CNN with Spatial Pyramid Pooling application in dynamically recognizing human gestures

Yufan Liu, Yuwei Miao, Shuguang Wang

University of North Carolina at Chapel Hill
{yufan, yuwei, shuguang}@live.unc.edu

Keywords: CNN, shape classify, gesture detection, Spatial Pyramid Pooling

Abstract

In this paper, we propose a new approach to recognize hand pose (rock, paper, scissors) which has a better accuracy and smaller model size compared to existing models. We design a CNN model with 5 layers, specially designed kernel size to best fit the image in our dataset, and a Spatial Pyramid Pooling layer. We explicitly test the dataset with five different existing models representing the latest technology in the field and compare the result with our approach. The result shows a great precision, efficiency, and compatibility in terms of testing accuracy and model size. The trained model is used to applied in the computer camera and can detect hand shape in a real-time environment.

1 Introduction

Different human body language often corresponds to different human emotions or intentions, among which hand gesture is one of the easiest body languages to capture and recognize. Using machines to dynamically capture and recognize human gestures not only expands the range of applications of human-computer interaction (HCI), but also improves the accuracy of machines in predicting human emotions or intentions. The mapping from three kinds of gestures in Rock-Paper-Scissors game on to three different meanings is both an injection and surjection. With the bijective map from gestures on their meanings, Rock-Paper-Scissors game is suitable to be used as a research object to compare the advantages and disadvantages of hand gesture recognition models. In this paper, we apply several different gesture recognition models to rock-paper-scissors games to compare their advantages and disadvantages. Based on the comparison, we developed a Convolutional Neural Network(CNN) with Spatial Pyramid Pooling(SPP)layer. The CNN which uses SPP layer performs better accuracy with smaller model size in comparison to other existing models. Moreover, we further introduce cameras to accomplish the dynamic capture and recognition of different human gestures.

2 Data

We used the dataset named Rock-Paper-Scissors Images from Kaggle. This dataset consists of a total of 2188 hand gesture images from Rock-Paper-Scissors game. The images in this dataset are almost evenly distributed in three categories: 726 images of "Rock"(33.2%),710 images of "Paper"(32.4%), and 752 images of "Scissors"(34.4%). All images are taken in a relatively consistent environment setup, including background color, lightning and white balance etc. These images are 300*200 RGB images in png format.

Figure 1. examples of figures in the dataset: paper(left), rock(middle), scissors(right).

2.1 Data Pre-processing

We decided to split the dataset into three parts: training set, validation set, and test set in the ratio of 60%-20%-20%. Thus, we have the following distribution.

Category	Total	Train	Validation	Test
Paper	712	428	142	142
Rock	726	436	145	145
Scissors	750	450	150	150

2.2 DataLoader

We design a DataFolder for PyTorch which assigned each category a label of 0, 1, 2. Given the GPU RAM limitation, we design the batch size for training set to be 16 and the batch size for testing set to be 8. Both DataLoader are randomly shuffled.

3 Methodology

In this section, we show in detail how we build the model to classify the hand gesture in terms of Rock, Paper, and Scissors. We first test the dataset with existing models for object

detection and classification. Then we give a heuristic approach which uses a specially designed Convolutional Neural Network with Spatial Pyramid Pooling layer to solve this problem. All models are trained with 50 epoch, with 83 steps in each epoch.

3.1 AlexNet

We start with the most famous and the most ancient net: AlexNet. However, the results were very bad, with less than 40% accuracy. It seems we encounter a overfitting issue with AlexNet. The final accuracy ends up with 34.129%.

3.2 SqueezeNet

Compared to AlexNet, SqueezeNet achieves the same correct rate as AlexNet on ImageNet dataset with only 1/50 of the parameters. Using the model compression technology, SqueezeNet can be compressed to 0.5MB. We tested the SqueezeNet with both pretrained model and not pretrained model. Both results are very close, which means pretrain has little or no effect in our task and the final accuracy are both beyond 96%.

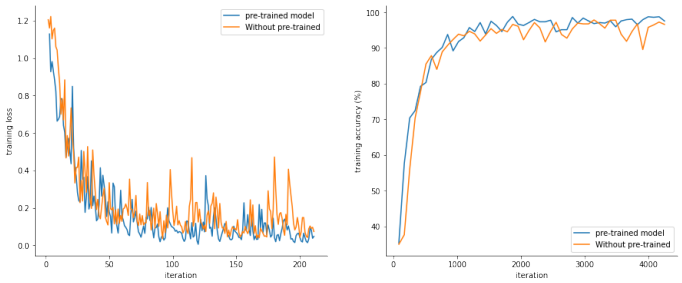

Figure 2. Training loss (left) and Test accuracy (right) on Rock-Paper-Scissors dataset with SqueezeNet. Yellow line represents without pretrained model and blue line represents pretrained model.

3.3 GoogLeNet

In deep neural networks, most of the activation values are unnecessary (value of 0), or the correlation is redundant. Therefore, the most efficient deep network architecture should be that the activation values are sparsely connected, which means that 512 output feature maps are not necessarily connected to all 512 input feature maps. There are some techniques to prune the network to get sparse weights or connections. However, the multiplication of the sparse convolution kernel is not optimized in BLAS and CuBlas, which causes the sparse connection structure to be slower than the dense structure. GoogLeNet designed a “inception” module, which uses a dense structure to approximate a sparse CNN, and uses different sizes of convolution kernels to capture receptive fields of different sizes.
In our case, we run the pre-trained GoogLeNet and get better results than SqueezeNet with an accuracy of 97% above.

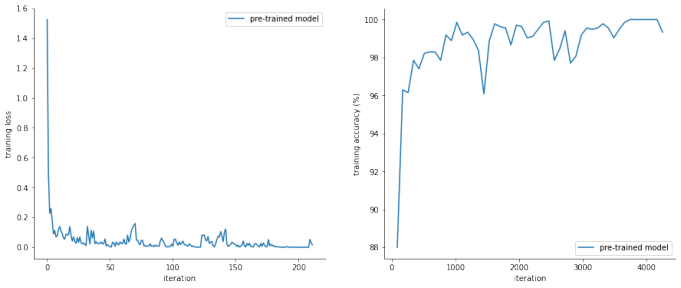

Figure 3. Training loss (left) and Validate accuracy (right) on Rock-Paper-Scissors dataset with pre trained GoogLeNet.

3.4 ResNet-50

It can be seen from the previous models that, as the network depth increases, the accuracy of the network should increase simultaneously. But we need to pay special attention to the problem of overfitting. When the network is deeper, it means that the parameter space is larger and the optimization problem becomes more difficult. Therefore, simply increasing the network depth will result in higher training errors. Residual Network, ResNet, designed a residual module so that we can train deeper networks. Because of the particularity of the residual structure, when the input is derived by loss, the derivative term will be decomposed into two, and there is a derivative term directly to the input that will not disappear, so the gradient always exists. In this way, the problem of vanishing gradients is solved. We here use the ResNet-50 model and see very good accuracy with both pre-trained and not pre-trained ResNet models.

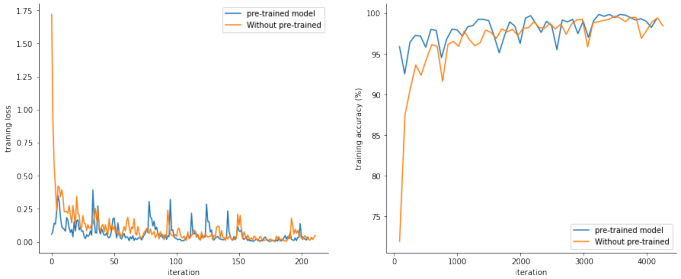

Figure 4. Training loss (left) and Test accuracy (right) on Rock-Paper-Scissors dataset with ResNet-50. Yellow line represents without pretrained model and blue line represents pretrained model.

3.5 DenseNet

Compared to DenseNet, ResNet directly adds up the features through the ”Summation” operation, which impedes the information flow in the network to a certain extent. DenseNet combines the feature map through concatenate operations, and each layer is related to other layers and has ”communication”, which maximizes the flow of information. In fact, dense connectivity in DenseNet is an upgraded version of shortcut connection, which improves the robustness of the network and speeds

up learning. However, in our cases, though the accuracy remains high (above 96%),the results of DenseNet is less than the ResNet-50.

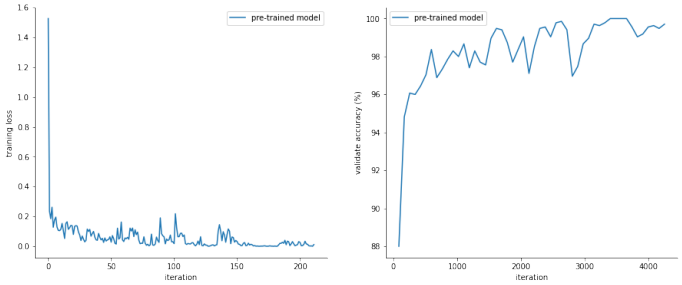

Figure 5. Training loss (left) and Validate accuracy (right) on Rock-Paper-Scissors dataset with pre trained DenseNet.

3.6 CNN with SPP Layer

Based on the results from the existing models above, we designed our own Convolutional Neural Network with Spatial Pyramid Pooling layer for this problem. Based on the dataset, we designed the following CNN model with SPP layer in the end.

SPP is a practical method that can improve the classification effect of CNN, and by adjusting the number of bins, parameters can also be reduced. SPP structure makes the original image regions corresponding to the bins of the most different levels different, and the multiple regions can correspond to objects of different scales, thus improving the robustness of the network to object scaling. This is similar to the Inception multi-scale extraction of information in GoogLeNet. Adjust the area corresponding to each activation value by adjusting the size of the filter or pooling window, thereby extracting object features of different scales.

The CNN network requires fixed input mainly in the fully connected classifier layer, and the feature extraction layer can be adjusted by controlling the sub-sampling ratio and filter size to accept inputs of various scales and sizes to obtain a fixed feature output. Given our dataset that the hand may not fill the entire image and the hand may place anywhere within the image, the fixed input constraint may influence the model efficiency and accuracy. Therefore, we decided to add the Spatial Pyramid Pooling layer to remove the requirement of CNN for fixed input.

The results are very good compared to the previous models. Especially considering that our CNN only contains 6 Convolution Layers.

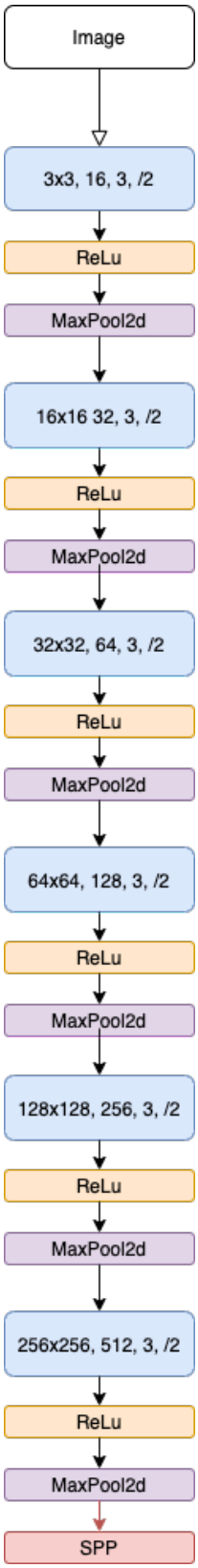

Figure 6. Structure of our CNN with SPP Layer

4 Result and Comparison

model	test accuracy	model size (MB)
CNN+SPP	**97.136%**	**8.8**
SqueezeNet [pre-trained]	96.778%	5
SqueezeNet	96.301%	5
GoogLeNet	97.613%	26.7
ResNet-50 [pre-trained]	97.941 %	102.6
ResNet-50	94.988 %	102.6
DenseNet	96.181 %	116

Table 1. Model Accuracy and Model Size on Test set

Then, we connected our model with our computer camera in order to detect the hand shape in a real-time basis. We designed an algorithm that once the model returns the prediction, the computer will give its move in response to that prediction. The figure below shows an example of how it works.

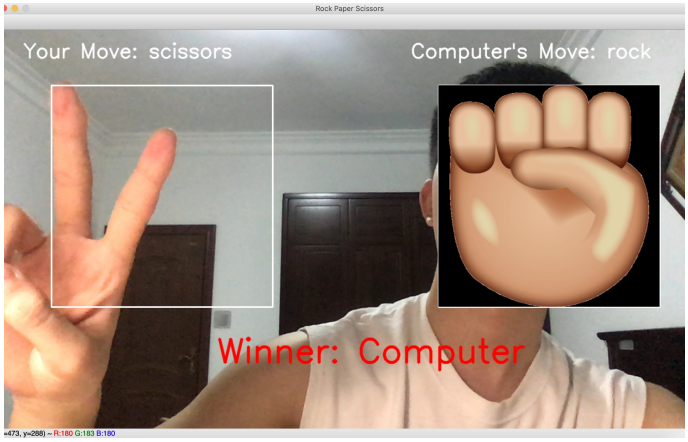

Figure 7. Example of model running with Real-time camera

5 Conclusion

Based on the results we got, it is clearly that the CNN model we proposed is considerably better than other models based on test accuracy and the model size. With a few MB higher than SqueezeNet, we get more than 0.5 percent higher accuracy. Though we see that the ResNet-50 model has the highest accuracy of 97.941%, but given that it requires 102.6 MB to store all parameters, it is less efficiency in our cases. Especially considering we are going to use that for real-time detecting purpose, which requires a higher processing efficiency.

References

[1] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. V. Esesn, A. A. S. Awwal, and V. K. Asari. The history began from alexnet: A comprehensive survey on deep learning approaches, 2018.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition, 2014. ISSN 1611-3349.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[4] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks, 2018.

[5] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size, 2016.

[6] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt. Real-time joint tracking of a hand manipulating an object from rgb-d input, 2016.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions, 2014.