Hand Gesture Recognition with 3D and Bilinear CNN Model

Hanwen Bi
College of Engineering and Computer Science
Australian National University
Canberra, Australia
hanwen.bi@anu.edu.au

Chaoyun Gong
College of Engineering and Computer Science
Australian National University
Canberra, Australia
chaoyun.gong@anu.edu.au

***Abstract*—In this paper, we look into the problem of hand gesture recognition which is becoming important in the automotive user interface. Hence, a bilinear 3D CNN model is developed, using RGB and depth data. The spatio-temporal data augmentation is employed to reduce over-fitting and improve model generalization. To further test the bilinear 3D CNN model, we compare its performance with the performance of 3D CNN architecture, without the bilinear classifier, on VIVA challenge dataset. The bilinear 3D CNN model achieves 72.0% classification accuracy.**

***Index Terms*—Hand gesture recognition, bilinear model, depth analysis, human-machine interaction, 3D convolutional neural networks**

I. Introduction

Recently, devices and techniques in human-computer interaction (HCI) have been significantly developed. In particular, the touchless interface in cars by using hand gestures can enhance the driver's safety and comfort. For example, passenger can control the car entertainment system by hand gestures, which allows the driver to focus on driving. However, some challenges are remaining in this area, such as the changes of lighting condition and time duration. In this paper, we are concerned with developing a robust real-time hand gesture recognition model with high classification accuracy.

In our study, we first reconstructed the two channels 3D CNN architecture proposed by Molchanov et al. [1], with different optimizers and hyperparameters. To further improve the classification accuracy and running speed, motivated by Lin et al. [2], we introduced a novel model that combines the 3D CNN architecture and the bilinear classifier. The Input of these two models is 4D spatio-temporal volumes, containing both depth and intensity information. Besides, an effective data augmentation method was introduced to reduce potential overfitting and improve generalization of the gesture classifier. We demonstrated two network architecture with data augmentation for spatio-temporal input in training, on VIVA challenge's dataset to test the performance of two network architecture.

The rest of the report will be as follows. We introduce the progress of recent researches in the hand gestures recognition and related fields in section II. In section III, we describe the spatio-temporal data augmentation, two channels 3D CNN architecture, and 3D bilinear CNN architecture. In section IV, the experimental results of two network architectures were presented. Section V is the conclusion.

II. Related research studies

As the quality and quantity of RGB and depth data from cameras have been enhanced in the recent decade, the research interest in hand gestures recognition was increased. Relevant literature related to hand gestures recognition and bilinear CNN is summarized below.

A. Hand gesture recognition

Some feature extractor and video descriptors are introduced in [3] and [4]. In general, hand shapes and motion features are important in hand gestures recognition, as they indicate the temporal changes of gestures, which was proposed in [5] and [6]. Methods of extracting pose, which is significantly helpful in this topic, were proposed in [7] - [10].

As the price high-quality sensors is becoming lower, the methods with multi-sensor were proposed in [1, 11, 12, 13]. Neverova et al. utilized the RGB and depth (RGBD) data and the skeletal data for recognizing 20 Italian sign language gestures [11]. Molchanov et al. successfully combined the RGBD data and data from radar sensors, using convolutional neural networks (CNNs) [12]. Ohn-Bar and Trivedi experimented various hand gestures classifiers by using RGBD data [13]. Molchanov et al. developed a two-channel 3D CNN model with RGBD data, achieving the highest classification accuracy in VIVA challenge dataset [1].

High diversity of training data is essential for CNNs model. Therefore, data augmentation is essential to reduce overfitting and improve generalization, especially when training data contains a limited variety. To enhance the performance of the video-based human activity recognition model, a spatial augmentation on the video frame was introduced in [14]. Pigou et al. developed a video-based data augmentation method by temporally translating the video frames [15]. Molchanov et al. proposed on-lone and off-line data augmentation approaches, containing various spatial and temporal transformation [1].

B. Bilinear Model

In 2000, to model two factor variations for images, Tanenbaum and Freeman [15] introduced bilinear models. Pirsiavash

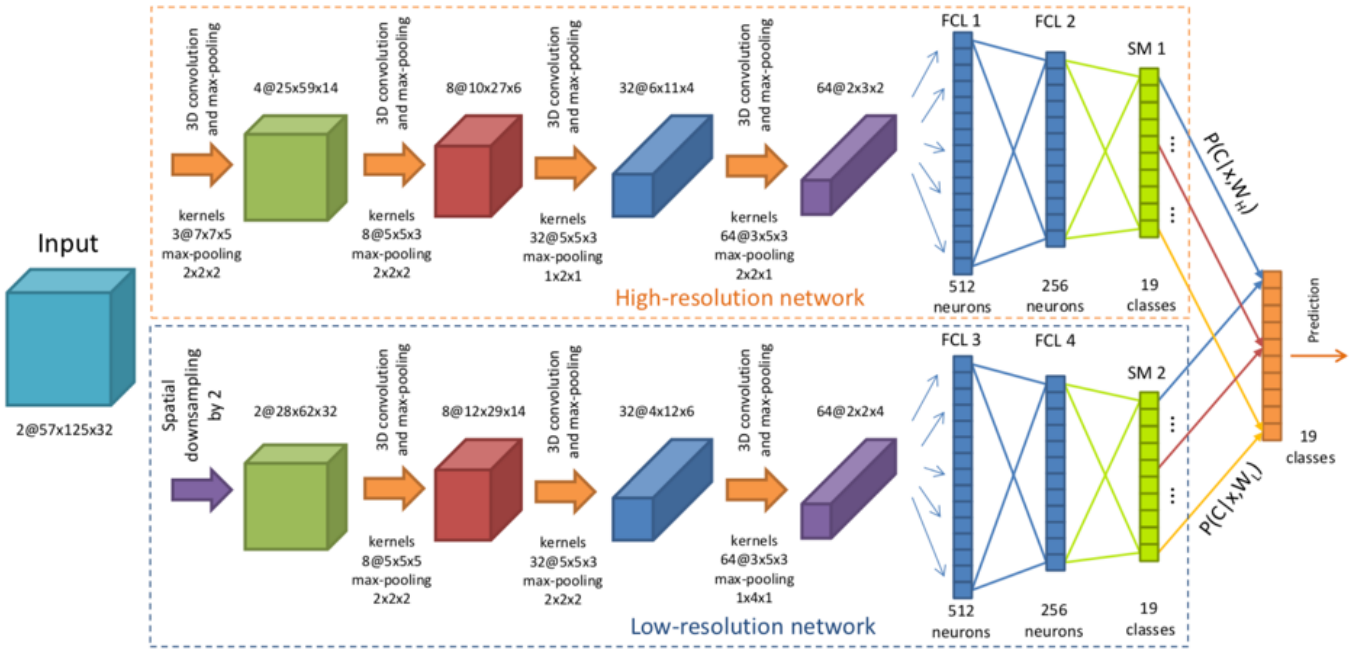

Fig. 1. The architecture of the 3D CNN classifier. The input of each channel is a combination of intensity image gradient and depth values, with the size of 57 × 125 × 32. The classifier is composed of two streams: a high-resolution network (HRN) and a low-resolution network (LRN). The final output of the classifier is the inner product of the two class-membership probabilities vectors outputted from LRN and HRN [1].

et al. [16] proposed the bilinear classifier that defines the bilinear classifier as a product of two low-rank matrices. Lin et al. [2] proposed a two-steam architecture with the bilinear classifier model for image classification.

III. Method

We used a two-stream 3D convolutional neural network and a two-stream 3D convolutional neural network with the bilinear classifier for hand gesture recognition. Subsection A briefly describes the VIVA challenge's hand gesture dataset used in this paper. Subsection B introduces the preprocessing steps and spatio-temporal data augmentation. The architectures of two different CNN models and training settings are described in subsection C to E.

A. Dataset

The VIVA challenge's hand gesture dataset (Fig. 2) was used to test the performance of models proposed by this state-of-the-art. The VIVA challenge's hand gesture dataset includes 19 different classes by 8 subjects inside a vehicle, in total 885 intensity and depth video sequence with various lighting conditions [1]. Both intensity and depth channels of hand gestures videos are recoded by Microsoft Kinect device, recoding the hand gestures performed by the driver's right hand or the front passenger's left hand, involving the motion of the hand and/or finger.

B. Preprocessing and spatio-temporal data augmentation

The preprocessing contains two kinds of operation, spatial operation and temporal operation. Temporally, because hand

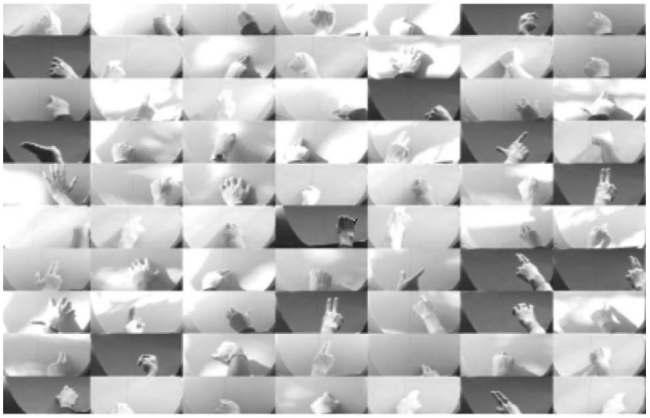

Fig. 2. Examples of different hand gestures in VIVA challenge dataset, in a large variation of illumination condition [13].

gesture videos have various time duration, we re-sampled each sequence to 32 frames using nearest neighbor interpolation (NNI) to normalize the length of sequences. For the spatial processing, we firstly down sampled the original intensity and depth images in each frame, from 115 × 250 pixels to 57 × 125 pixels. After that, we use 3 × 3 Sobel operator to derive the gradient of intensity images to improve the robustness [1], before normalizing each video sequence to be zero mean and unit variance.

The spatio-temporal data augmentation (Fig. 3) includes spatial and temporal augmentations. For the temporal operation, inspired by [1,17], we randomly reversed the or-

der of frames of both intensity and depth channels. Spatial augmentation mainly includes four kinds of operations: *(a) horizontal mirroring:* randomly horizontal mirroring images of each sequence, (b) affine transformations: rotation ($\pm 10^o$), translation (± 5 pixels along x axis, and ± 10 pixel along y axis), scaling ($\pm 20\%$), *(c) spatial elastic deformation* [1, 18]: using Gaussian filter with a=4 std =8 to smooth images, *(d) random drop-out:* randomly setting 50% pixels of images to zero value.

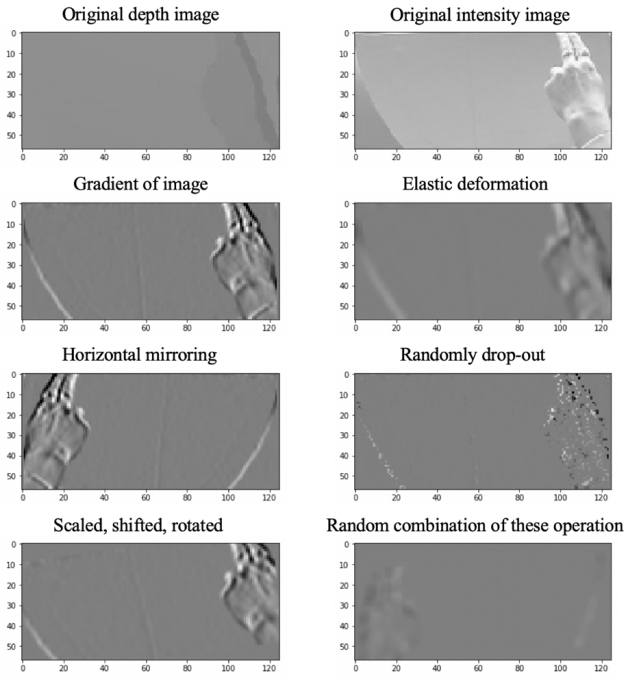

Fig. 3. The spatio-temporal data augmentation. The figure shows the original intensity and depth images and the spatio-temporal data augmentation outputs for different transformation

C. Architecture of two-channel 3D CNN model

The two-channel of the 3D CNN model (3D CNN) [1], named high-resolution network (HRN) and low-resolution network (LRN) respectively (Fig. 1). The input of 3D CNN model is a 4D volume, combining depth and intensity video sequences. The main differences of LRN and HRN is the first layer setting. For LRN, the first layer is 3D max pooling layer, aiming to down sample the inputs (reducing the resolution of inputs), while, for HRN, the first layer is a 3D convolutional layer. Except the first layers of LRN and HRN, both LRN and HRN contain 3 unit layers (one unit layer contains a 3D convolutional layer with ReLU activation function, a max-pooling layer, and a batch normalization layer), 2 fully connected layers, and one softmax function. Details of 3D CNN architecture is shown in figure 1. In the decision-making stage, the 19-dimension outputs of LRN and HRN were multiplied together to get the final output and prediction.

D. Bilinear 3D CNN model

The technology in Bilinear CNN model has been applied in this project [2]. Bilinear CNN model has a wonderful performance at fine-grained recognition [2]. It can simplify the gradient computation [2] and reduce the complexity of network in this project by using Bilinear vector to replace all the fully connection layers. The architecture of Bilinear CNN has been shown in the following figure. It uses two convolutional neural networks to extract the features of input image. The output of the CNN is a size axb image with n channels. Thus, the output image has axb locations and each location has n features. Then applying outer product for each location of the two images from two CNNs (nx1*1xn) and get size nxn features for each location. Then, doing sum-pooling to sum axb numbers of size nxn features and get size nxn Bilinear vector. The bilinear vector can be used to describe the input image. After that, Support vector machine (SVM) can be used to classify the different categories. In this project, the SoftMax has been attached after the bilinear vector to classify the different categories.

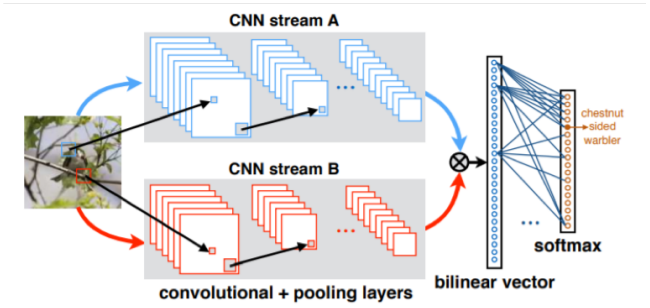

Fig. 4. A typical bilinear CNN model, containing two parallel CNN streams A and B. To obtain the bilinear vector, outputs of steam A and B are multiplied using outer product at each location of the image. The bilinear vector is passed through the softmax layer to produce the final output of the classifier [2].

Comparing the 3D convolutional neural networks in the previous paper [1] and the Bilinear CNN model, the architectures are similar. Both of them apply two convectional neutral networks which include convolutional layers, pooling layers, activation functions and normalization layers. Then the output of the two CNN are combined together by using fully connection layers or bilinear vector in 3D convolutional neural network and Bilinear CNN model respectively.

To achieve the high performance of Bilinear CNN model by using the 3D convolutional neural network, the two models have been combined together. The new architecture of the neural network has been shown in Fig.5.

E. Training

We used cross-entropy loss as the cost function for both models. The optimizer is Adam with mini-batch of 32 training samples for both channels of the two models. For the learning rate, we first initialized it to 0.002, and reduce it by multiplying it by a factor of 0.8 for every 50 epochs. We terminated the network training after processing 500 epochs.

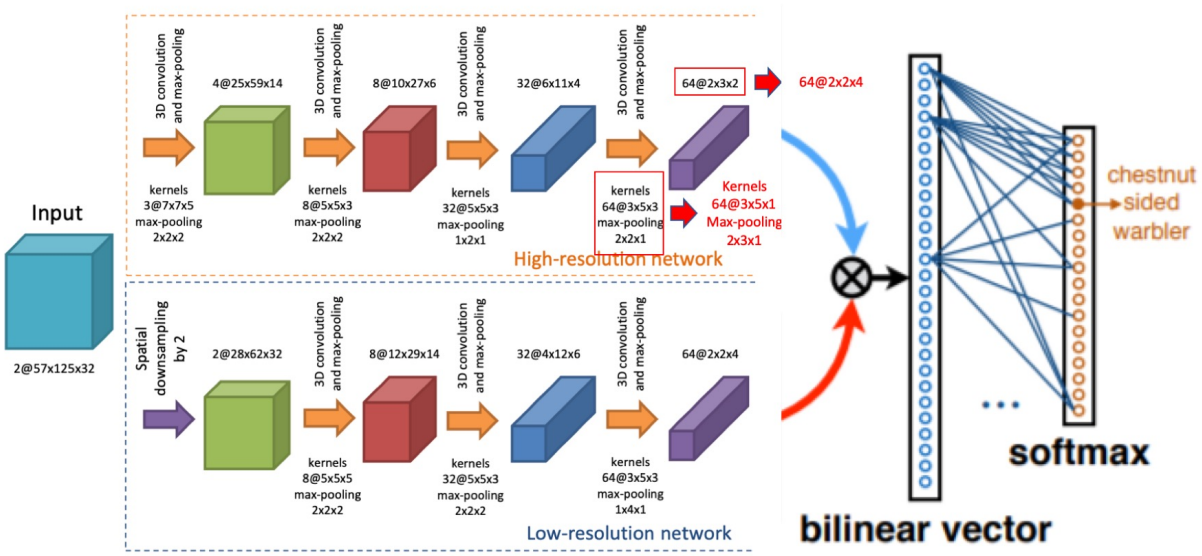

Fig. 5. Architecture of bilinear 3D CNN classifier. The inputs to each channel are a combination of intensity image gradient and depth values, with the size of 57 x 125 x 32. Outputs of LRN and HRN are multiplied using outer product at each location of the image to obtain the bilinear vector. After passing through the softmax layer, the final output of the classifier can be obtained.

IV. Experimental Result

The model has been trained in one Nvidia GeForce RTX 2080Ti GPU with 11GB memory. The total number in the dataset is 885 and there are 19 classes. k-fold cross-validation has been applied for this model. The has been divided into 8 folds randomly with a specific random seed. 7 folds are used for training and one fold is for validation. Thus, there are 8 models and 8 different accuracies. The epoch is 500. The learning rate is dynamic with initial value of 0.002. It reduces to the 80% of the previous value after 50 epochs each time. The optimizer is Adam, and the loss function is Cross-Entropy Loss. And for the data argumentation, both Offline and Online argumentation in paper have been applied in the training set [1].

Fig.6 and Fig.7 show that the loss and accuracy of training and validation set by 500 epochs in the randomly set of 7 folds for the training and one fold for validation. It can be observed that the model does not overfit after 500 epochs from Fig.6. In Fig.7 the accuracy of training set can over 95% steadily and the accuracy of the validation set can achieve 74% after about 450 epochs. The reason why the validation accuracy is fluctuant is that the small size of the validation set. There are just about 110 data in the validation set. Thus, the result will include a little burstiness. And the fluctuation is reasonable in this situation.

Then the validation accuracies of the 8 models are 74.4%, 80.5%, 71.6%, 79.6%, 67.6%, 73.3%, 60.7%, 68.0% respectively. Comparing with the previous model which is constructed depending on Molchanov, Gupta, Kim and Kautz's paper [1] with both Offline and Online data argumentation and 8-fold cross validation. The Table.1 shows the result.

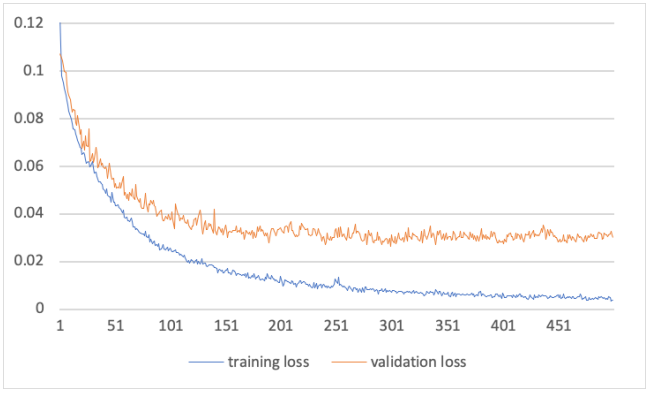

Fig. 6. Training and validation loss - epochs. The epochs are 500, the training loss can achieve 0.004 and the validation loss can reach 0.028

Table.1 shows the accuracy of the 3D convolutional neural network from Molchanov, Gupta, Kim and Kautz's paper [1] and the new model (3D CNN + Bilinear vector). The mean of accuracy is the mean of the eight validation accuracy of the eight models by using 8-fold cross-validation. And the Std

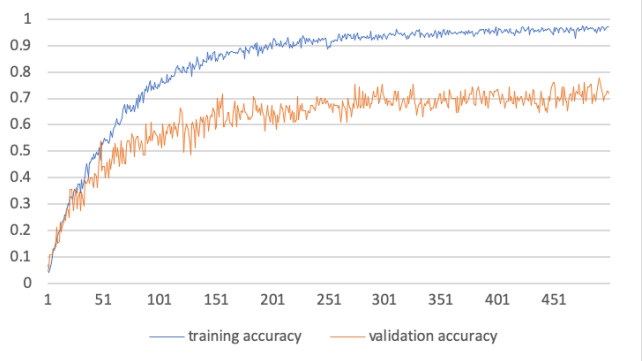

Fig. 7. Training and validation accuracy - epochs. The epochs are 500, the training accuracy can achieve 97% and the validation accuracy can reach 74%

TABLE I
THE ACCURACY OF PREVIOUS 3D CNN AND 3D CNN + BILINEAR

Accuracy	3D CNN	3D CNN + Bilinear
Mean	67.5%	72.0 %
Std	5.3%	6.6 %

is the standard deviation of the eight accuracies. The mean accuracy of the previous 3D convolutional neural network is 67.5% and the mean accuracy of the new model can achieve 72.0%.

Table.2 shows the training time of the 3D convolutional neural network from Molchanov, Gupta, Kim and Kautz's paper [1] and the new model (3D CNN + Bilinear vector). The epochs are 500. And the GPU RTX1080Ti has been used to train the two model separately. The training time for the two models is almost the same (about 9 hours). The new model cost 31 seconds less than the previous model. Thus, the cost of training time between two models can be ignored.

Table.3 shows the confusion matrix of the validation set for the new model. We can see the 19 classes of hand gestures in the confusion matrix. The value in the table is the accuracy percentage of the recognition. the accuracy of each class is from 60% to 90%. It still has the confusion in swipe and scroll comparing with the result in Molchanov, Gupta, Kim and Kautz's paper [1]. And our model is also confused in rotate.

V. DISCUSSION

A. Comparison of the previous 3D CNN model and the new 3D CNN +Bilinear vector model

The reason why we try to apply Bilinear in the 3D convolutional neural network from Molchanov, Gupta, Kim and Kautz's paper [1] is to increase the performance of the model.

TABLE II
THE TRAINING TIME OF PREVIOUS 3D CNN AND 3D CNN + BILINEAR

Model:	3D CNN	3D CNN + Bilinear
Training time	9h 03m 59s	9h 03m 28s

TABLE III
THE CONFUSION MATRIX OF THE VALIDATION SET, THE LETTER IN THE TOP LEVEL: L - LEFT, R - RIGHT, D - DOWN, U - UP, CW/CCW - CLOCK/COUNTER-CLOCK WISE [1]

class	1. Swipe R	2. Swipe L	3. Swipe D	4. Swipe U	5. Swipe V	6. Swipe X	7. Swipe +	8. Scroll R	9. Scroll L	10. Scroll D	11. Scroll U	12. Tap-1	13. Tap-3	14. Pinch	15. Expand	16. Rotate CCW	17. Rotate CW	18. Open	19. Close
1	75					8			7				10						
2		65	5		5				15			4			6				
3			60			5		9			8		4	4			4		6
4	5			85				10											
5					80					7				13					
6						90		2			5					3			
7			4		5		70			6			6			4		10	
8								81		5			8		4			2	
9		9			4				71			6		4	4	2			
10			2	2		5	2			83			3						3
11			7					10			60		9			11			3
12			10					2				58	15			4		11	
13	2		2		5			2					79			6		4	
14		4			4			4			4		5	63		4		7	5
15	2		4					4					5		75	6		4	
16								11		4		6				73	6		
17									7		4		7		3		75		4
18				11														79	
19			4			6		7			13			3					67

TABLE IV
COST TIME FOR EACH VIDEO IN GPU AND CPU, THE GPU IS NVIDIA GEFORCE RTX 2080TI GPU WITH 11GB MEMORY, AND CPU IS A 12 CORES CPU WITH 47.2G MEMORY AND 16.0G SWAP SPACE

	GPU (RTX1080Ti)	CPU
Cost time/each:	0.08s	0.96s

Compare the accuracies of the two models, the new model which combines 3D convolutional neural network and Bilinear vector has much better performance than the previous 3D CNN model. The data pre-process, data argumentation and parameters of training are all the same. The average accuracy can increase 4.5%. And the training time of the model is also a little less than the previous model. Thus, the new model is successful.

B. Implementation of the real time recognition

The input dataset includes RGB video and depth video. Thus, both RGB camera and depth camera are needed for real time recognition. We apply the trained model to recognize the hand gesture for a one-second video. The cost time is shown in the Table.2.

If GPU has been used to do the real time recognition, the delay can be ignored. It is almost the real time. When using CPU. After doing the hand gesture, the user still needs to wait about one second to get the response. It can be applied in some situations which have low requirement of real time, like the car multimedia system. In the real situation, to reduce the waiting time for the users, the two videos which are captured from the real-time camera should have an interval. For the system with CPU, the interval should be more than 1 second. And each captured video is about 1 second. Thus, to avoid the cut-off of one whole hand gesture, the video from the camera needs to be pre-processed. Another system is used to detect the hand motion. When it has detected hand motion, those frames

from the camera will be captured and are transported to the 3D Convolutional model to do recognition. If the system has GPU, the algorithm can be easy, and the recognition will be much faster.

VI. Conclusion

At the beginning, we have implemented the 3D convolutional neural network model in Molchanov, Gupta, Kim and Kautz's paper [1], applying the model on the VIVA hand gesture dataset. The final average validation accuracy can be 67.5%. Then, to achieve better performance, the Bilinear vector is used to replace the fully connected layers for the previous 3D convolutional neural network. The final average validation accuracy can achieve 72%. It is much higher than the previous model. And it is very close to the result in Molchanov, Gupta, Kim and Kautz's paper (77.5%) [1].

For the future work, because of the lack of dataset, we need to add more data or to do some other argumentations for the data. Besides, the model should be more robust for different hand gestures. The hyperparameters of the model will be modified.

References

[1] Molchanov, P., Gupta, S., Kim, K. and Kautz, J., 2015. Hand gesture recognition with 3D convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 1-7).

[2] Lin T Y, RoyChowdhury A, Maji S. Bilinear cnn models for fine-grained visual recognition[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1449-1457.

[3] P. Trindade, J. Lobo, and J. Barreto. Hand gesture recogni- tion using color and depth images enhanced with hand an- gular pose data. In IEEE Conf. on Multisensor Fusion and Integration for Intelligent Systems, pages 71–76, 2012.

[4] J. J. LaViola Jr. An introduction to 3D gestural interfaces. In SIGGRAPH Course, 2014

[5] W. Heng, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," Int. J. Comput. Vis., vol. 103, no. 1, pp. 60–79, May 2013.

[6] S. Sathyanarayana, G. Littlewort, and M. Bartlett, "Hand gestures for intelligent tutoring systems: Dataset, techniques and evaluation," in Proc. IEEE Int. Conf. Comput. Vis. Workshops, 2013, pp. 769–776.

[7] C. Keskin, F. Kirac, Y. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in Proc. IEEE Intl. Conf. Comput. Vis. Workshops, 2011, pp. 1228–1234.

[8] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D human skeletons as points in a lie group," in Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2014, pp. 1–8.

[9] I. Kapsouras and N. Nikolaidis, "Action recognition on motion capture data using a dynemes and forward differences representation," J. Vis. Commun. Image Represent., vol. 25, no. 6, pp. 1432–1445, Aug. 2014.

[10] G. Evangelidis, G. Singh, and R. Horaud, "Skeletal quads: Human action recognition using joint quadruples," in Proc. IEEE Intl. Conf. Pattern Recog., 2014, pp. 1–6.

[11] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Multi- scale deep learning for gesture detection and localization. In ECCVW, 2014.

[12] P. Molchanov, S. Gupta, K. Kim, and K. Pulli. Multi-sensor System for Driver's Hand-gesture Recognition. In AFGR, 2015.

[13] E.Ohn-BarandM.Trivedi.Handgesturerecognitioninreal time for automotive interfaces: A multimodal vision-based approach and evaluations. IEEE Trans. on Intelligent Trans- portation Systems, 15(6):1–10, 2014.

[14] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In NIPS, pages 568–576, 2014.

[15]] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. Neural computation, 12(6):1247–1283, 2000.

[16] H.Pirsiavash,D.Ramanan,andC.C.Fowlkes.Bilinearclas- sifiers for visual recognition. In NIPS. 2009.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pages 1097–1105. 2012.

[18]] P. Y. Simard, D. Steinkraus, and J. C. Platt. J.c.: Best prac- tices for convolutional neural networks applied to visual doc- ument analysis. In Int. Conference on Document Analysis and Recognition, pages 958–963, 2003.