RESEARCH-ARTICLE

Skeleton Based Dynamic Hand Gesture Recognition using LSTM and CNN

AAMRAH IKRAM, Beijing Institute of Technology, Beijing, China

YUE LIU, Beijing Film Academy, Beijing, China

Open Access Support provided by:

Beijing Film Academy

Beijing Institute of Technology

Published: 05 August 2020

Citation in BibTeX format

IPMV 2020: 2020 2nd International Conference on Image Processing and Machine Vision
August 5 - 7, 2020
Bangkok, Thailand

Skeleton Based Dynamic Hand Gesture Recognition using LSTM and CNN

Aaahm Ikram
Beijing Engineering Research Center of Mixed Reality and Advanced Display, School of Optics and Photonics, Beijing Institute of Technology, 5 South Zhongguancun Street'Haidian, Beijing, China.

Yue Liu
CFVE of Beijing Film Academy, 4 Xitucheng Road, Haidian, Beijing, China

ABSTRACT

Dynamic Hand Gestures offer a natural, non-verbal way of communication that can substitute other communication modalities like verbal speech and script writing. Not only for the voice command, hand gestures also play significant role in Augmented Reality (AR), Virtual Reality (VR) and games. There are some factors like computational cost, flexibility and recognition accuracy that can impact the incorporation of hand gestures in these fields. In this paper, a Dynamic Hand Gesture Recognition (DHGR) approach is propose that is based on Convolutional Neural Network (CNN) and long-short term memory (LSTM). This system is trained to execute the sequence of 3D input data along with the velocities and positions information learned from Leap Motion Controller (LMC).When evaluated and compared with state-of-art DHGR methods, this architecture shows relative high accuracy of 98%.

CCS CONCEPTS

• **Human-centered computing** → Human computer interaction (HCI); HCI design and evaluation methods; Usability testing.

KEYWORDS

Dynamic Hand Gestures Recognition (DHGR), Leap Motion Controller (LMC), Convolutional Neural Network (CNN)

ACM Reference Format:
Aaahm Ikram and Yue Liu. 2020. Skeleton Based Dynamic Hand Gesture Recognition using LSTM and CNN. In *2020 2nd International Conference on Image Processing and Machine Vision (IPMV 2020), August 05–07, 2020, Bangkok, Thailand.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3421558.3421568

1 INTRODUCTION

Hand Gestures are the common Human Computer Interface (HCI) because of effectiveness and flexibility. Gestures over a contact surface are common way to Interact with all those devices which are touched enabled. Whereas, touchless hand gestures are now being used in controlling game consoles also in the new fields of technology like AR, VR and holographic views. Dynamic Hand Gesture Recognition (DHGR) is a technology which tells us that what a hand sequence is trying to convey. In the past decade HCI has improve considerably. Still it is a challenge because of intra-class variance due to the complex recognition algorithms and variation of a same gesture performed by several persons.

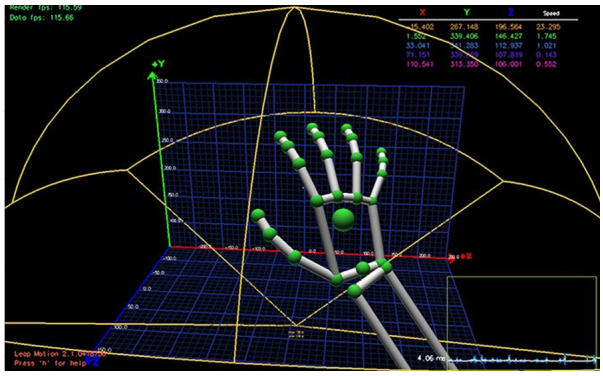

Figure 1: Hands in 3-D space as seen by Leap Motion Controller

A remarkable work has been done in this research field with the help of machine learning and with different types of depth cameras [1], [2], [3], [4]. Work done on DHGR mainly used RGDB and depth images as the main input [5],[6], [7]. Some of them used flexible motioncaptured devices such as: data glove or monocular video sensors [8] and audio stream [9].

In this paper we presented an approach that executes skeleton based hand gesture technique. Hand skeletal information is received from Leap Motion Controller (LMC) as shown in Figure 1. As compared to the other gesture recognition devices LMC delivers better outputs. Smedt.et.al [10], presented depth based approach for dynamic hand gesture with skeletal information. Yuanrong. et.al [11] provides a feature based SVM classifier to recognize number gestures in the air. In the earlier works, Histogram of Oriented Optical Flow (HOOF) terminology has been used in which HOOF creates featured tracks of 3D trajectories of hand movement. This is a useful technique for 3D hand signature sequences. For sign language recognition [12], calculated data from several LMC are processed and fused with the help of Kalman filter, and classification has done using Hidden Markov Model (HMM).

All these above mention approaches are mainly engineered features that can point out the difference in gesture in the 3D information. For the sake of automatic and high level recognition from input 3D data can be achieved with deep learning. To avoid these

IPMV 2020, August 05–07, 2020, Bangkok, Thailand

ACM ISBN 978-1-4503-8841-2/20/08…$15.00
https://doi.org/10.1145/3421558.3421568

Table 1: Eight hand gestures Dataset from LMDI [16].

G	E	T	P	S/L	S/R	C/W	AC/W
100	100	100	100	100	100	100	100

engineered features, we have introduced such a deep learning approach for HGR. Using a de-noising approach data can be directly or indirectly input to the LSTM and CNN. We have used the LMDI dataset of eight hand gestures [16]. To access this information LMC SDK has been used.

2 CONVOLUTIONAL NEURAL NETWORK AND DATASETS

2.1 Convolutional Neural Network For image classification

CNN is very successful neural network [13], [14]. CNN is made up of three different layers as: convolutional layer, pooling layer and a fully connected one. Mostly CNN's architecture is made by stacking of above mentioned layers. Features that are extracted from these three types of CNN layers are hierarchical. High level layers collect and learn features with intangible data that is very helpful for classification while low-level layers collect low-level features [15]. Features mapping layer is to examine input regarding to spatially-local correlation. Different kernels are built up using several feature maps. We can calculate the i^{th}feature map in l^{th}layer is calculated by (1). Whereas,$y_i{}^{l-1}$is the i^{th} feature map in l^{th}layer.$W^l{}_{i,j}$is the weight matrix for every$y_j{}^{l-1}$, also weight matrix connects to$y_i{}^{l-1}$ the feature map$y_j{}^{l-1}$. $b_i{}^l$is bias of the i^{th} feature map in thel^{th} layer. A non-linear activation function *f(.)* used such asauto encoder. The * is denoted as the convolution operator [15]. Every first layer is followed by sampling layer an in this project we have used Softmax function. To minimize the time of calculation and to decrease the size of features sampling layer is used.

$$y_i^t = \sum_j f\left(w_{ij}^t * y_j^{l-t} + b_i^l\right) \quad (1)$$

Whereas, $\boldsymbol{y_i{}^{l-1}}$ is the $\boldsymbol{i^{th}}$ feature map in $\boldsymbol{l^{th}}$ layer. $\boldsymbol{W^l{}_{i,j}}$is the +weight matrix for every $\boldsymbol{y_j{}^{l-1}}$, also weight matrix connects $\boldsymbol{y_i{}^{l-1}}$ to the feature map $\boldsymbol{y_j{}^{l-1}}$. $\boldsymbol{b_i{}^l}$ is bias of the $\boldsymbol{i^{th}}$ feature map in the $\boldsymbol{l^{th}}$ layer. A non-linear activation function *f(.)* used such asauto encoder. The * is denoted as the convolution operator [15]. Every first layer is followed by sampling layer an in this project we have used Softmax function. To minimize the time of calculation and to decrease the size of features this layer is used.

2.2 Datasets

We used a dataset for Dynamic Interactive Gesture with LMC that provides sequences of hand skeleton. This dataset consists of eight gestures such as: expand (E), grab (G), pinch (P), tap (T), clockwise (C/W), anticlockwise (AC/W), swipe left (S/L) and swipe right (S/R) named as LMDI [16].

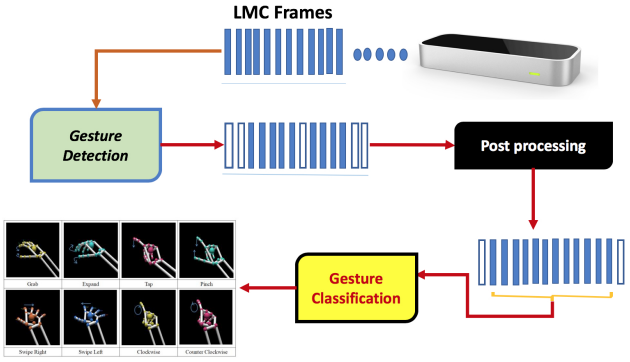

Figure 2: System Architecture

Table 1. shows the total number of gestures that are documented and manually explained. These gestures were the results of 800 sequences, each gesture is performed 10 times by 10 participants. This hand data is built at 50 frame per second for single hand information. A transition gesture is introduced to capture hand movement between any of two interaction gestures. Table 1 shows the total number of gestures that are documented and manually explained. These gestures were the results of 800 sequences, each gesture is performed 10 times by 10 participants. This hand data is built at 50 frame per second for single hand information. A transition gesture is introduced to capture hand movement between any of two interaction gestures.

3 PROPOSED MODEL

Figure 2. shows that Gesture Recognition (GR) is made up by Gesture Detection (GD) and Gesture Classification (GC). The hand movements are received by LMC, which provides 3D positions and velocities of the figures in the form of generated time series frames. Gesture Detection (Section 3.1) describes whether the frame lies inside a gesture or not. The frame sequences that are labeled are then passed to post-processing modules. Post processing (Section 3.2) is a module that segregates sequence of frames that are extremely similar and form a large group inside the frame. The passed sequence of gestures is then moved to Gesture Detection modules. This module decides that particular sequence of gesture is present in the predefined set of gesture or not.

3.1 Gesture Detection

Gesture Detection modules takes input frames from LMC and evaluate that which time series frame matches to one of predefined gestures.

This is done by sequence tagging problem, such as the frames which are within the gesture are named as "Inside" and the frames those are not in the gesture are declared to be "Outside" as depicted

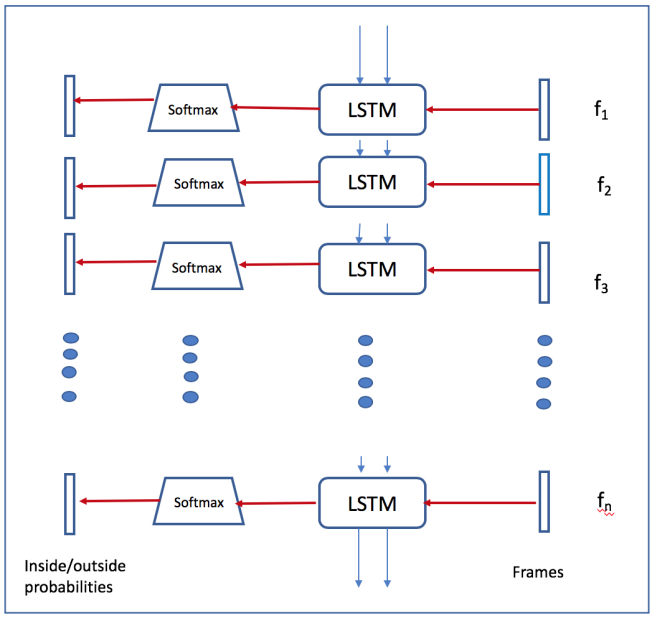

Figure 3: Gesture Detection Sequence

in Figure 3. In order to overcome the Vanishing gradient problem in Recurrent Neural Network (RNNs) [17], input is also given to the RNN that is implementing LSTM (Long Short-Term Memory) units [18]. LSTM cell receives three types of inputs such as: Current frame (velocities and finger positions), the previous LSTM cell output and the content of the memory cell from the previous time step [18]. Logistic regression model receives output from LSTM cell at every instant and then classifies the frames that whether it lies inside or outside the gesture. It is difficult to predict a gesture by watching starting and ending frames, it is required to study the few next frames for this look ahead frames are introduced. Subsequently, when GD modules produced a label at time t, this relates the frame l to the past i.e. at position k-l. For this work we used l=*20* frames, that is actually quarter of the length of the faster gesture (Swipe) in the Hand gesture dataset. If we see this in real time scale, it will relate to a delay of *20 *10 = 200ms* (that is trivial as lag perception).

The trained GD model also labeled a small number frames in middle of gesture as outside the gesture. These such labels can produce gaps and cause breaking up in gesture, that is damaging the overall efficiency of HGR. Also there are some circumstances in which starting and ending frames of gesture are not detected. Here important point is recall rather than precision. To resolve these issues three post processing steps has been used [19]:

- The initial stage joins any two groups of frames marked as inside frames, if they are parted by at most 10 frames, selecting the value 10 by inspecting the system output on a development set.
- The second stage is to point out the frames as outside, for this any inside regions that are less than 20 frames are marked as outside.
- The last step is to add 10 frames on each side of the sequences pull out by GD model.

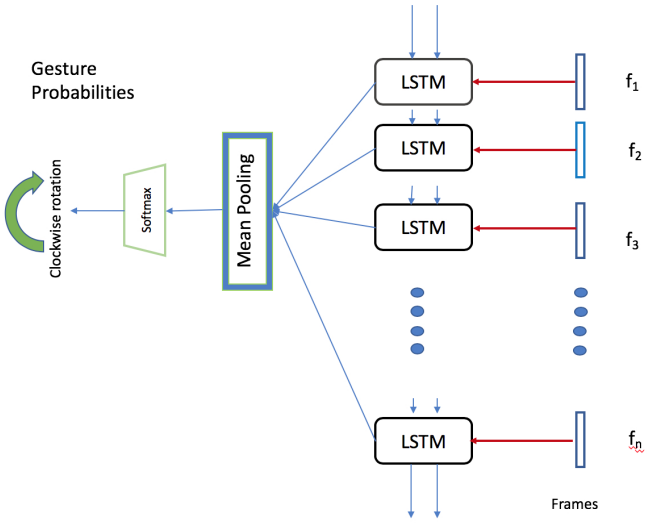

Figure 4: Gesture classification with LSTM.

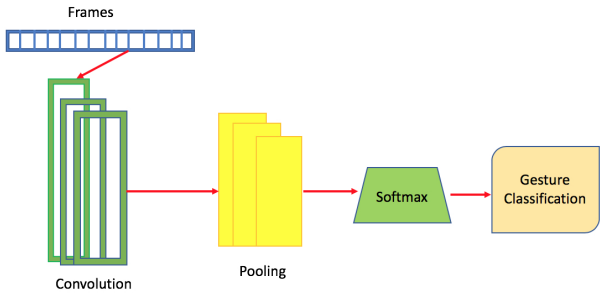

Figure 5: Gesture classification with CNN.

3.2 Gesture Classification

The gesture Classification (GC) module is to classify the gesture segments and post-processed by the GD module and put it into one of the predefined types of gestures with the help of Softmax model presented in Figure 4. Softmax has a limitation to receive a particular number of input, however, input segments have variable number of frames depending on the type of gesture.

To avoid this constraint a mean pooling layer is introduced before passing the output to softmax based LSTM network or CNN as depicted in Figure 5. In LSTM technique the same configuration is used as described in GD model, whereas; in CNN output is first mean pool and then sent to Softmax model. Here we introduced a denoising auto encoder [20] to train the filter.

3.3 Denoising Auto Encoder (DAE)

We trained it to rebuilt a clear input from a noisy or destroyed one. To do this an input $y$is corrupted and get a destroyed version of y by stochastic mapping:

$$\tilde{y} \sim q_b\left(\frac{\tilde{y}}{y}\right) \tag{2}$$

Table 2: Gesture detection percentage with Gesture detection and post processing.

Modules	Precision	Recall	F1-frame	Accuracy
Gesture Detection	85.4	70.3	75.4	90.2
Gesture detection + post processing	80.2	88.7	86.1	93..3

Table 3: Gesture classification in terms of Precision, Recall and f_1 measure

LSTM	G	E	T	P	S/L	S/R	C/W	AC/W
Precise	97.4	98.6	98.7	97.6	99.1	99.5	100	100
Recall	97.9	98.5	96.2	98.1	97.2	96.4	99.3	99.5
f_1-measures	97.1	98.2	98.1	96.5	99.3	98.8	98.4	99.1

This corrupted input is than mapped to the basic auto encoder $x = f_\theta(\tilde{y}) = (w\tilde{y} + b)$. From this equation we can construct a new equation (also from schematic diagram we can see the processes) $z = g\theta' = s(w'x + b)$. Parameters are trained over a training set i.e. to have z as close as possible to the uncorrupted input y so that we can minimize average reconstruction error$L_H(y,\ z) = H(B_y||B_z)$. The basic difference is that z is a deterministic function of ỹ except y and result in stochastic mapping of y.

4 EXPERIMENRAL RESULTS AND EVALUATIONS

We have used k–fold cross validation criteria to train GD and GC. The dataset of 800 sequences are scuffled and divided into 10 folds. This scheme worked in way such that 9 out of 10 folds are used for training and testing from which 8 fold are for training and 1 is for testing, while the remaining one is for the validation purpose. We repeated it for 9 times to use it as a testing data. These test results are pooled in a group of nine folds to have an overall assessment. To make sure that test data are not visible in training by GD and GC, we evaluate them independently using same training and testing for both. A real gesture boundaries are defined for testing and training of GC section.

The Adadelta, with one gradient update per sequence is used to minimize cross-entropy cost function for training of GD and GC. These models are trained up to 600 epochs to fulfill the initial stopping standards. To find out an average performance validation data is tracked whereas when the efficiency is seen to be degrade the initialstopping is done. All the sequences are independent of each other, so there is no continuity of hand gesture between difference sequences. Eventually, we reset the last cellstate and last LSTM output before starting each sequence. Memory cell measurement is set to be same as the input frame vector. Using singular value decomposition, orthogonal vectors can be derived. For LSTM input these derivatives can be used to start the parameters. A faster convergence during training can be achieved by this initializing technique Sax et al. [20]. The initialized the parameters for Softmax layer with univariate Standard Gaussian Distribution and L2 using random values are regularized a weight decay of 1e-4. For a maximum of 1,000 epochs on 10,000 segments without labels of 20 frames, filters are trained for CNN architecture of GC module. Samples are randomly taken from LMC and passed over a denoising auto encoder with noise probability of 0.2.

Table 2. is presenting the GD results in reference to post-processing. Inside frames includes accuracy, memory and f_i – measurement. As it is obvious that post-processing can enhance the recall significantly but impose a little damage for precision. The accuracy for GC in CNN and LSTM implementation is shown in Table 3. From these results we can see LSTM performance is better, therefore, we used it for final HGR.

The whole HGR is done by implementing trained GC module on the output of GD module on every test fold as: eight training folds are used to train GD module; using the real gesture boundaries GC trained on those 8 folds; on the test fold GD module is working; the sensed gesture is post-processed and worked as a test input for GC. Here we have implemented two condition whether the recognized gesture is correct or not: A) at least 50% part of the sequences from the system gesture lies within a real gesture; and B) the label given by the system is the one that is label of real gesture.

A generalized HGR results ate presented in Table 4. This system is fast enough to be implemented in real time techniques because there an unnoticeable lag between the classification results and the gesture completion.

5 INTERACTIVE DEMONSTRATION AND INTERACTION (IDE)

An IDE is designed and Implemented in order to examine and give an idea about the performance of user's interaction in virtual game using DHGR algorithm. The IDE is implemented specification as: Intel(R) Core (TM) i7-6950 CPU @ 3.2 GHz, 3001 MHz, 10 Core(s), 20 logical processor(s) and NVIDIA GeForce GTX 750. Software environment is Mi Microsoft Windows 10 Pro OS, Unity 3D 2018.3.19f1 for Windows along with Leap motion core assets v2.3.1.

The IDE is made up of three scenes, scene one and scene two are implemented using keyboard to have information about the user's hand and to select the scene. In the third Scene user can interact through LMC. Scene is a starting interface in which user selects whether to start interaction according to the interface information. If wants to enter to Scene 2 press "Enter" and if not press "Delete" as shown in Figure 6.

Table 4: Gesture recognition Results of overall system.

Precision	Recall	F1-measures
97.9	99.1	97.0

Figure 6: Scene 1, Starting Interface

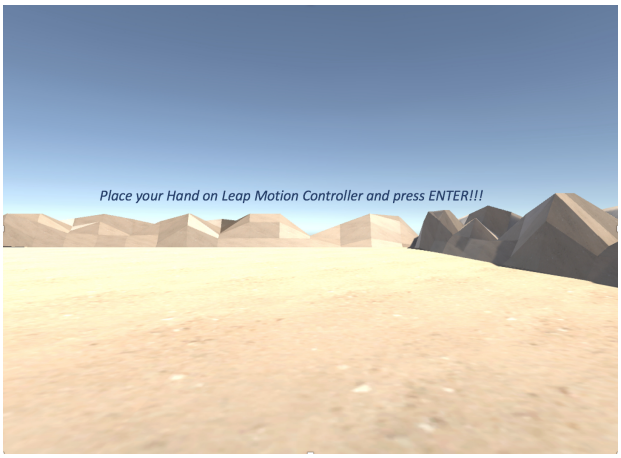

Figure 7: Scene 2, Hand size Acquiring Interface

Figure 7. shows the scene 2 that is an interface to acquire hand information and ask the user to open his hands and place them over the LMC (Scene 3 will be open once the LMC receives 30 frames of Hand Information). Scene 3 is an interactive interface that is a puzzle game of 6 pieces. With the hand gesture user need to select and organize the pieces and drag them to puzzle panel in order to complete the picture. User can drag the pieces, expand, rotate clockwise and anti-clockwise, swipe left and right. The interaction processes using different hand gesture is presented by Figure 8

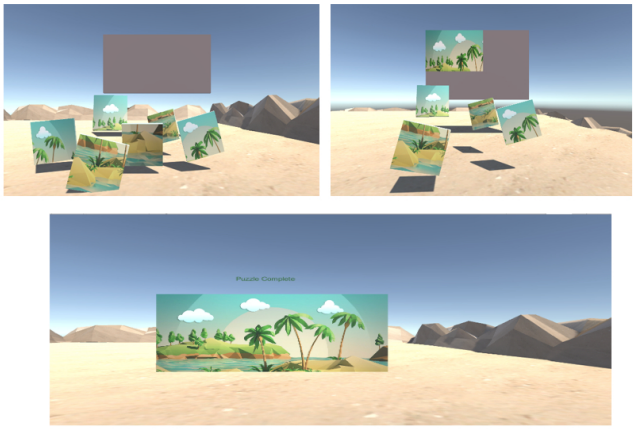

Figure 8: (a) Pinch gesture: Show the pieces. (b) Swipe Right/Left gesture: Next/Previous pieces.

6 CONCLUSION

In this paper, we have proposed a gesture recognition approach that acquires 3D data from LMC. The CNN and LSTM architectures are implemented, obtained results showed that the module using LSTM is showing better performance as compare with CNN. These models are trained up to 600 epochs to fulfill the initial stopping standards. To find out an average performance validation data is tracked whereas when the efficiency is seen to be degrade the initial stopping is done. This architecture is highly accurate up to 98% to deal with real time recognition (without visible lag). A virtual scene is also developed to understand the interaction of hand gestures with virtual objects. In future, we will evaluate this system by using a large number of dataset and gesture types. From such a dataset we can examine the system performance while switching from one user to another.

ACKNOWLEDGMENTS

This work has been supported by the National Key R&D Program of China (No. 2016YFB1001502) and National Science Foundation of China (61631010).

REFERENCES

[1] Voronkov, A. 2014. Keynote talk: EasyChair. In Proceeedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (pp. 3-4). ACM. B. Ionescu, D. Coquin, P. Lambert, and V. Buzuloiu, "Dynamic hand gesture recognition using the skeleton of the hand," EURASIP Journal on Advances in Signal Processing, vol. 2005, no. 13, pp. 2101–2109, 2005.
[2] H. Francke, J. Ruiz-del Solar, and R. Verschae. 2007. Real-time hand gesture detection and recognition using boosted classifiers and active learning, in Pacific-Rim Symposium on Image and Video Technology, Santiago, Chile, , pp. 533–547.
[3] N. H. A-Q. Dardas. 2012. Real-time hand gesture detection and recognition for human computer interaction, Ph.D. dissertation, Universite d'Ottawa/University

of Ottawa.
[4] Y.Yin, 2014. Real-time continuous gesture recognition for natural multi modal interaction," Ph.D. dissertation, MIT.
[5] Guijin Wang, Xuanwu Yin, Xiaokang Pei, and Chenbo Shi. 2013. Depth estimation for speckle projection system us- ing progressive reliable points growing matching," Ap- plied optics, vol. 52, no. 3, pp. 516–524.
[6] Chenbo Shi, Guijin Wang, Xuanwu Yin, Xiaokang Pei, Bei He, and Xinggang Lin. 2015. High-accuracy stereo matching based on adaptive ground control points," IEEE Transactions on Image Processing, vol. 24, no. 4, pp. 1412–1423.
[7] Eshed Ohn-Bar and Mohan Manubhai Trivedi. 2014. Hand gesture recognition in real time for automotive inter- faces: A multimodal vision-based approach and evalua- tions, IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 6, pp. 2368–2377.
[8] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kih- wan Kim, Stephen Tyree, and Jan Kautz. 2016. Online de- tection and classification of dynamic hand gestures with recurrent 3d convolutional neural network," in Proceed- ings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4207–4215.
[9] N. Neverova, C. Wolf, G. Taylor, and F. Nebout. 2016. Mod- drop: Adaptive multi-modal gesture recognition," IEEE Transactions on Pattern Analysis and Machine Intelli- gence, vol. 38, no. 8, pp. 1692–1706, Aug.
[10] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. 2016. Skeleton-based dynamic hand gesture recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1–9.
[11] Y. Xu, Q. Wang, X. Bai, Y.-L. Chen, and X. Wu. 2014. A novel feature extracting method for dynamic gesture recognition based on Support Vector Machine," in IEEE International Conference on Image Process- ing. Paris, France: IEEE, October, pp. 437–441.
[12] K.-Y. Fok, N. Ganganath, C.-T. Cheng, and C. K. Tse. 2015. A real-time asl recognition system using leap motion sensors," in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC). Xi'an, China: IEEE, September, pp. 411–414.
[13] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. 2015. The application of two-level attention models in deep convolutional neural network for fine-grained image classification," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 842-850.
[14] T. X. Luong, B.-K. Kim, and S.-Y. Lee. 2014. Color image processing based on Nonnegative Matrix Factorization with Convolutional Neural Network," 2014 International Joint Conference on Neural Networks (IJCNN), pp 2130-2135.
[15] J. Yang, Y. Zhao, J. C.-W. Chan, and C. Yi. 2016. Hyperspectral image classification using two-channel deep convolutional neural network," 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5079–5082.
[16] Dan Zhao, Yue Liu*, Guangchuan Li. 2018. Skeleton-based Dynamic Hand Gesture Recognition using 3D Depth Data ,IS&T International Symposium on Electronic Imaging 2018 3D Image Processing, Measurement (3DIPM), and Applications.
[17] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov.
[18] Y. Bengio, P. Simard, and P. Frasconi. 1994. Learning long-term dependen- cies with gradient descent is difficult," IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157–166, Mar.
[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. 2008. Extracting and composing robust features with denoising autoencoders," in Interna- tional Conference on Machine Learning. New York, NY, USA: ACM, pp. 1096–1103.
[20] M. D. Zeiler. 2012. ADADELTA: an adaptive learning rate method," CoRR, vol. abs/1212.5701,. [Online]. Available: http://arxiv.org/abs/1212.5701.