Large-scale Isolated Gesture Recognition Using a Refined Fused Model based on Masked Res-C3D Network and Skeleton LSTM

Chi Lin[†], Jun Wan[‡], Yanyan Liang[*§], Stan Z. Li[‡§]
[†]*University of Southern California, Los Angeles, California 90089-0911, USA*
[‡]*National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, BeiJing, China*
[§]*Faculty of Information Technology, Macau University of Science and Technology, Macau SAR, China*

***Abstract*—In this paper, we present a novel refined fused model combining Masked Res-C3D Network and Skeleton LSTM for large-scale isolated gesture recognition in RGB-D videos. The key of our design is to learn a compact and effective presentation of gesture sequences and integrate multiple models together. First, deep spatio-temporal features are well extracted by 3D Convolutional Neural Networks (CNNs) with residual architecture (Res-C3D). As gestures are mainly derived from the arm or hand movements, a masked Res-C3D network is built to decrease the effect of background and other variations via exacting the skeleton of body and reserving arm regions with discarding other regions via a mask. And then, relative positions and angles of different keypoints are extracted and used to build a time-series model by Long Short Term Memory network (LSTM). Based the above representations, a fusion scheme for blending classification results via a convolution layer (called fused layer) is developed, in which the weights of each voting sub-classifier being of advantage to a certain class in our ensemble model are adaptively obtained by training in place of fixed weights. Our experimental results show that the proposed method has obtained a state-of-the-art performance with accuracy 0.6842 in the IsoGD dataset.**

***Keywords*-Large-scale Isolated Gesture Recognition; Multi-Modal Fusion Model; Res-C3D Network;**

I. INTRODUCTION

Gesture recognition is a fast expanding field with applications beyond gaming, consumer electronics UI, automotive, sports training and etc. The main task of gesture recognition is to extract features from an image or a video and then classify or determine each sample to a certain label.

At present, although most existing models have reached a high performance for isolated gesture recognition, there are still a lot of insufficiencies and restrictions in practical applications. Firstly, most ensemble models [1][2] are used to extract different features to improve performance, in which the model size and its training/inference time may be unacceptable in most practical uses. For example, our rebuilt implementations indicate that each sub-model needs more than ten hours or even several days to be trained and to be integrated into a ensemble model. Moreover, the inference cost is relative high as the testing data needs to be propagated through all the sub-model to obtain the feature in inference stage. Secondly, although lots of skeleton data in favor of classification have been used [3], their valuable potential information have not yet been fully excavated. Finally, in most of current ensemble models, different weights of each voting sub-classifier being of advantage to a certain class have been noticed and used [4][5], however, they are still fixed values throwing away flexibility. Intuitively, it does not make sense for setting fixed weights of voting sub-classifier for each task.

To solve these problems above, we proposed a relatively simple model to reduce computational cost but more effectively extracting features from videos and skeleton data, and employed adaptive weights of each voting sub-classifier to improve performance. The pipeline of our method is depicted in Figure 1, and the main contribution can be summarized as following:

- We built a relatively simple model and obtained state-of-the-art performance on the large-scale isolated gesture dataset (IsoGD). Most of existing high-performance methods adopt ensemble strategy to merge a large number of models for extracting and blending features. Our approach is more simpler but achieves better performance.
- We developed an adaptive scheme for setting weights of each voting sub-classifier via a convolution layer (called fused layer). Unlike traditional methods [4][5] with the fixed-weighted values among different models, the weights can be learned in training stage via common BP method.
- We excavated more potential information of skeleton data in favor of classification. Beyond the current works [1][2], we used masked res-C3D network and proposed skeleton LSTM to use a variety of information, such as angle, relative displacement, skeleton information, to more thoroughly dig out the efficient features for gesture recognition.

The remainder of this paper is organized as follows: Section 2 is a review of related works. Section 3 illustrates our method and section 4 validates our results respectively. And finally the section 5 is the conclusion.

*Corresponding Author
978-1-5386-2335-0/18/$31.00

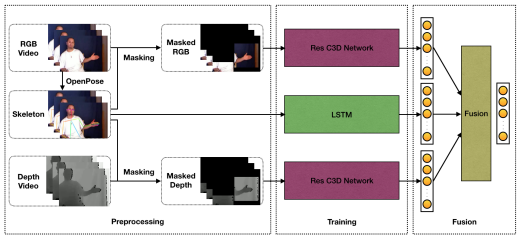

Figure 1. The pipeline of our method. Left: The preprocessing of training data, including extracting skeleton data, reducing the influence of background via OpenPose and masking the RGB videos and Depth videos; Middle: The training phase, the data need to be normalized and augmented, and then be fed into the models for training; Right: Fusion phase, the features extracted from two Res-18 C3D Networks and LSTM model are fused and classification results are output.

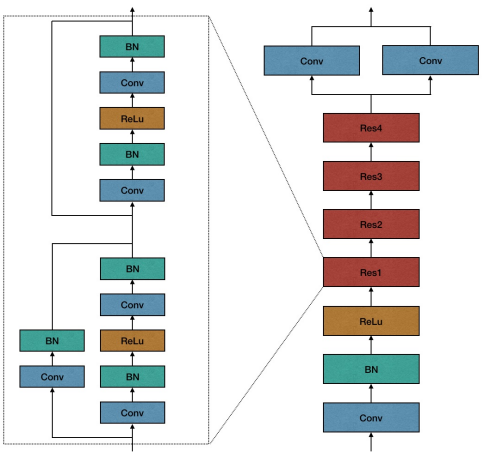

Figure 2. The structure of Res-18 C3D Network.

II. RELATED WORKS

Many hand-crafted features have been proposed and widely used in early traditional gesture recognition methods, such as histogram of oriented gradients (HOG), finite-state machine, hidden Markov model (HMM) [6]. Other typical works include novel spatial-temporal feature named MFSK [7], SIFT-like descriptors on 3D gradient and motion spaces respectively for action recognition in RGB-D videos [8]. Besides, Klaser et al. [9] treat videos as spatio-temporal volumes and proposed a novel spatial-temporal descriptor based on HoG-based descriptors used in pattern recognition for static images.

Recently, convolutional neural networks (CNNs) [10] have made a great breakthrough on computer vision related tasks by their powerful feature extraction ability, thus the features extracted by CNNs are widely used in many action classification tasks instead of hand-crafted features for better performance. Features are extracted by 2D-CNN from the start. Wang et al. [11][2] used bi-directional rank pooling [12] to encode the spatial and temporal information of videos. L. Pigou et al. [13] introduced temporal convolutions for gesture recognition in videos. On the other hand, C3D [14] model is deveploed and provides a better performance. Li and Miao et al. [1] use C3D model and won the first place twice in ChaLearn LAP Multi-modal Isolated Gesture Recognition Challenges 2016 [15] and 2017 [16]. Cangoz et al. [17] and Liu et al. [3] used also C3D model for continuous gesture recognition.

Regarding the temporal information of the video sequences, Long Short Term Memory (LSTM) networks is a common choice to gesture recognition. For instance, Zhu et al. [18] introduced convolutional LSTM for spatio-temporal feature maps. Chai et al. [19] used 2S-RNN (RGB and Depth) for continuous gesture recognition.

Moreover, traditional N-shot learning approaches are extended to one shot gesture learning. Wan et al. introduced simulation orthogonal matching pursuit (SOMP) [20] and HJ Escalante et al. [21] introduced principal motion components (PMC) for one shot learning. ME Cabrera et al. [22] proposed to deal with gesture perception and production as part of the recognition problem. There is a trend using kinematic, cognitive and biomechanic characteristics of human interaction.

III. THE PROPOSED METHOD

A. Model

As main contribution, our fusion model is mainly composed of three parts: two Res-18 3D Convolutional Network [14] (for RGB and Depth video feature extraction), an LSTM Network is to extract skeleton features, and a fused layer blending all the model.

1) Res-C3D Model: We implemented a C3D Network [14] with the residual network architecture for this task. The model consists of 8 residual units. Each residual unit has two convolutional layers, and each layer contains $3\times3\times3$ filters. Batch normalization [23] layer follows convolutional layer which accelerates the training process. Similar to [24], the number of filters are set to 64, 64, 128, 128, 256, 256, 512, 512. Different form [1], two global average pooling layers with kernel size $4\times1\times1$ and $1\times7\times7$ are performed and replaced the fully-connected layer. The output of different average pooling layer will be simply added together to get the final result.

2) Skeleton LSTM Network: We feed more data than the others [16] to improve the accuracy of the model, which including body keypoints, hands keypoints, and the angle information. Each frame of videos is converted to a vector with 241 elements, and each video is normalized with 32 frames. The bi-directional LSTM has four inter-middle hidden layers with 60 (or 120) size. The dropout rate is set to 0.1, with the 0.001 learning rate and 256 batch size is.

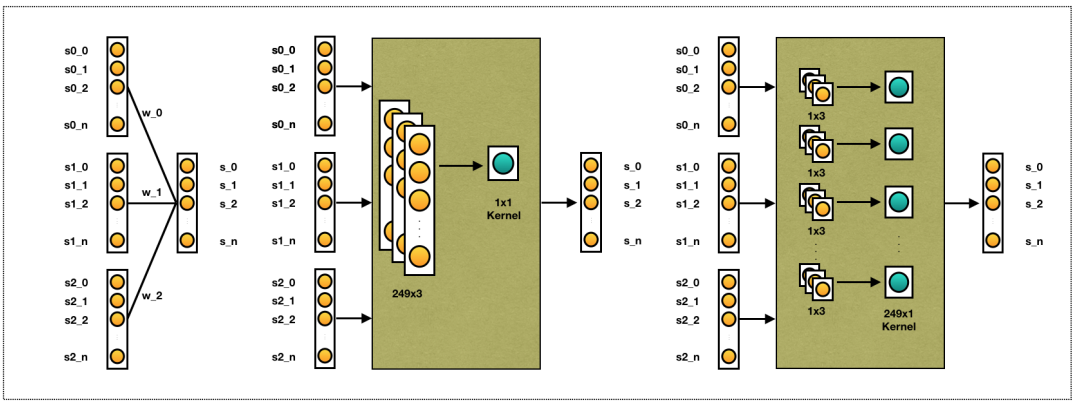

Figure 3. There are three different fusion models. From the left to right: Strategy 1: w_0, w_1 and w_2 were fixed; Strategy 2: the fusion weights are adaptive but unified for each model; Strategy 3: each class has its own voting weight.

B. Fused Layer

Result fusion usually uses 1:1 voting directly to obtain an ensemble result. Wang et al. [4] first set voting weights as 1:2 to improve accuracy, and Duan et al. [5] first use this strategy to deal with depth and RGB videos. We name it *Strategy 1*. Another strategy (*Strategy 2*) is that the voting weights of each model can be adjusted, however, they are unified for each model, so that it is hardly suitable for various cases. Noticed that each model has its own advantages or disadvantages for each class (it means it does not make sense to share the weight for all classes), our key idea here is to train adaptive weights of each voting sub-classifier being of advantage to a certain class in place of fixed weights via a convolution layer, namely fused layer. We name it (*Strategy 3*) (see Fig 3).

Denote the fusion input and output by x_i and y_i, where i is the index of classes. There are three different models in our model, let $x_i = x_{1i}, x_{2i}, x_{3i}$, thus we have output

$$y_i = f_i(\{x_{1i}, x_{2i}, x_{3i}\}),$$

where f_i is a 1×1 convolutional kernel with 3 channels, and the fusion layer consists of n 1×1 convolutional kernels with 3 channels, such that the number of fusion weights is $n \times 3$. Each $f_i()$ will be determined in the training phase.

In implementation details, we concatenate the result on a additional dimension and then connect to a 1-dimension convolutional layer which the kernel size is 1. We obtain different weights for each class by training the fused layer that consists of n convolutional kernels without biases. The fused layer has s simpler structure than a fully connected (FC) layer, although it is more complex than strategy 2. The advantage to FC layer is that we use our prior knowledge, in special, we know exactly the meaning of each value, so that we know that if there is relationship between the input and output, and how reduce the weights of the FC layer. This helps us to improve the accuracy and limit the complexity of our model. The difference between these two structures can be illustrated in Figure 3.

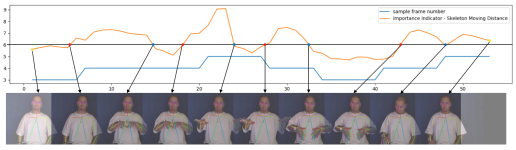

Figure 4. An example of our indicator to a RGB gesture video. Orange: The importance indicator obtained by the moving distance of the Skeleton in each frame; Blue: The sampled frame number by the weight sampler; as illustrated, the sampler chooses more frame in the video section which has higher importance.

C. Data Preparation, Preprocessing and Augmentation

1) Skeleton Data Excavation: We use convolutional pose machines [25] to generate the skeleton data. However, Openpose may fail to detect person because some keypoints are occlusive. We first extract the same keypoints at different times and concatenate them to a sequence in chronological order. Then, linear interpolation is employed to generate the keypoints in the middle missing points of time sequence. Furthermore, we explore some extra significant features from skeleton data, including the relative position and the angle of keypoints. Inspired by Guo et al. [26], we subtract the mean of the keypoints' position from all the keypoints' position to extract the relative displacement for micro emotion recognition. We further obtain the relative angle of the skeleton. We pick 121 terms to concatenate with the position vector. The 121 dimensional vector includes 10 fingers, which should be the most important part of the gesture, each one has 10 (C_5^2) value to represent the angle. The remaining 21 (C_7^2) items are made by 7 keypoints of the upper body, including wrist, elbow, shoulder, and chest.

2) Data Normalization: A fixed dimension of input data is required in the C3D model and LSTM model, in details,

Figure 5. From left to right: 1. Raw RGB video data; 2. Raw depth video data; 3. Skeleton data derived from Openpose (it is visualized by a RGB image); 4. RGB video data after masking; 5. Depth video data after masking.

all the input examples should be 32 frames as the average number of frames is 32 for each IsoGD example, and it is really the best choice validated by tuning testing. Besides, it is fixed 32 to accommodate the GPU memory of Nvidia Titan X. Thus data normalization is necessary. For the videos in which frames are less or more than 32, *uniform normalization* with upsampling and downsampling is used to uniform the number of frames.

A given video V with n frames needs to be compressed or extended to k frames. There are two situations as follows:

1) For case $n > k$, we split the video V into a k section video set V_S averagely, where $V_S = \{V_1, V_2, V_3..., V_k\}$. For each piece in the video set V_S, we randomly choose one frame as the representation of the sub-video fragment. Finally we concatenate all the represent frames and make them as the result of the normalization.
2) For case $n < k$, we randomly choose $k - n$ frame in the video, then repeat them follow by themselves.

Although the frame distribution remains unchanged, it may not be a suitable to train the model directly. For most of the videos with a single gesture, the frames in different phases share different importance. For example, in Figure 4, the section which the performer throws his hand out and the section he put down his hand on his belly should be more noticed by the model because they include the key frame of the gesture. For this case it is unfair to perform uniform sampling for the normalization but necessary to use different weights for non-uniform sampling.

We split the video V into a m section video set V_{set} averagely, where $V_{set} = \{V_1, V_2, V_3..., V_m\}$. For each $i-th$ piece V_i in the video set V_{set}, we define a importance function F to indicate the importance of V_i, and randomly choose n frame using uniform sampling method as mentioned above, such n can be obtained by

$$n = N(\frac{F(V_i) \cdot k}{\sum_{j=1}^{m} F(V_j) \cdot m}),$$

where N rounds float number to an integer. Inspired by Miao et al. [1] who use average optical flow as the importance for detecting the key frames of the video, we first use the displacement of the skeleton keypoints as the indicator. As discussed above, skeleton keypoints (include 18 body keypoints, 21 left-hand keypoints, and 21 right-hand keypoints) are extracted by Openpose, denote key point set $kp_{set} = OP(f)$ for each frame f where $kp_{set} = \{kp_1, kp_2, kp_3..., kp_{60}\}$. Define a distance function D as below:

$$D(kp1_{set}, kp2_{set}) = \sum_{i=1}^{len}(|kp1_{ix} - kp2_{ix}|+|kp1_{iy} - kp2_{iy}|),$$

so the indicator function F can be define as

$$F(V) = \ln \sum_{i=1}^{len-1} D(OP(f_i), OP(f_{i+1})).$$

The nature logarithm function ensures the distance of a pair of sections is not too big. Figure 4 shows the distinct relationship between our indicator and the keyframe position in the video and how the sampler chooses sample number via our skeleton keypoints indicator.

3) Data Augmentation: In this chapter, masking, re-sampling, and horizontal flipping are used for video data augmentation, and re-sampling, horizontal flipping, and random perturbation are used for skeleton data augmentation.

I. Masking

A difficulty for gesture recognition is that the videos are very easy to be affected by the negative influence of the distractors, such as the background, cloth and the body and so on. One way to resolve this problem is to mask [27] the location except important information such as hands. The masking area of the hands (in details, the keypoints of the elbow and wrist) is calculated by the skeleton data. We drew a rectangle using these two keypoints for each hand and block the other region of the image in each frame of depth videos and RGB videos.

II. Re-sampling

Data need to be normalized before being fed into our models. However, weight (non-uniform) sampling and uniform sampling have their respective advantage and disadvantage. For uniform sampling, it may miss some of the keyframes and get more frames on the section which does not include any useful features. For weight sampling, some action pauses in temporal dimension may be ignored because there are not enough importance. We do re-sampling before training the model in every epoch, and randomly choose one method to implement normalization. This makes sure that our models can learn the feature both in the keyframes and in temporal dimension.

III. Flipping and shifting

To augment the training dataset, horizontal flipping and shifting are used for skeleton sequence and video data augmentation. For the video data, we flip each frame in the video and randomly choose a small range to do the shifting. As fig 5, the images after masking exist black meaningless borders, thus we can do the shifting to let the convolutional kernel in the Res-C3D network learn better. For the skeleton data, we flip both the key point coordination and the keypoints index, which achieve the effect of horizontal flipping.

Table I
C3D AUGMENTATION STRATEGY

Data	Masking&Perturbation	Accuracy	Promotion
RGB	√	0.5903	0.04
RGB	×	0.5563	-
Depth	√	0.6447	0.06
Depth	×	0.5814	-
Skeleton	√	0.3539	0.03
Skeleton	×	0.3278	-

IV. Perturbation

Because one skeleton training sample has only 32 vectors of 241 dimension (it means the number of data has been considerably reduced), we use another way to augment the amount of training data. We randomly set some keypoints coordination in the sequence to the invalid state, then use the fixed point approach as mentioned above to fill the invalid value. After that, we calculate the angle as we have done before. This guarantees there are sufficient data for training, such that the performance of skeleton LSTM model can be improved.

IV. EXPERIMENT

A. Dataset

Two public datasets are used to evaluate our proposed model: the ChaLearn LAP large scale isolated gesture dataset (IsoGD) [28] and RGBD-HuDaAct dataset (HuDa) [29].

1) IsoGD: IsoGD is a large-scale isolated gesture dataset derived from the ChaLearn Gesture Dataset (CGD). IsoGD contains 47,933 RGBD gesture videos divided into 249 kinds of gestures performed by 21 individuals.

2) HuDa: HuDa includes 30 different humans and each performing the same 12 activities, e.g. 'eat a meal'. Also, it includes a random 'background' activity. All performed in a lab environment. Around 5,000,000 frames in total.

B. Implementation

All of our models are implemented by pyTorch. Three NVIDIA TITAN XP GPUs are employed for training. There are some tricks, we pre-trained RGB Res-C3D data and Depth video data respectively to obtain two pre-trained models, and then tuned these two models based on Depth video data and RGB Res-C3D data respectively. The learning rate is initialized to 1e-3 and decayed by 10 every 30 epochs. The weight decay is set to 5e-5 and the momentum is 0.9. The spatial size of the inputs is restricted to 112×112. For the skeleton LSTM model, we use Adam for training, and the learning rate is initialized to 1e-3.

C. Experimental result

1) Data Augmentation on IsoGD: Table I gives the results with/without data augmentation, which are evaluated on the dataset of IsoGD. Masking is used in RGB and Depth C3D models and the perturbation is used for Skeleton data. Our strategy gained performance improvements 0.04, 0.06 and 0.03 for RGB, Depth, and Skeleton respectively.

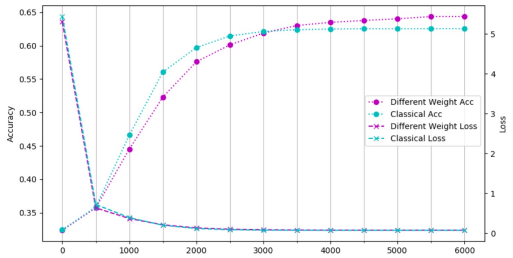

Figure 6. The Comparisons on different fusion model. The left y-axis indicated the accuracy while the right one indicated the loss. In the case of convergence, the adaptive weight can achieve better accuracy.

Table II
COMPARISONS WITH THE ISOLATED GESTURE RECOGNITION CHALLENGE

#	Team	Valid	Test	Model Number
1	ASU [1]	**0.6440**	0.6771	7 (4*C3D+2*TSN+1*SVM)
2	SYSU_ISEE	0.5970	0.6702	6 (5*VGG16+1*LSTM)
3	Lostoy	0.6202	0.6597	2 (2*C3D)
4	AMRL [2]	0.6081	0.6559	12 (8*CNN+4ConvLSTM)
5	XDETVP [18]	0.5800	0.6047	3 (3*ConvLSTM)
-	Baseline [5]	0.4917	0.6726	6 (4*CNN+2C3D)
-	Ours	0.6437	**0.6842**	**3 (2*C3D+1*LSTM)**

2) Fusion result on IsoGD: Tabel III shows the comparisons with different fusion strategy on IsoGD dataset. As illustrated, skeleton LSTM did not perform well in IsoGD dataset, thus we did not use it for 1:1:1 fusion directly. All single models' accuracy is less than 0.65, and the fusion strategy allowed them to increase by 0.01 to 0.03. 1:2 (proposed by Duan [5]) and achieved the accuracy 0.66, and our voting strategy 2 verified that we can train the voting rate and reach the same level. Our final model shows that this strategy can get more improvement if we use different weights rather than a fixed weights.

3) Comparison on IsoGD: Our proposed method obtained 0.6842 accuracies on the IsoGD dataset, and this result is better than the best result (ranks 1st) in the 2017 Chalearn LAP isolated gesture recognition challenge. Besides, our model is simpler than the other state-of-the-art model, in other words, we obtained the same accuracy with simpler models. The comparison of the result and the model complexity are illustrated in Table II. Compared with their methods, we successfully reduce the model number and obtain a better accuracy in this dataset.

4) Fusion result on HuDaAct: Table IV shows the comparisons with different fusion strategy on HuDaAct dataset. Since HuDaAct dataset only has 13 classes, the Skeleton LSTM model perform better than C3D models. Compare

Table III
COMPARISONS WITH DIFFERENT FUSION STRATEGY ON ISOGD

Fusion Method	Valid	Test
Skeleton LSTM	0.3178	0.3539
RGB Res-net	0.5621	0.5903
Depth Res-net	0.5635	0.6447
2:1 (RGB+Depth)	0.5991	0.6349
1:1 (RGB+Depth)	0.6189	0.6580
1:2 (RGB+Depth)	0.6112	0.6605
Strategy 2 [Fig.3.2]	0.6253	0.6669
Strategy 3 [Fig.3.3]	**0.6437**	**0.6842**

Table IV
COMPARISONS WITH DIFFERENT FUSION STRATEGY ON HUDAACT

Fusion Method	Accuracy
Skeleton LSTM	0.9138
RGB Res-net	0.7672
Depth Res-net	0.7759
1:1:1 [Fig.3.1]	0.8621
Strategy 2 [Fig.3.2]	0.9310
Strategy 3 [Fig.3.3]	**0.9569**

with the result on the IsoGD dataset, intuitively, it does not make sense to set fixed weights of voting for every task. Voting strategy 2 improves 0.02 accuracy compared with the best single model, the skeleton LSTM. Voting strategy 3 finally reaches accuracy 0.9569, which is better than the voting strategy 2.

5) Comparison on HuDaAct: Table V shows the comparison on HuDaAct Dataset. As illustrated, our proposed approach outperforms other old competing algorithms by a large margin with over 0.1 and 0.15 accuracy and catching up with the state-of-the-art model. This also verified that our model has strong generalization capacity–it is not only be used on gesture recognition but also on some other video-based classification tasks. Furthermore, as the median of frames in each HUDAACT example is 256 , we cannot normalize all the examples to 256 frames as it would lead to huge GPU memory demands that would be in excess of the memory of Nvidia Titian XP (total 12GB) used in experiments. Hence, each HUDAACT example was normalized to 32 frames such that this experiment can be completed, whereas 32 frames are insufficient for HUDAACT examples feature exaction and learning. In contrast, the median of frames in each IsoGD example is 32 such that we proposed method can achieve state-of-the-art performance. For this reason, the performance of our method is little lower than the approach [5].

Table V
COMPARISONS ON RGBD-HUDAACT DATASET

Method	Accuracy
STIPs(K=512) [Laptev and Lindeberg] [30]	0.7977
DLMC-STIPs(M =8) [Ni] [29]	0.7949
DLMC-STIPs(K=512,SPM) [Ni] [29]	0.8148
2SCVN-3DDSN [Duan et al.] [5]	0.9783
Ours	**0.9569**

V. CONCLUSION

In this paper, we proposed a relatively simple model for gesture recognition and achieved state-of-the-art accuracy in the IsoGD dataset. We chiefly developed a fusion scheme for blending features via a convolution layer (called fused layer) to improve fusion performance. Besides, we introduced some ways for extraction of the potential information of training data with a variety of data enhancement technology for both skeleton and RGB-D videos.

However, our model is still not an end-to-end model and has to be trained step by step. Meanwhile, ChaLearn LAP IsoGD still have a large room for improvement, we still have a lot of work to enhance the accuracy of the model.

ACKNOWLEDGMENT

This work was partially supported by Science and Technology Development Fund of Macau (No. 112/2014/A3, 151/2017/A, 152/2017/A), the National Key Research and Development Plan (Grant No. 2016YFC0801002), the Chinese National Natural Science Foundation Projects ♯61502491, ♯61473291, ♯61572501, ♯61572536, ♯61673052.

REFERENCES

[1] Q. Miao, Y. Li, W. Ouyang, Z. Ma, X. Xu, W. Shi, X. Cao, Z. Liu, X. Chai, Z. Liu *et al.*, "Multimodal gesture recognition based on the resc3d network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3047–3055.

[2] H. Wang, P. Wang, Z. Song, and W. Li, "Large-scale multimodal gesture recognition using heterogeneous networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3129–3137.

[3] Z. Liu, X. Chai, Z. Liu, and X. Chen, "Continuous gesture recognition with hand-oriented spatiotemporal feature," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3056–3064.

[4] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 20–36.

[5] J. Duan, J. Wan, S. Zhou, X. Guo, and S. Z. Li, "A unified framework for multi-modal isolated gesture recognition," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1s, p. 21, 2018.

[6] J. Wan, Q. Ruan, G. An, and W. Li, "Gesture recognition based on hidden markov model from sparse representative observations," in *Signal Processing (ICSP), 2012 IEEE 11th International Conference on*, vol. 2. IEEE, 2012, pp. 1180–1183.

[7] J. Wan, G. Guo, and S. Z. Li, "Explore efficient local features from rgb-d data for one-shot learning gesture recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 8, pp. 1626–1639, 2016.

[8] J. Wan, Q. Ruan, W. Li, G. An, and R. Zhao, "3d smosift: three-dimensional sparse motion scale invariant feature transform for activity recognition from rgb-d videos," *Journal of Electronic Imaging*, vol. 23, no. 2, p. 023017, 2014.

[9] A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *BMVC 2008-19th British Machine Vision Conference*. British Machine Vision Association, 2008, pp. 275–1.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[11] P. Wang, W. Li, S. Liu, Z. Gao, C. Tang, and P. Ogunbona, "Large-scale isolated gesture recognition using convolutional neural networks," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 7–12.

[12] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 773–787, 2017.

[13] L. Pigou, A. Van Den Oord, S. Dieleman, M. Van Herreweghe, and J. Dambre, "Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video," *International Journal of Computer Vision*, pp. 1–10, 2015.

[14] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4489–4497.

[15] H. J. Escalante, V. Ponce-López, J. Wan, M. A. Riegler, B. Chen, A. Clapés, S. Escalera, I. Guyon, X. Baró, P. Halvorsen *et al.*, "Chalearn joint contest on multimedia challenges beyond visual analysis: An overview," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 67–73.

[16] J. Wan, S. Escalera, X. Baro, H. J. Escalante, I. Guyon, M. Madadi, J. Allik, J. Gorbova, and G. Anbarjafari, "Results and analysis of chalearn lap multi-modal isolated and continuous gesture recognition, and real versus fake expressed emotions challenges," in *ChaLearn LaP, Action, Gesture, and Emotion Recognition Workshop and Competitions: Large Scale Multimodal Gesture Recognition and Real versus Fake expressed emotions, ICCV*, vol. 4, no. 6, 2017.

[17] N. C. Camgoz, S. Hadfield, O. Koller, and R. Bowden, "Using convolutional 3d neural networks for user-independent continuous gesture recognition," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 49–54.

[18] L. Zhang, G. Zhu, P. Shen, J. Song, S. A. Shah, and M. Bennamoun, "Learning spatiotemporal features using 3dcnn and convolutional lstm for gesture recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3120–3128.

[19] X. Chai, Z. Liu, F. Yin, Z. Liu, and X. Chen, "Two streams recurrent neural networks for large-scale continuous gesture recognition," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 31–36.

[20] J. Wan, Q. Ruan, W. Li, and S. Deng, "One-shot learning gesture recognition from rgb-d data using bag of features," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2549–2582, 2013.

[21] H. J. Escalante, I. Guyon, V. Athitsos, P. Jangyodsuk, and J. Wan, "Principal motion components for one-shot gesture recognition," *Pattern Analysis and Applications*, vol. 20, no. 1, pp. 167–182, 2017.

[22] M. E. Cabrera, N. Sanchez-Tamayo, R. Voyles, and J. P. Wachs, "One-shot gesture recognition: One step towards adaptive learning," in *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*. IEEE, 2017, pp. 784–789.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[25] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4724–4732.

[26] J. Guo, S. Zhou, J. Wu, J. Wan, X. Zhu, Z. Lei, and S. Z. Li, "Multi-modality network with visual and geometrical information for micro emotion recognition," in *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*. IEEE, 2017, pp. 814–819.

[27] Z. Liu, X. Chai, Z. Liu, and X. Chen, "Continuous gesture recognition with hand-oriented spatiotemporal feature," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3056–3064.

[28] J. Wan, Y. Zhao, S. Zhou, I. Guyon, S. Escalera, and S. Z. Li, "Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 56–64.

[29] B. Ni, G. Wang, and P. Moulin, "Rgbd-hudaact: A color-depth video database for human daily activity recognition," in *Consumer Depth Cameras for Computer Vision*. Springer, 2013, pp. 193–208.

[30] I. Laptev, "On space-time interest points," *International journal of computer vision*, vol. 64, no. 2-3, pp. 107–123, 2005.