Design of gesture recognition system based on Deep Learning

Qiya Niu a, Yunlai Teng b, Lin Chen c
International School, Beijing University of Posts and Telecommunications, Beijing 100876, China

a Corresponding author: 2016213183@bupt.edu.cn
b2016213490@bupt.edu.cn
cchenlin6@bupt.edu.cn

Abstract. The rapid development of computer vision has had a significant impact on all areas of life. Real-time image capture using computer images for small movements has strong utility in some areas of life. Based on the EgoHands dataset, this paper uses the SSD detection method and the MobileNet neural network to establish a recognition model for students’ gestures. In the end, the model adopted in this paper achieved a score of 96.86% for mAP.

1. Introduction
With the development of computer science and technology, artificial intelligence has gradually become one of the hottest research directions in the frontiers of science and technology. As the mainstream of artificial intelligence, computer vision has become a classic field which combines science and technology with life. The framework of neural network based on deep learning is more and more widely used in this field. Among them, the use of computer vision for the application of gesture recognition is also a practical research direction, intelligent hand gesture recognition can not only be used in military, transportation, and even in the intelligent classroom system.

In this paper, we will take deep learning as the basic framework, and build a gesture recognition model by applying some advanced technologies in the field of computer vision. In this paper, the structure of hand gesture recognition will be studied mainly from the literature review, implementation, discussion and other parts.

2. Literature Review
In this part, this paper will mainly introduce the methods needed to carry out this experiment and the commonly used model methods. The method used in this paper is mainly based on convolution neural network.

2.1. Evaluation method
In this paper, the generalization ability of the model will be evaluated by experimental measurements. In this way, a test set is needed to test the model to classify the new samples, and the resulting test errors can be approximated to the generalization errors. The project takes a retention approach, which means generating training sets and test (validation) sets from a single dataset.
The preserving method divides the dataset $D$ into two sets of mutually exclusive and independent data sets, one set of $S$ is used for training to obtain the model, and the other set is used to verify the performance of the model.

$$D = S \cup T$$
$$S \cap T = \emptyset$$

(1)

After each iteration of the training model based on the training set $S$, the set $T$ can evaluate the current test error of the model. The method requires consistency of data distribution in order to avoid the introduction of additional deviations, which can be solved by stratified sampling. In order to obtain a more robust and reliable model, this project has carried out many experiments using the retention method, and the data set has an appropriate proportion division, of which nearly 80%-90% is used for training.

### 2.2. Performance measurement

The evaluation of the generalization ability of the model not only needs effective and feasible experimental estimation methods, but also needs the evaluation standard to measure the generalization ability of the model, that is, performance measurement [1].

1. **Error rate and accuracy**

   Error rate and accuracy are the two most commonly used performance indicators in classification, which can be applied to dichotomy tasks and multi-classification tasks. Error rate refers to the ratio of the number of sample errors to the total number of samples in classification testing, and accuracy refers to the ratio of the number of samples to the total number of correctly classified samples. If the prediction function is $f$, and for data set $D = \{(x_1,y_1),(x_2,y_2),...,(x_m,y_m)\}$, $y_i$ is the true symbol of sample $x_i$, then the error rate can be defined as:

$$E(f;D) = \frac{1}{m} \sum_{i=1}^{m} I(f(x_i) \neq y_i)$$

(2)

If $I(K)$ is an index function, $K$ is true when $K = 1$ and false when $K = 0$, then the precision can be defined as:

$$Acc(f;D) = \frac{1}{m} \sum_{i=1}^{m} I(f(x_i) = y_i)$$

$$= 1 - E(f;D)$$

(3)

2. **Precision rate, recall rate and mAP**

   In the field of image information retrieval and object detection, in fact, users are more concerned about these indicators, such as how much information they are interested in is retrieved or detected. In this way, precision and recall are more suitable for performance measurement.

   According to the results of the actual and predictive categories, the samples can be divided into true-positive (TP), false-positive (FP), true-negative (TN) and false-negative (FN). Table 1 shows the obfuscation matrix.

| Actual Class | Predicted Class |
|--------------|-----------------|
| 1            | 0               |
| 1            | TP  FN          |
| 0            | FP  TN          |

The precision rate $P$ can be defined as:

$$P = \frac{TP}{TP+FP}$$

(4)

The recall rate $R$ can be defined as:

$$R = \frac{TP}{TP+FN}$$

(5)

Accuracy and recall are two contradictory measures. Under normal circumstances, it is not possible for them to obtain relatively high value at the same time. In some cases, the prediction results of the model can be sorted according to the confidence level from maximum to minimum; then, the samples.
with the highest confidence are predicted in this order as positive examples, and the current recall rate is calculated after each prediction. Figure 1 shows the exact recall curve.

![Figure 1. Precision-Recall rate curve example](image)

The AP (average accuracy) can be obtained by calculating the area under the exact recall curve. In this way, the mAP (average precision) can be obtained by calculating the average value of the multilevel AP, which is the most commonly used index to evaluate the performance of the model in the field of object detection.

3. Design and Implementation

3.1. Dataset

The EgoHands dataset contains high-quality pixel level hand segmentation and all marked frames as JPEG files (720x1280px) [2]. Each of the 48 videos has 100 tagged frames, for a total of 4800 frames. In 48 different hand activities (chess, cards, puzzles, etc.) and environments (indoor, outdoor), all images were captured from a self-centered view (Google glass). As shown in figure 2.

![Figure 2 EgoHands dataset examples with frames](image)

3.2. Dataset pre-processing

In order to achieve real-time action, the project uses EgoHands data set to train the appropriate model to achieve detection. The project uses the hold-out method to split the EgoHands dataset into a training set (80 percent) and a test set (20 percent) and place them in different folders. The project then generates a bounding box for each of the two sets of images for visualization. In addition, each image folder in different environments contains a csv tag file that can be used to generate tf records in TensorFlow [3] format. TensorFlow is an open source library that supports deep learning. This project realizes the model of real-time target detection mainly through TensorFlow.
3.3. Models
To detect hand movements, real-time hand tracking and detection is the most important task to be solved. As mentioned above, deep learning has always been the dominant method in the field of object detection, because traditional rule-based methods (extracting features and using classifiers such as HOG) make the detection results not so accurate or robust. Therefore, in this module, neural network is used to construct a better model with higher detection accuracy to solve more problems such as noise interference or background change, which still exists in the traditional object detection methods.

Therefore, using the regression-based depth learning method, this project uses SSD detector combined with MobileNet, which is a lightweight CNN model that can run efficiently. The MobileNet architecture is also constructed using depth-separable convolution, which is used in the facial expression recognition model described above. MobileNet consists of 28 layers, including deep convolution layer, 1 × 1 point convolution layer, batchnorm, ReLU, average collecting layer and softmax. Figure 3 shows the MobileNet architecture.

![Figure 3 layers of MobileNet architecture](image)

Based on the EgoHands dataset and the TensorFlow deep learning framework, the project uses a transfer learning process to retrain the last layer of the SSD-MobileNet model that has been trained. Because SSD-MobileNet's original neural network architecture contains millions of parameters, the process reduces training time and provides a more efficient way for hand motion detection tasks. In addition, to reduce detection errors, the projection uses NMS to merge duplicate bounding boxes and uses a threshold to filter the actual action of raising hands.

3.4. Implementation.
The training process was completed on NVIDIA Jetson TX1 and has been introduced in the facial detection and facial expression recognition modules. It took about five hours to run 200k steps to get a manual detection model, and about 0.32 ms in real time on a MacBook Pro laptop. The current model can detect the hand and recognize the movement of the hand, which is a very common behavior.
4. Results and Discussion.
To detect the "show of hands" action, the project used SSD-MobileNet to train the hand detection model, which is the deep learning method described above. Based on the transfer learning process, the project retrained the model with a modified final layer for hand detection. In the retraining process, the total loss is reduced to 2.614 and stopped at a relatively stable state, which means that the retraining process is almost complete after 200k steps, and the error is reduced to its minimum value. Figure 4 shows the total losses of the retraining process under the TensorFlow deep learning framework.

The mAP of the hand detection model is about 96.86%, which shows that the model performs well and is suitable for real time computer vision system. Figure 5 shows the mAP diagram during the retraining process. Figure 6 shows the results of the multi-hand detection using the retraining model.

5. Conclusion
In the manual detection module, based on the EgoHands data set, the project retrained the MobileNet model with modified final layer for hand detection, and uses SSD method to improve the detection speed and accuracy. Due to some detection errors caused by repeating the bounding boxes, the item then applies NMS to merge them and uses a threshold to filter the actual hands of the objects detected in the region of interest. After 200k steps of training, the model can detect manual action, mAP is 96.86%. For future work, the project will detect more manual operations, such as playing on the phone or writing, which can be achieved by integrating multiple deep neural network (DNN) into a model. The project will seek to combine more methods with future speech or behavior recognition. In addition, the project
considers the design and implementation of visual interfaces that can be used for Web sites or mobile applications. In the future, these methods could be used in more scenarios, such as detecting students who cheat on exams, or analyzing the real-time performance of interviewees during interviews.

References
[1] Zhihua Zhou (2016). Machine Learning. Tsinghua University Press, 2016: 24-25
[2] Indiana University Computer Vision Lab. EgoHands: A Dataset for Hands in Complex Egocentric Interactions. Retrieved December 2015, from Indiana University Web site: http://vision.soic.indiana.edu/projects/egohands/
[3] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. SSD: Single shot multibox detector. In European Conference on Computer Vision (2016), Springer, pp. 21–37.
[4] Andrew G. Howard et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017