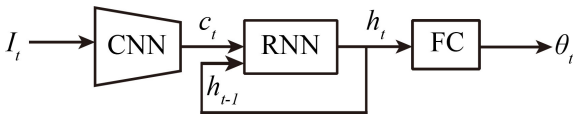

Figure 2. Illustration of the localization network.

sequences. Compared to the traditional RNN, it addresses the problem of gradient vanishing and explosion by inserting gate units. Specifically, we connect a single-layer LSTM with 256 hidden units after the first fully-connected layer ($fc6$ layer) of C3D to process sequence inputs. The recurrent 3D convolutional neural network can process video sequences of arbitrary lengths. For classification, the prediction of the last time slice can be used as the video label. Alternatively, we can also utilize the $lstm$ layer features to train a linear SVM classifier for recognition.

3.2. Recurrent Spatiotemporal Transformer

There are 3 parts in a recurrent spatiotemporal transformer module: a localization network, a grid generator and a sampler. As shown in Figure 1, the localization network predicts a set of transformation parameters conditioned on the input through a number of hidden layers. Then, the grid generator uses the predicted transformation parameters to construct a sampling grid, which is a set of points where the source map should be sampled to generate the target transformed output. Finally, the sampler takes the feature map to be transformed and the sampling grid as inputs, producing the output map sampled from the input at the grid points.

Specifically, at any time slice t, the localization network takes a 3D convolutional feature map I_t as input and predicts the transformation parameters θ_t that should be applied to the feature map U_t. Note that the input feature map of the localization network is not necessarily the feature map to be transformed, i.e. I_t and U_t can be different feature maps. The size of θ_t is determined by the transformation type, e.g. θ_t is 6-dimensional for 2D affine transformation and 9-dimensional for 2D homography transformation. We choose to use the most general homography transformation and apply it on 3D spatiotemporal convolutional feature maps. A homography is a non-singular, line preserving, projective mapping. A n-dimension homography is presented by a square $(n+1)$-dim matrix with $(n+1)^2-1$ degrees of freedom (DOF), where homogeneous coordinates are used to manipulate n-dim vectors in a $(n+1)$-dim space. In 3D homography case, the transformation predicted by the localization network is 16-dimensional:

$$f_{loc}(I_t) = H_{\theta_t} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} \\ \theta_{31} & \theta_{32} & \theta_{33} & \theta_{34} \\ \theta_{41} & \theta_{42} & \theta_{43} & \theta_{44} \end{bmatrix} \quad (1)$$

where we omit the symbol of time t in the matrix for clarity.

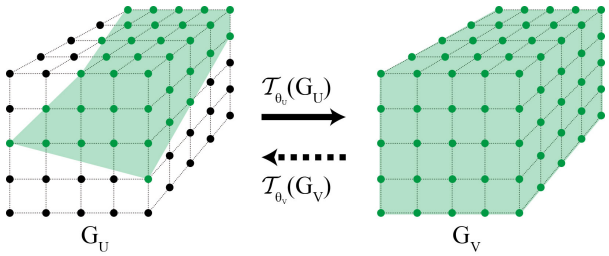

Figure 3. Example of the 3D grids before and after a transformation of homography.

We include recurrence in the localization network as illustrated in Figure 2 to predict the current transformation parameters conditioned on the previous ones:

$$c_t = f_{loc}^{cnn}(I_t) \quad (2)$$

$$h_t = f_{loc}^{rnn}(c_t, h_{t-1}) \quad (3)$$

$$H_{\theta_t} = f_{loc}^{fc}(h_t) \quad (4)$$

where f_{loc}^{cnn} is a 3D CNN which takes I_t as input and outputs a feature map c_t. f_{loc}^{rnn} is an RNN with hidden state h_t and f_{loc}^{fc} is a fully-connected layer to regress the transformation parameters. In our experiments, we use a GRU [5] layer with 256 hidden units as the f_{loc}^{rnn}.

The output transformation is used to create a sampling grid, which is done by the grid generator. For 3D feature maps, define the activations of the target feature map V to lie on a regular grid $G = \{G_i\}$ of coordinates $G_i = (x_i^t, y_i^t, z_i^t)^\mathsf{T}$. $(x_i^s, y_i^s, z_i^s)^\mathsf{T}$ is used to represent the source coordinate in the feature map U that defines the sample points. Note that, the index of time in the symbols of parameters is omitted for clarity from now on. In the program, all the coordinates are normalized to the range of $[-1, 1]$. The process of grid generator can be formulated as following:

$$\mathcal{T}_\theta(G_i) = (x_i^s, y_i^s, z_i^s)^\mathsf{T} \quad (5)$$

$$(x'^s_i, y'^s_i, z'^s_i, w)^\mathsf{T} = H_\theta(x_i^t, y_i^t, z_i^t, 1)^\mathsf{T} \quad (6)$$

$$(x'^s_i, y'^s_i, z'^s_i, w)^\mathsf{T} \Rightarrow (x'^s_i/w, y'^s_i/w, z'^s_i/w)^\mathsf{T} \quad (7)$$

$$\Rightarrow (x_i^s, y_i^s, z_i^s)^\mathsf{T} \quad (8)$$

An example of the 3D grids before and after a transformation of homography is shown in Figure 3. Given one sampling grid and the original pixel values in the grid, we are able to construct a new output by interpolation or sampling the pixel values on the corresponding grid positons.

Finally, the sampling grid $\mathcal{T}_\theta(G_i)$ and the feature map $U \in \mathbb{R}^{C \times L \times H \times W}$ with width W, height H, length L and C channels to be transformed are taken as input to a sampler. The sampler outputs feature map $V \in \mathbb{R}^{C \times L' \times H' \times W'}$ by sampling from U at the grid points. In the paper, we use bilinear kernel to do sampling.

$$V_i^c = \sum_m^W \sum_n^H \sum_k^L U_{mnl}^c \cdot [1 - |x_i^s - m|]_+ \cdot [1 - |y_i^s - n|]_+ \cdot [1 - |z_i^s - k|]_+ \quad (9)$$

where $[x]_+ = \max(0, x)$ and $\forall i \in [1, \ldots, L'H'W'], \forall c \in [1, \ldots, C]$. Note that the size of feature map V can be different from U by varying the number of sample points in the target and source coordinates. We specify a down-sampling parameter with three elements (r_l, r_h, r_w) to adjust the ratio of input size $L \times H \times W$ to output size $L' \times H' \times W'$ in spatiotemporal dimensions.

4. Dataset

In this paper, we introduce up-to-date the largest dataset called EgoGesture for the task of egocentric gesture recognition. The dataset, which is publicly available [1], contains 2,081 RGB-D videos, 24,161 gesture samples and 2,953,224 frames from 50 distinct subjects. We carefully design 83 classes of static or dynamic gestures specifically for interaction with wearable devices. Our dataset is more complex than any existing dataset as our data is collected from the most diverse yet representative scenes with large variations. The 6 scenes we designed consist of 4 indoor scenes: 1) the subject in a stationary state with a static clutter background; 2) the subject in a stationary state with a dynamic background; 3) the subject in a stationary state facing a window with drastic-changing sunlight; 4) the subject in a walking state; and 2 outdoor scenes: 5) the subject in a stationary state with a dynamic background; 6) the subject in a walking state with a dynamic background. We select Intel RealSense SR300 as our egocentric camera due to its small size and integrating both RGB and depth modules. The two-modality videos are recorded in a resolution of 640×480 pixel with 30 fps. The subjects wearing the RealSense camera with a strap belt on their heads are asked to continuously perform 9-14 gestures as a session and recorded as a video. Since the order of the gestures performed is randomly generated, the videos can be used to evaluate gesture detection in continuous stream. Besides the annotation of class label, the start and end frame index of each gesture sample are also manually labeled, which provides the test-bed for segmented gesture classification. We believe the proposed dataset can be used as a benchmark and help the community to move steps forward in egocentric gesture recognition, making it possible to apply data-hungry methods such as deep neural networks for this task.

[1] http://www.nlpr.ia.ac.cn/iva/yfzhang/datasets/egogesture.html

5. Experiments

We test the proposed model in our newly created EgoGesture dataset and another egocentric action dataset GTEA [8] which focuses on the cooking actions of hands. There are few appropriate egocentric gesture datasets for research as mentioned before. We do not take the Interactive Museum database [2] for experiment as it is less challenging. The recognition results on it are already high even with two gesture samples from each class as training set. Moreover, our proposed method is not restricted to gesture recognition field, it is a general framework for video analysis especially in the first person view. Hence we also evaluate the proposed model on GTEA dataset. All the models are implemented using Theano [3] and Lasagne [6]. Cross entropy loss and SGD are used for training.

For comparision, in our proposed EgoGesture Dataset, we systematically evaluate state-of-the-art methods based on both hand-crafted features and deep networks as baselines on two tasks: gesture classification and gesture detection. These methods are also the winner approaches in ChaLearn 2016 Looking at People ICPR Challenge [14]. We randomly split the data by subject into training (60%), validation (20%) and testing (20%) sets, resulting in 1,239 training, 411 validation and 431 testing videos. The numbers of gesture samples in training, validation and testing splits are 14416, 4768 and 4977 respectively.

5.1. Classification Results on EgoGesture dataset

For classification, we segment the video sequences into isolated gesture samples based on the manual annotation of the begin and the end frames. The learning task is to predict the class labels for each gesture sample. Classification accuracy is used as the evaluation metric.

We select one hand-craft features: iDT-FV [32], and four deep learning based methods: VGG16 [23], C3D [28], VGG16+LSTM [7] and IDMM+CaffeNet [34] as baselines. iDT-FV is a representative hand-crafted feature for local motion modeling where global camera motion is canceled out by optical flow estimation. We compute the Trajectory, HOG, HOF and MBH descriptors in the RGB videos. After PCA, we train GMMs with 256 Gaussians to generate Fisher Vectors (FV) for each type of the descriptor. Then, the FVs after L2 normalization are concatenated to form a video descriptor. Finally, linear SVM is used for classification. There are mainly four kinds of frameworks to classify video sequences with deep learning methods: 1) Use 2D CNNs to extract features of single frames. Then frame-level features are encoded as video descriptors and classifiers are trained to predict the labels of videos. 2) Use 3D CNNs to extract features of video clips. After that, clip features are aggregated into video descriptors for classifier training. 3) Make use of RNNs to model the temporal evolution of sequences based on convolutional features. 4) Represent a video as one or multiple compact images and then input it to a neural network for classification. We choose VGG16, C3D, VGG16+LSTM and IDMM+CaffeNet as baselines which correspond to the four kinds of deep learning frameworks described above. Among them, IDMM+CaffeNet [34] encodes both spatial and temporal information of a depth video into an image called improved depth motion map (IDMM), then uses CaffeNet [12] for classification. The other baselines take either an image or a 16-frame clip as input, then average pooling with $L2$ normalization are used to aggregate the frame-level or clip-level features (activations of the first fully-connected layer for CNNs and the $lstm$ layer for VGG16+LSTM) into a video-level descriptor. Finally, linear SVM is employed for classification. For RGB-D fusion, the scores of classification probability obtained from RGB and depth modalities are added with a weight which is chosen on the validation split.

The classification accuracies of the baselines are listed in Table 1. As we can see, in most cases, deep features performs much better than hand-crafted features. VGG16 does not perform as well as other deep approaches since it can only characterize the visual appearance and losses the temporal information seriously. Benefit from the attached temporal model, VGG16+LSTM improves the performance of VGG16 significantly. The performance of C3D is obviously superior to those of other methods with a margin more than 10%. It is probably because of the excellent spatiotemporal learning ability of C3D. For different modalities, the results on depth data are better than those on RGB data, since the short-range depth sensor can eliminate most of the noise from the background and the RGB data are sensitive to illumination changes. However, the depth sensor is easy to be affected with infrared and fast movements. The performance is further improved by fusing the results from the RGB and depth modalities together.

We analyse the performance of different transformers inserted to 2D CNNs and 3D CNNs (i.e. VGG16 and C3D). The results are shown in Table 2. This time, we use LSTM to model the evolutions of the whole gesture samples instead of fixed-length clips. In EgoGesture dataset, the average length of a segmented gesture is 38 frames, while the minimum and the maximum length are 3 and 196 frames respectively. For convenience, we constrain the maximum length of a gesture sample to be 40 frames by downsampling the longer ones. In the experiments based on C3D, we choose the last convolutional feature map and the second last convolutional feature map of C3D as I and U in the spatiotemporal transformer module. The f_{loc}^{cnn} is designed to consist of three 3D convolutional layers with 20 filters. The same experimental setting is applied to VGG16 based models, which use 2D convolutional layers instead of 3D ones. We test $(1, 1, 1)$, $(1, 2, 2)$, $(2, 2, 2)$, $(1, 3, 3)$, $(2, 3, 3)$ down-sampling factors (where the three elements represent

Table 1. The Baseline Classification results on EgoGesture dataset.

Method	Modality	Accuracy
iDT-FV [32]	RGB	0.643
VGG16 [23]	RGB	0.625
VGG16+LSTM [7]	RGB	0.747
C3D [28]	RGB	0.864
IDMM+CaffeNet [34]	depth	0.664
VGG16 [23]	depth	0.623
VGG16+LSTM [7]	depth	0.777
C3D [28]	depth	0.881
VGG16 [23]	RGB-D	0.665
VGG16+LSTM [7]	RGB-D	0.814
C3D [28]	RGB-D	**0.897**

Table 2. Analysis of the transformers on EgoGesture dataset. "A" and "H" stand for the affine and homography transformations respectively. The LSTMs are used to model the evolution of a whole sequence. For convenience, the maximum length of a sequence is constrained to 40 frames.

Method	Modality	Accuracy
VGG16+LSTM, 40frm	RGB	0.808
VGG16+LSTM+RSTM (A)	RGB	0.812
VGG16+LSTM+RSTM (H)	RGB	0.838
C3D	RGB	0.864
C3D+STM (A)	RGB	0.880
C3D+STM (H)	RGB	0.882
C3D+STTM (H)	RGB	0.887
C3D+LSTM	RGB	0.889
C3D+LSTM+STTM (H)	RGB	0.890
C3D+LSTM+RSTTM (H)	RGB	0.893
VGG16+LSTM+RSTM (H)	depth	0.857
C3D+STTM (H)	depth	0.895
C3D+LSTM+RSTTM (H)	depth	0.906
VGG16+LSTM+RSTM (H)	RGB-D	0.885
C3D+STTM (H)	RGB-D	0.917
C3D+LSTM+RSTTM (H)	RGB-D	**0.922**

the down-sampling coefficients in length, height, width respectively) for C3D in advance, and choose the best parameter with $(1, 3, 3)$. Similarly, the VGG16 based architecture performs best with $(3, 3)$ down-sampling factor in space. Actually, when the down-sampling factor is greater than one we introduce an information bottleneck forcing the model to zoom in the attention regions. We evaluate the performance of affine transformation and homography transformation in experiments. Table 2 shows that both 2D CNNs and 3D CNNs benefit from adding a transformer module. The improvement produced by homography is higher than that from affine. Spatiotemporal transformers can further improve the recognition results of spatial transformers. The best accuracy is achieved by recurrent 3D CNNs with recurrent spatiotemporal homography transformer modules.

Analysis of confusion matrix: The confusion matrix of C3D and C3D+LSTM+RSTTM with RGB-D fusion is shown in Figure 6. The gesture classes with the highest accuracy are: "Dual hands heart" (Class 53), "Pause" (Class 36), and "Cross index fingers" (Class 7) which are with an accuracy of 98.3% by C3D and 100% by C3D+LSTM+RSTTM. The gestures with the lowest classification accuracies are: "Grasp" (66.1% by C3D, 74.6% by our method), "Sweep cross" (71.2% by C3D, 83.1% by our method) and "Scroll hand towards right" (72.4% by C3D, 75.9% by our method). Specifically, the most confusing class of "Grasp" (Class 48) is "Palm to fist" (Class 43), "Sweep cross" (Class 19) is easy to be classified as "Sweep checkmark" (Class 20), while "Scroll hand towards right" (Class 1) is likely to be regard as "Scroll hand towards left" (Class 2). It is reasonable since these gestures contain similar movements. Our method improves the performance of C3D significantly, especially for the confusing classes.

Analysis of different scenes: By analyzing the classification results of each scene shown in figure 4, we can find that deep learning features are more robust than handcrafted features for illumination and global motions including egocentric movements and background dynamics. In general, RGB features are more sensitive to illumination changes which can be seen with the results of iDT-FV and C3D-RGB in scene 3. It is noteworthy that the large egocentric motion caused by walking hurts the performance of all the methods except our proposed model which can be seen in the results of scene 4 and 6. The results of scene 4 do not degenerate too much because the walking speed is low due to the space limit in an indoor environment. Our method persistently performs well in all the scenarios even if the egocentric motion is obviously.

Results on different egocentric movement intensities: Table 3 lists the improvements with our method on stationary and walking scenario domains using RGB data. Adding STTM to C3D can increase the accuracy on both domains. Especially the walking scenario domain benefits a lot from the spatiotemporal transformation ability of STTM. By introducing recurrence to handling temporal sequences, the performance is further improved. The recognition difference of C3D on the two domains is 3.8%, while the difference of C3D+RSTTM+LSTM is decreased to 1.4%. C3D+RSTTM+LSTM increases the accuracy on the walking scenario domain by 4% demonstrating its good ability to deal with egocentric motion.

Visualization of the spatiotemporal transformer: We visualize the feature maps before and after the 3D homography transformation in Figure 5. For more intuitive comparison, we use the $(1, 1, 1)$ down-sampling factor without changing the size of feature maps and choose two gesture samples belonging to the same class from two different subjects. Comparing the feature maps of sample1 and sample2 shows that the activations on the layer after transformation have a more similar appearance and especially more con-

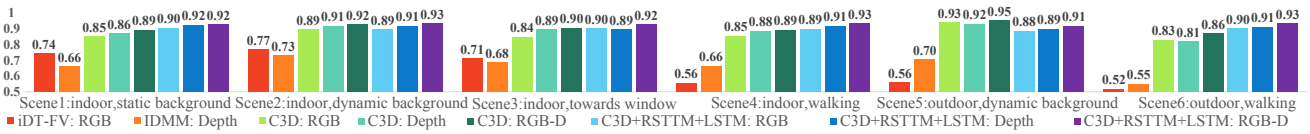

Figure 4. Classification accuracies in the 6 different scenes on EgoGesture dataset.

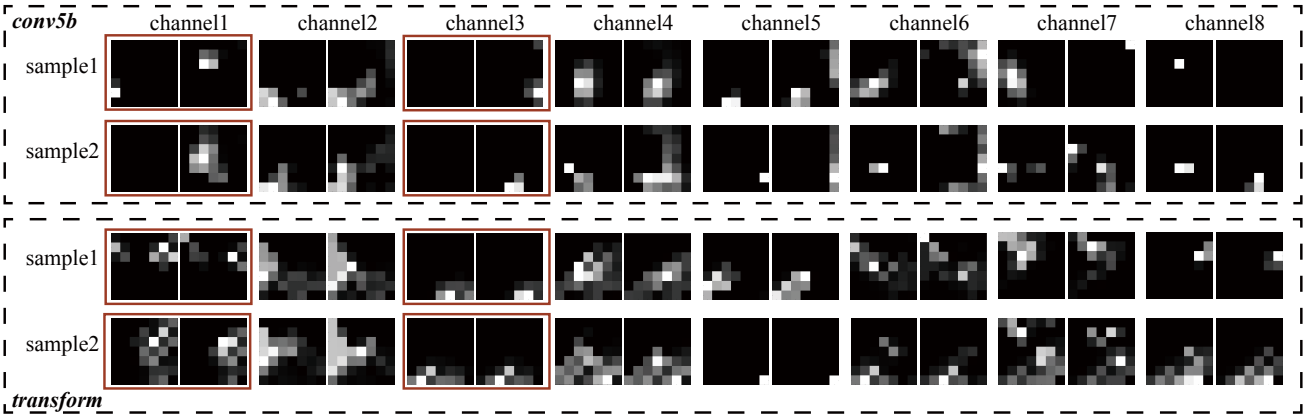

Figure 5. Feature maps before and after 3D homography transformation. We visualize the 3D feature maps of $conv5b$ (the layer to be transformed) and $transform$ (the layer after transformation) for comparison. The spatiotemporal feature maps are shown by a series of spatial images. We choose two gesture samples belonging to the same class from two different subjects as input.The first 8 channels are drawn in rows. A few representative channels are highlighted.

Table 3. Classification accuracies of the models trained on different domains with RGB data.

Domain	C3D	+STTM	+RSTTM+LSTM
stationary (scene1,2,3,5)	0.866	0.870 (↑0.004)	0.882 (↑0.016)
walking (scene4,6)	0.828	0.839 (↑0.011)	0.868 (↑0.040)

sistent temporal distribution than that on the layer before transformation. Taking the channels highlighted with red boxes for example, the feature maps before transformation are much different not only in the temporal dimension but also between the two samples, while the feature maps after transformation are evenly distributed in the temporal dimension and more similar between samples.

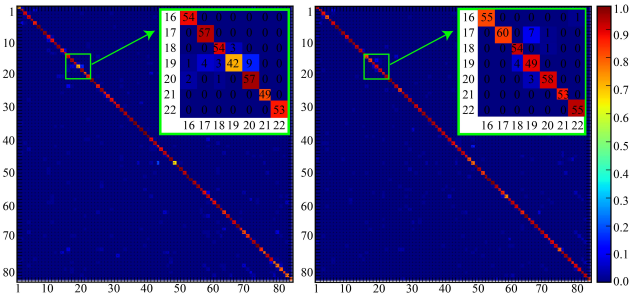

Figure 6. The confusion matrix of C3D and the recurrent C3D with RSTTM using RGB-D fusion on EgoGesture dataset.

5.2. Detection Results on EgoGesture dataset

For detection, we aim to spot and recognize gestures from continuous unsegmented video sequences. Performance is evaluated by the Jaccard index used in ChaLearn LAP 2016 challenges [30]. This metric measures the average relative overlap between the ground truth and the predicted label sequence for a given input.

We test two detection baselines: For the first baseline, we train a C3D to classify 84 gestures (with an extra non-gesture class). When testing, a 16-frame length sliding window with 8 or 16 frame stride is used to slide through the whole sequence to generate video clips. The class probability of each clip predicted by C3D's softmax layer is used to label all the frames in the clip. After summing the overlapping probabilities, the most possible class is chosen as the label for each single frame. The second baseline [34] tackles temporal segmentation and classification separately and sequentially. Firstly, the quantity of movement (QOM) [13] is used to detect the start and end frame of each candidate gesture in the stream. Then the IDMM is generated within a candidate gesture and input to CaffeNet for classification.

The Jaccard index for detection is shown in Table 4, where $l16s16$ denotes 16-frame length sliding window with 16-frame stride. We also list the runtime tested on a single GTX Titan X GPU and Intel i7-3770 CPU @3.4GHz. As shown in Table 4, small stride setting can achieve better performance with more computations. The best performance (70.9%) is achieved by C3D+STTM-$l16s8$ with RGB-D inputs. Optimizing the computation of grid gener-

Table 4. Detection results on EgoGesture dataset.

Method	Modality	Jaccard	Runtime
C3D [28]-$l16s16$	RGB	0.585	624fps
C3D-$l16s8$	RGB	0.659	312fps
C3D+STTM-$l16s8$	RGB	0.670	215fps
QOM+IDMM [34]	depth	0.430	30fps
C3D-$l16s16$	depth	0.600	626fps
C3D-$l16s8$	depth	0.678	313fps
C3D+STTM-$l16s8$	depth	0.681	229fps
C3D-$l16s16$	RGB-D	0.618	312fps
C3D-$l16s8$	RGB-D	0.698	156fps
C3D+STTM-$l16s8$	RGB-D	**0.709**	111fps

Table 5. Detection results on GTEA dataset. The results are evaluated in terms of frame-level accuracy for comparison with the published outcomes. We further evaluate our methods with data augmentation.

Method	Accuracy
DT [31]	0.452
iDT [32]	0.524
TDD (Spatial) [33]	0.586
TDD (Temporal) [33]	0.571
TDD (Spatial+Temporal) [33]	0.595
Ego ConvNet (2D) [24]	0.576
Ego ConvNet (3D) [24]	0.558
Ego ConvNet (2D+3D) [24]	0.589
Ours	**0.615**
Ours (augmentation)	**0.630**

ation and feature map interpolation should further speed up the runtime of STTM based models. In the second baseline method, the most time consuming step is to convert the depth sequence into one image with IDMM, making it less efficient than C3D with sliding window. Another disadvantage is that the detection performance heavily relies on the pre-segmentation which could be the bottleneck of the two-stage framework.

5.3. Detection Results on GTEA dataset

GTEA dataset [8] contains 28 videos belonging to 7 activities performed by 4 subjects. The activities are composed of several actions. Take the activity "Cheese" for example, it consists of "take" bread, "take" cheese, "open" cheese etc. actions operating with different objects. There are 10 annotations of actions defined by verbs. Including the idle state, the number of actions is 11. To compare with the published results on GTEA dataset, we follow the experimental settings of [24] by using the data of subject 1 and subject 3 for training, subject 4 for validation and subject 2 for testing. Performance is evaluated with frame-level accuracy for continuous video understanding.

The results of our proposed model and competing methods are shown in Table 5. From the table we can see that iDT [32] improves DT [31] significantly by canceling the global camera motion with optical flow estimation. TDD [33] is a kind of video descriptor which conduct trajectory-constrained pooling on two-stream networks [22]. The spatial stream takes a RGB image as input at a time, while the temporal stream takes a stack of optical flow images for input. Both the spatial stream and the temporal stream use 2D CNN to extract features. For TDD [33], the performance of the temporal stream is inferior to that of the spatial stream on the egocentric GTEA dataset, which is contrary to the results on the traditional video analysis fields. This demonstrates that head motion severely damage the performance of the representative methods proposed for traditional video-based action recognition. Ego ConveNet [24] uses hand-crafted egocentric cues (including hand masks, head motions and saliency maps) as input to a 2D CNN or 3D CNN model. The 2D CNN and 3D CNN both are small networks with only 2 convolutional layers. Since GTEA dataset is a relatively small dataset, even for the Ego ConveNet which used multiple well-designed features and shallow network architecture, the network is fine-tuned on a pre-trained gesture model. In order to alleviate overfitting, we use the model pre-trained on our proposed EgoGesture dataset to initialize the model on GTEA dataset. We also try to scale ($\pm 20\%$) and rotate ($\pm 15^{\circ}$) the input video clips in space randomly for data augmentation. The performance of our method outperforms competing methods significantly demonstrating its good learning ability.

For continuous video understanding, the accuracy of our method on the test split is close to that on the validation split. While for segmented video classification, due to the insufficient training data, overfitting is severe. Even though the accuracy on the test split is about 10% lower than that on the validation split with isolated actions, the recognition result can reach 75.0% without combining other features.

6. Conclusion

In this work, we propose a novel recurrent 3D CNN model with recurrent spatiotemporal transformer module which can deal with the egocentric motion effectively. We extend spatial affine transformers to spatiotemporal homography transformers for better learning ability and include recurrent connections between time steps to deal with video sequences. We also introduce up-to-date the largest dataset called EgoGesture for the task of egocentric gesture recognition with sufficient size, variation and reality, to successfully train deep networks.

7. Acknowledgments

We thank the valuable comments from the reviewers. This work is partly supported by National Natural Science Foundation of China (61332016, 61572500, 61370036) and Youth Innovation Promotion Association CAS.

References

[1] S. Bambach, S. Lee, D. J. Crandall, and C. Yu. Lending A hand: Detecting hands and recognizing activities in complex egocentric interactions. In *ICCV*, 2015.

[2] L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara. Gesture recognition in ego-centric videos using dense trajectories and hand segmentation. In *CVPRW*, 2014.

[3] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio. Theano: new features and speed improvements. *CoRR*, abs/1211.5590, 2012.

[4] N. C. Camgoz, S. Hadfield, O. Koller, and R. Bowden. Using convolutional 3d neural networks for user-independent continuous gesture recognition. *ICPRW*, 2016.

[5] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[6] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, D. Maturana, M. Thoma, E. Battenberg, J. Kelly, and other contibutors. Lasagne: First release. `http://dx.doi.org/10.5281/zenodo.27878`, Aug. 2015.

[7] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[8] A. Fathi, X. Ren, and J. M. Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011.

[9] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[10] Y. Huang, X. Liu, X. Zhang, and L. Jin. A pointing gesture based egocentric interaction system: Dataset, approach and application. In *CVPRW*, 2016.

[11] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM, MM*, 2014.

[13] F. Jiang, S. Zhang, S. Wu, Y. Gao, and D. Zhao. Multi-layered gesture recognition with kinect. *JMLR*, 16, 2015.

[14] H. J. E. Jun Wan. 2016 looking at people icpr challenge. `http://chalearnlap.cvc.uab.es/challenge/15/description/`. Accessed March 12, 2017.

[15] Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu. Online human action detection using joint classification-regression recurrent neural networks. In *ECCV*, 2016.

[16] L. Liu and L. Shao. Learning discriminative representations from rgb-d video data. In *IJCAI*, volume 1, 2013.

[17] M. Ma, H. Fan, and K. M. Kitani. Going deeper into first-person activity recognition. In *CVPR*, 2016.

[18] P. Molchanov, S. Gupta, K. Kim, and J. Kautz. Hand gesture recognition with 3d convolutional neural networks. In *CVPRW*, 2015.

[19] P. Molchanov, X. Yang, S. D. Mello, K. Kim, S. Tyree, and J. Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural networks. In *CVPR*, 2016.

[20] E. Ohn-Bar and M. M. Trivedi. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE TITS*, 15(6), 2014.

[21] G. Rogez, J. S. Supancic, and D. Ramanan. Understanding everyday hands in action from rgb-d images. In *ICCV*, 2015.

[22] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[24] S. Singh, C. Arora, and C. V. Jawahar. First person action recognition using deep learned descriptors. In *CVPR*, 2016.

[25] S. Singh, C. Arora, and C. V. Jawahar. Trajectory aligned features for first person action recognition. *PR*, 62, 2017.

[26] S. K. Sønderby, C. K. Sønderby, L. Maaløe, and O. Winther. Recurrent spatial transformer networks. *CoRR*, abs/1509.05329, 2015.

[27] T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *IEEE TPAMI*, 20(12), 1998.

[28] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[29] J. R. van Amersfoort, A. Kannan, M. Ranzato, A. Szlam, D. Tran, and S. Chintala. Transformation-based models of video sequences. *CoRR*, abs/1701.08435, 2017.

[30] J. Wan, S. Z. Li, Y. Zhao, S. Zhou, I. Guyon, and S. Escalera. Chalearn looking at people RGB-D isolated and continuous datasets for gesture recognition. In *CVPRW*, 2016.

[31] H. Wang, A. Kläser, C. Schmid, and C. Liu. Action recognition by dense trajectories. In *CVPR*, 2011.

[32] H. Wang and C. Schmid. Action Recognition with Improved Trajectories. In *ICCV*, 2013.

[33] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015.

[34] P. Wang, W. Li, S. Liu, Y. Zhang, Z. Gao, and P. Ogunbona. Large-scale continuous gesture recognition using convolutional neutral networks. *CoRR*, abs/1608.06338, 2016.

[35] Y. Zhong, J. Chen, and B. Huang. Towards end-to-end face recognition through alignment learning. *CoRR*, abs/1701.07174, 2017.