# Facial Keypoints Detection with Deep Learning

Ran Gao*, Qi Liu
Department of Applied Mathematics, College of Science, Zhongyuan University of Technology, Zhengzhou, Henan, China.

* Corresponding author. Tel.: 037162506034; email: nygr@163.com
Manuscript submitted July 26, 2018; accepted September 13, 2018.
doi: 10.17706/jcp.13.12.1403-1410

**Abstract:** The facial keypoints detection is a challenging task due to the large variation of facial features, the change in 3D viewing angle, and difference in size and position of the face. Over the years, researchers have proposed a variety of algorithms such as combining multiple weak classifiers in cascade. However, a lot of work still needs to be done to further improve the detection accuracy and to accommodate for extreme cases. In this project, we proposed to use deep convolutional neural networks to locate the facial keypoints. Specifically, we experimented with LeNet, VGGNet and a 14-layer CNN on the Kaggle dataset. We also adopted image augmentation techniques to further increase the training set size. Finally, we were able to achieve a MSE of 3.02 with the VGGNet. The result indicated that deep CNNs have fairly good performance for the facial keypoints detection task.

**Key words:** Facial keypoints detection, deep convolutional neural network, LeNet, VGGNet, data augmentation.

## 1. Introduction

The facial keypoints detection is a very challenging task since the facial features vary significantly among different subjects. Even for the same individual, there still exists large variations such as the 3D viewing angle, the size of the image, or the position of the face. Over the years, researchers have proposed a variety of algorithms for feature detection, such as combining multiple weak classifiers in a cascade [1]. However, a lot of works still need to be done to further improve the detection accuracy and to accommodate for extreme cases. For this project, we aim to using deep learning techniques to detect the locations of key points on face images. The project will have wide applications such asface recognition and face tracking in videos, analyzing facial expressions and detecting facial signs for medical diagnosis.

## 2. Convolutional Neural Network (CNN)

CNN is one type of Deep Neural Network that specifically useful in image related applications such as image classification, object detection and recognition. For our project the CNN is chosen since the datasets consist image information. Several CNN architectures have been proposed to learn high level features from images. For example the Alex Net, LeNet, VGG, and GoogleNet have shown great performance in the image Net contest. More complicated CNN architectures can be found in recent publications. In this project we tested several types of CNN architectures, and evaluate their performance.

## 2.1. LeNet

LeNet is one of the early CNN architecture proposed by YannLeCun *et. al* [2]. It consists 4 convolutional layers used to extract low and high level features from 2D images. The convolutional layers are followed with fully connected layers. Traditional back propagation and gradient based method are used to train the network, allows it to learn from the provided image datasets. Fig. 1 shows an illustration of the LeNet-5 architecture.

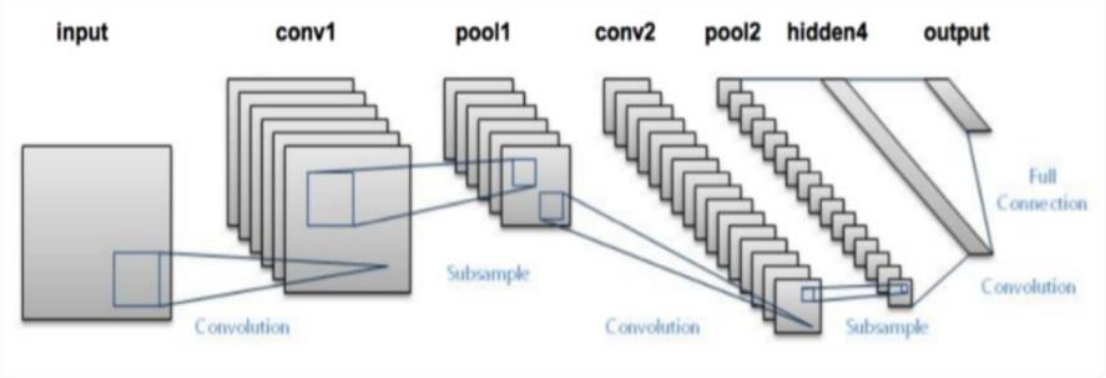



Fig. 1. An illustration of the LeNet-5 architecture.

## 2.2. VGGNet

VGG network is proposed by Karen Simonyan and Andrew Zisserman in the paper Very deep convolutional networks for large-scale image recognition [3]. It was usedto compete in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) and won the first and second place in2014. Compare to the LeNet, VGG net has more hidden convolutional layers, allows it to learn higher features from the image datasets. It also introduced other types of hidden layers into the architecture such as the zero padding layer, consecutive convolutional layers. Fig. 2 shows an illustration of the VGG-16 network architecture.

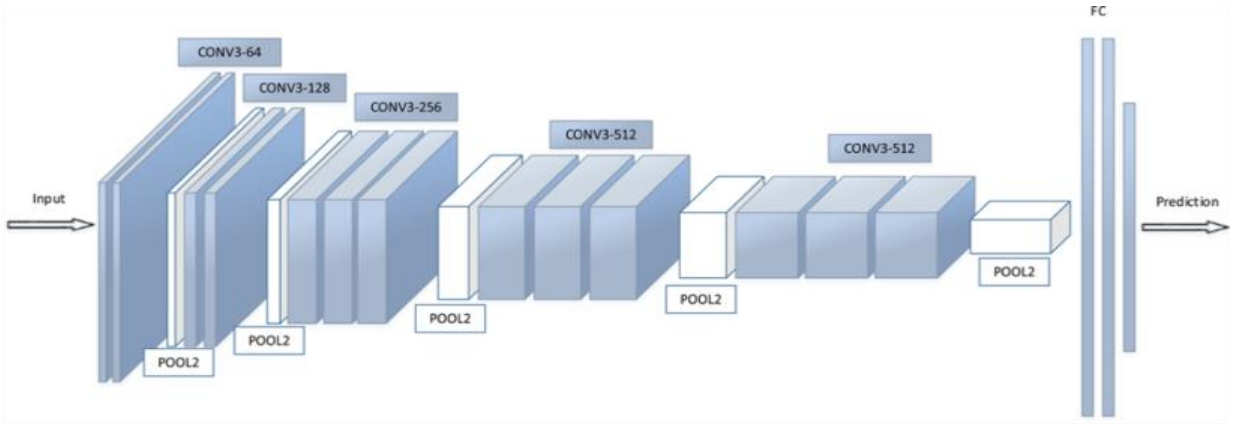



Fig. 2. An illustration of the VGG-16 architecture.

## 2.3. 14-Layer CNN

We also implemented a 14-layer CNN, with ReLU activation function, and dropout regularization after each max pooling layer and fully-connected layer. The round mean square error (RMSE) is used as the loss function. The architecture of the three-layer CNN is shown in Fig. 3.

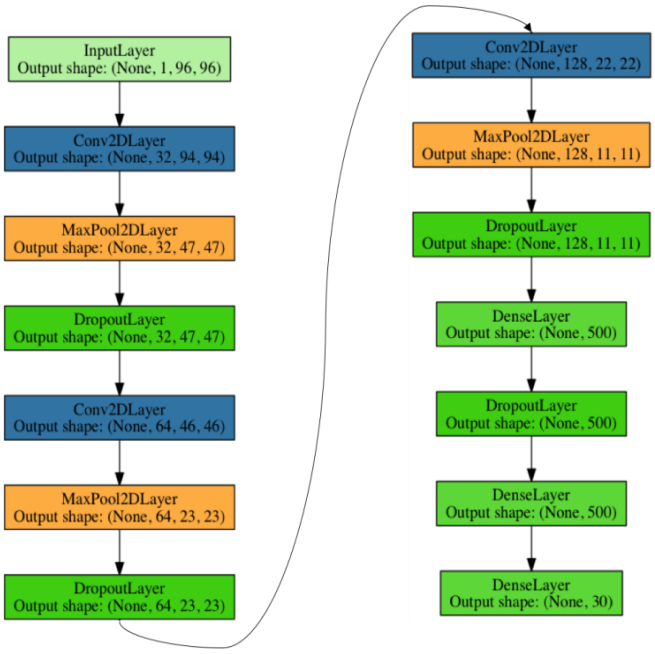



Fig. 3. Architecture of the 14-layer deep convolutional neural network.

## 3. Dataset

For the project, we will use the dataset available from anongoing Kaggle challenge: Detect the location of keypoints on face images (https://www.kaggle.com/c/facial-keypointsdetection).The dataset consists of 7049 gray scale training images, with 96*96 pixels each, and a list of 1783 test images. The training images have the (x, y) coordinate pairs of 15 features, such as eye center, nose tip, mouth center bottom lip, etc. Fig. 4 shows an example of the face image with 15 key points marked by blue star. However the dataset has the problem of being incomplete, that is, not every image has the 15 facial keypoints labeled. After eliminating these incomplete data, we have a total of 2140 training images.

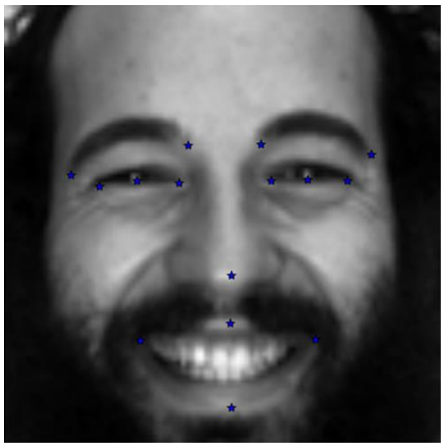


Fig. 4. An example image from the dataset along with the facial key points marked.

## 4. Training

CNNs are used to find the mapping between the raw pixel values, and the location of the facial keypoint features. The goal is to minimize the prediction error of the feature locations on the testing image set as well as the classification accuracy for each feature. The objective function is the mean-square-root value

between the predicted feature locations and the ground truth. Instead of using the softmax activation in the last layer for the image classification application, here we used Tanh active function to obtain a value between -1 and 1 for each feature location. This requires us to first transform the ground truth feature points within the scale of -1 and 1. Gradient descent based optimization methods are used to minimize the objective function. To be specific, the Adaptive Moment Estimation(Adam) method [4]. Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters. Adam computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients vt like Adadelta and RMSprop, Adam also keeps an exponentially decaying average of past gradients mtmt, similar to momentum [4]:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\,\mathrm{g}_t,$$
$$v_t = \beta_2 m_{t-1} + (1-\beta_2)\,\mathrm{g}_t^2,$$

where $m_t$ and $v_t$ are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients respectively, hence the name of the method.

## 5. Results and Discussion

### 5.1. LeNet-5

We start with the LeNet5 neural network architecture and train 100 epochs. The dataset is divided into 80 percent training and 20 percent validation data. We also set a stopping criteria that is the validation results do not get improved after 6 epochs the training will stop. The testing dataset from the Kaggle project is used to generate the prediction and submitted to the competition to acquire a error score. The first submission gives us a score of 3.35. The whole training process takes about 2 hours on the cluster with 8 CPU cores and 4G memory of each core.

### 5.2. VGGNet

Next we tested the VGGnet architecture, the VGGnet has much more layers than the LeNet5 and thus need more time to train. The training took 6 hours and came up with a final MSE score of 3.12, which is an improvement compare to the LeNet5.

### 5.3. 14-Layer CNN: Different Optimizers and Dropout

We tuned the architecture and the parameters of the 14-ayer CNN to study the performance of different optimizers and the influence of dropout regularization. The codes are written with Python Theano, Lasagne and Nolearn module.

The optimization methods we experimented with include:

a) Stochastic gradient descent (SGD)
b) Nesterovs accelerated gradient descent (Nesterov momentum): momentum = 0.9
c) Adaptive gradient algorithm (Adagrad): epsilon = 1e-6
d) RMSprop: rho=0.9, epsilon=1e-6
e) Adam: beta1=0.9, beta2=0.999, epsilon=1e-08

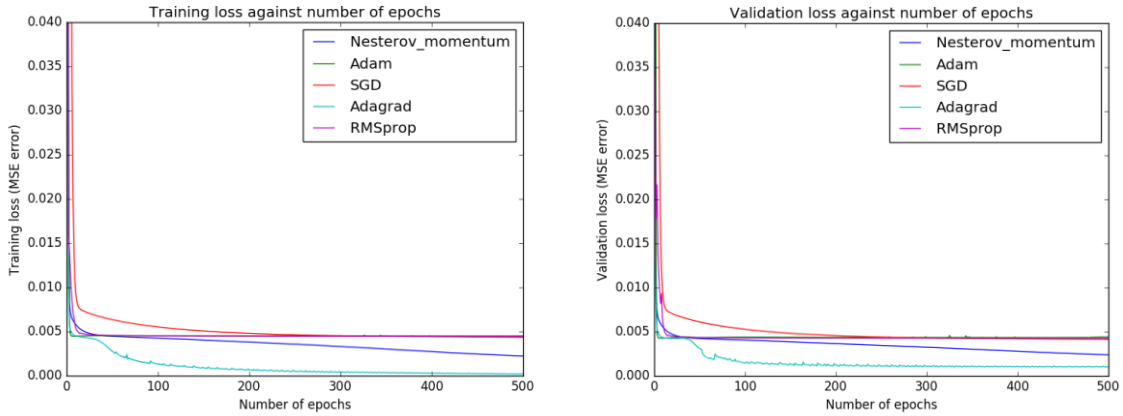



Fig. 5. Training history of the CNN using different optimization algorithms. Left: training loss against number of epochs; Right: validation loss against number of epochs.

Fig. 5 shows the training history of different optimizers. Here for each optimizer we trained the network for 500epochs on Davinci cluster, which took roughly 7 hours. The learning rate was set equal as 0.01 and the batch size was 128. We used five fold cross-validation to evaluate the training and validation loss. After parameter tuning, we found that the Adagrad optimization algorithm has the highest convergence speed. However, without regularization, the final training loss is 0.0002058, while the validation loss is 0.0010208, which indicated that the network is overfitted. Therefore, we added 4 dropout layer to control overfitting. The comparison of the network performance with and without dropout is shown in Fig. 6.

From the figure, we could observe that adding several dropout layers could narrow the gap between training loss and validation loss, and effectively control the overfitting problem.

For the optimized network architecture trained with Adagrad optimizer, the visualization of the first convolution layer filters is shown in Fig. 7.

Fig. 8 visualizes the pixel intensity of a sample image after passing through each convolutional layer.

Fig. 9 shows 16 sample face images with the predicted key points marked by blue crosses. The final submission has a MSE score of 3.07730 on the test set.

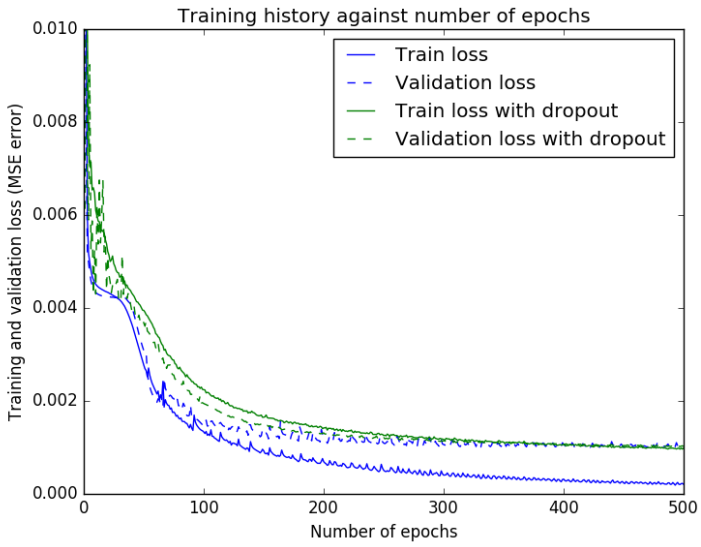



Fig. 6. Comparison of the network performance with and without dropout regularization.

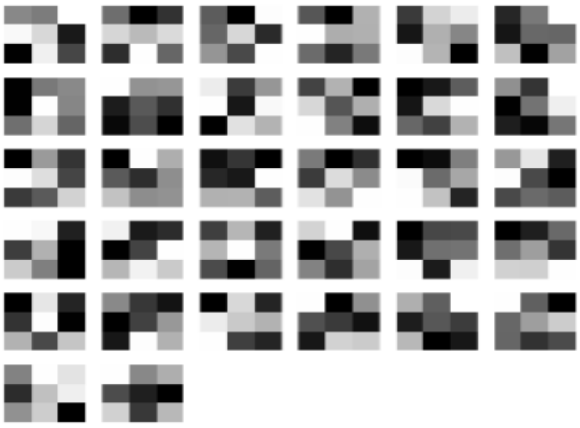
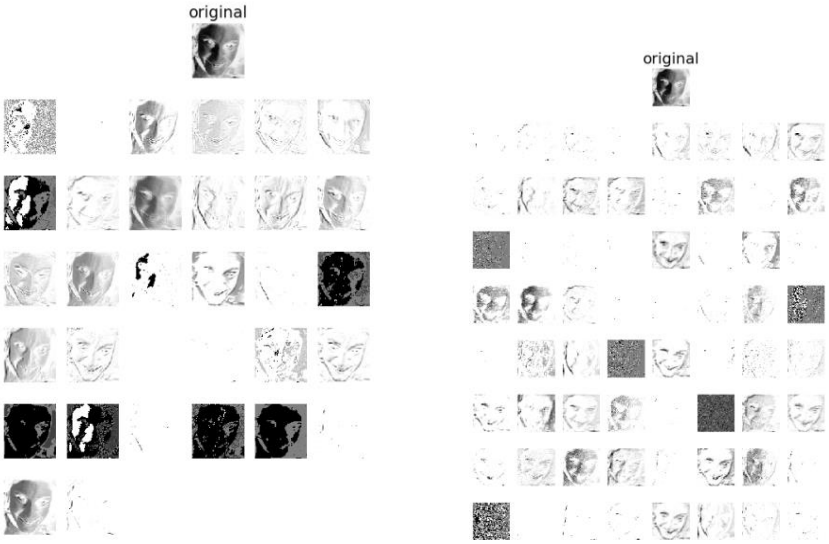


Fig. 7. Visualization of the first convolution layer filters.

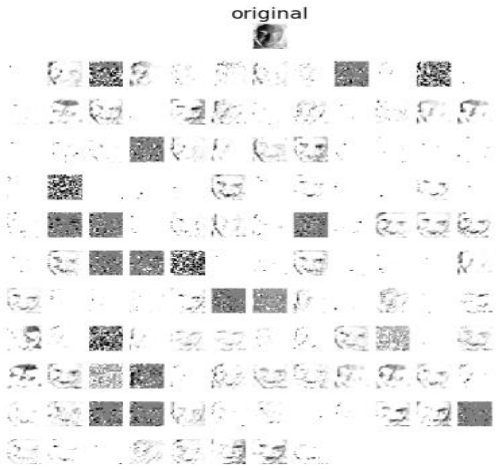
## 5.4. Image Augmentation

In order to further improve the detection performance, we expanded the training dataset using a image augmentation method. Here we flip each image and its corresponding facial key point to double the size of training images. Now we will have 4280 images, and as usual we used 80 percent of the images data set as training and 20 percent as validation. Fig. 10 shows an example of the flipped image and its corresponding facial key points.

After the image augmentation and expanding the training datasets. We retrained the LeNet5 network. Considering the increased number of training data, the training process is expected to be also longer than previous. Batch normalization [5] was proposed by Google in effort to accelerate the training process. Eventually the training took 121 epochs before stop having improvement on validation data, and the total training time is 15 hours. The final MSE score is 3.02, which proves the data augmentation is helpful in learning more precise facial features from augmented datasets. The training on VGGnet took a much longer time without the use of GPUs.




(a) First convolutional layer (b) Second convolutional layer




(c) Third convolutional layer

Fig. 8. Visualization the pixel intensity of a sample image after passing through each convolutional layer.

Due to the time limit, more types of image augmentation werent implemented such as rotation and rescaling. This could help detecting the facial keypoints more precisely. Another improvement we can make is use the GPUs to accelerate the training process. However we ran into some difficulties when trying to use the GPU cores on the Davinci clusters. These aspects can be worked on in the future research.

Fig. 9. Visualization of the sample face images with the predicted key points marked by blue crosses.

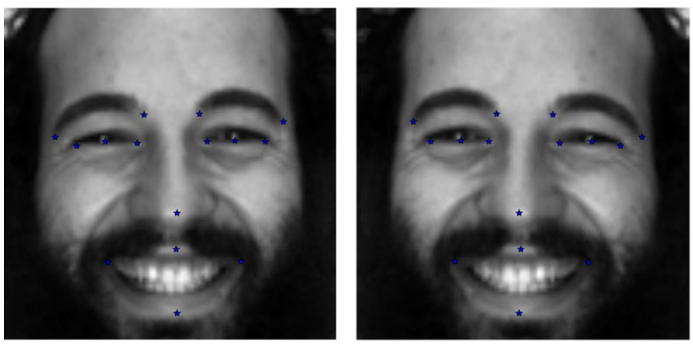


Fig. 10. An example of the flipped image and its corresponding facial keypoints.

## 6. Conclusion

In this project, we investigated the facial key points detection problem with deep convolutional neural networks. We experimented with different CNN architecture including the LeNet-5, VGGNet and a 14-layer network. We also experimented with various optimization algorithms and other techniques including dropout, data augmentation to increase the prediction accuracy on the test set. Finally, we are able to achieve a MSE score of 3.02 with the VGGNet image augmentation. For future research, we plan to go with even deeper convolutional neural network such as the ResNet [6], and explore more image augmentation technique such as rotation and crop.

## Acknowledgment

The authors thank the referee for her/his pertinent comments and valuable suggestions. This research was supported by the National Science Foundation of China (11601542), National Science Foundation of Henan Province(152300410226,152300410227) and key Research Project of Henan Higher Education Institutions(17A110036,18A110038).

## References

[1] Viola, P., & Jones, M. (2001). Robust real-time object detection. *International Journal of Computer Vision, 4.*

[2] LeCun, Y. (2015). *Lenet-5, Convolutional Neural Networks*. Retrieved from http://yann. lecun. com/exdb/lenet

[3] Simonyan, K., & Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition,* 1556.

[4] Kingma, D., & Ba, Adam, J. (2014). *A Method for Stochastic Optimization*, 6980.

[5] Ioffe, S., & Szegedy, C. (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*.

[6] He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*, 03385.

**Ran Gao** was born in Henan province in 1982. She obtained her bachelor degree in school of mathematics and statistics of Nanyang Normal University, majored in applied mathematics, afterwards, she got the master degree after three years' postgraduate education in Henan University also majored in applied mathematics. She worked as an instructor in the science academy of Zhongyuan University of Technology since 2009, the main research areas are PDE, image processing and deep learning. This research is part of her work in deep learning.

**Qi Liu** obtained the bachelor degree in Henan University in 2011. In 2016, she graduated from School of Mathematical Sciences, Zhejiang University for doctoral degree and gained the doctor of mathematicics (Ph.D). Currently, she is a lecture in science academy of Zhongyuan University of Technology.