Facial Keypoint Detection with Convolutional Neural Networks

Savina Colaco and Dong Seog Han*
School of Electronics Engineering
Kyungpook National University
Daegu, Republic of Korea
savinacolaco@knu.ac.kr, dshan@knu.ac.kr*

***Abstract*—Facial keypoint detection is a challenging problem in the field of computer vision. The keypoint detection is done by predicting the coordinates of certain facial features. In this paper, facial keypoint detection is predicted using convolutional neural networks. The models are trained to predict facial key points using the webcam input data. The facial keypoints includes eyebrow corners, nose tip, eye corners and center, and lip points. The predicted keypoints are mapped onto the webcam input data and compared with different models for better detection of keypoints. The mean square error is used to estimate the loss of each model.**

***Index Terms*—facial keypoint detection, convolutional neural network**

I. Introduction

In algorithmic perception, face recognition in unrestrained images is a spearhead. Due to various implications of face recognition, the performance gap between machines and the human visual system domain becomes a huge barrier. People can recognize faces effortlessly without giving much thought to it, this has been a challenging problem in computer vision area over many years [1]. Even though identification methods for fingerprint or iris scans are more precise, the research for face recognition has been the main focus since it is an important method for the identification of the person. Face recognition is closely related to many domains such as computer and pattern recognition, multimedia processing, security, biometrics, neuroscience, and psychology. Face recognition can be considered as a problem of identifying an individual person from face images. It is an important field of research with the interaction between psychologists and computer scientists. Face recognition in videos requires face tracking possibly with three dimensions for head pose estimation. This helps us to analyze a person's attention and estimate gaze which is useful in application such a driver monitoring system. This task can be completed by detecting keypoint positions on a person's face. Facial landmarks are the detection of essential points on the face. These features are useful in face analysis such as face verification, face tracking in images and videos, face recognition, facial signs for medical diagnosis and facial attribute inference [2]. In the field of computer vision, facial keypoint detection is a difficult and important problem that involves detecting positions like nose tip, mouth corners, eye corners, etc. The prediction of real-values co-ordinates in an image pixels space of keypoint position in a face image is difficult. The facial features vary significantly from one person to another and for the same person, the variation will be distinct due to size, position, pose, etc. It is more difficult under different conditions such as illumination, viewing angles, occlusion, etc. Recently in the field of computer vision, deep learning has been a significant breakthrough. The convolutional neural networks (CNNs) with deep structure, processes the raw pixels of image input into multiple levels of feature representations which have a semantic abstracting property required for a variety of applications. Since CNN with deep structure is a powerful tool for feature extraction and gives better detection methods [3]. Facial keypoint detection along with CNN has made significant advancements over the past few years. These networks have been used for applications such as image recognition, natural language processing, recommender systems, etc. In the paper, the CNN models are used for the detection of facial keypoint in real-time. A comparison with different models is done with facial keypoint detection.

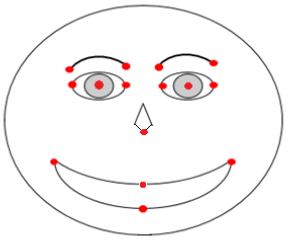

Fig. 1: 15 keypoints detected on face

II. Experiment

A. Dataset

The dataset chosen to train the models is Kaggle competition-facial keypoints detection [4]. There are 15 facial keypoint positions as shown in figure 1 per face image like left eyebrow inner corner, left eye outer corner, right eye inner corner, right eye outer corner, left eye inner corner, left eye center, left eye outer corner, right eye inner corner, right eye

center, right eye outer corner, nose tip, mouth left corner, mouth right corner, mouth center top and bottom lip. The left and right are from the subject point of view. The greyscale input images are with pixel values ranging from [0, 255] and each input size is 96×96 pixels. The training dataset consists of 7,049 images, with 30 targets, i.e. *x,y* coordinates for 15 facial keypoint positions. It also consists of missing target values for many keypoint positions for face images. The test dataset has 1,783 images with no target information. Each row has imageId and image data as a row-ordered list of pixels.

B. Models

The facial keypoints detection is done using different models such as the CNN, convolution network by LeCun (LeNet-5) [5], LeNet with dropout and convolution network by visual geometry group of University of Oxford (VGGnet) [6]. An Adam optimizer is used for all models with learning rate ranging from 0.01 to 0.0001 across 200 epochs. The batch size of 100 and validation split of 0.2 is taken. Table I shows the simple CNN information. The basic convolution neural network consists of convolutional layers followed by a max-pooling layer with stride 1. The first convolution layer has filter size 5×5 and other layers have a 3×3 filter size. The pooling size with 2×2 is taken for each subsampling. A dropout is added after every max-pooling layer. The dense layer is implemented after the layer is flattened.

TABLE I: Simple CNN

Layer	Name	Size
0	input	$1 \times 96 \times 96$
1	conv2d1	$32 \times 92 \times 92$
2	maxpool2d1	$32 \times 46 \times 46$
3	conv2d2	$64 \times 44 \times 44$
4	maxpool2d2	$64 \times 22 \times 22$
5	dropout1	$64 \times 22 \times 22$
6	conv2d3	$128 \times 20 \times 20$
7	maxpool2d3	$128 \times 10 \times 10$
8	dropout2	$128 \times 10 \times 10$
9	conv2d4	$30 \times 8 \times 8$
10	maxpool2d4	$30 \times 4 \times 4$
11	dropout3	$30 \times 4 \times 4$
12	dense1	64
13	dense2	128
14	dense3	256
15	dense4	64
16	output	30

LeNet-5 neural networks with five layers are implemented shown in Table II. Figure 2 shows the LeNet-5 architecture. This network consists of alternating convolutional layers and max-pooling layers followed by fully connected hidden layers. The filter size used is 3×3 and pooling size 2×2 with stride 1. The hidden layers have 500 neurons.

The same leNet network with similar parameters was implemented with dropout shown in Table III.

Different VGG style convolutional neural networks, which consist of convolutional layers followed by single max pool layers with fully connected layers and dropout is also implemented. The VGG network information is shown in Table IV.

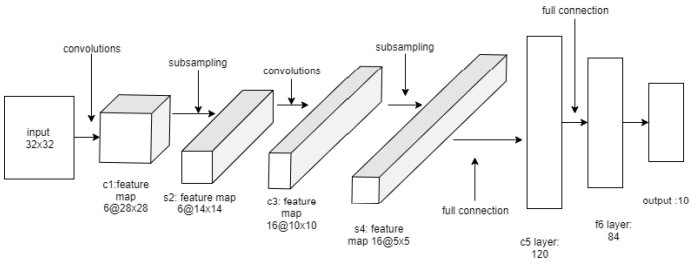

Fig. 2: LeNet-5 architecture

TABLE II: LeNet-5 layer

Layer	Name	Size
0	input	$1 \times 96 \times 96$
1	conv2d1	$16 \times 94 \times 94$
2	maxpool2d2	$16 \times 47 \times 47$
3	conv2d3	$32 \times 45 \times 45$
4	maxpool2d4	$32 \times 22 \times 22$
5	conv2d5	$64 \times 20 \times 20$
6	maxpool2d6	$64 \times 10 \times 10$
7	conv2d7	$128 \times 8 \times 8$
8	maxpool2d8	$128 \times 4 \times 4$
9	conv2d9	$256 \times 2 \times 2$
10	maxpool2d10	$256 \times 1 \times 1$
11	dense1	500
12	dense2	500
13	dense3	500
14	dense4	500
15	output	30

TABLE III: LeNet with dropout

Layer	Name	Size
0	input	$1 \times 96 \times 96$
1	conv2d1	$16 \times 94 \times 94$
2	maxpool2d2	$16 \times 47 \times 47$
3	dropout3	$16 \times 47 \times 47$
4	conv2d4	$32 \times 45 \times 45$
5	maxpool2d5	$32 \times 22 \times 22$
6	dropout6	$32 \times 22 \times 22$
7	conv2d7	$64 \times 20 \times 20$
8	maxpool2d8	$64 \times 10 \times 10$
9	dropout9	$64 \times 10 \times 10$
10	conv2d10	$128 \times 8 \times 8$
11	maxpool2d11	$128 \times 4 \times 4$
12	dropout12	$128 \times 4 \times 4$
13	conv213	$256 \times 2 \times 2$
14	maxpool2d14	$256 \times 1 \times 1$
15	dropout15	$256 \times 1 \times 1$
16	dense16	1000
17	dense17	1000
18	dense18	1000
19	dense19	1000
20	output	30

C. Facial Keypoint Detection

The input from the webcam is read and faces are detected using the Cascade Classifier object. The face cascade method locates the faces in the frame. We loop over each face to predict the facial keypoints. The detected face is normalized before passing it to the model. The model is trained on normalized images. The images are resized to 96×96 images to match the model input size. The model predicts the facial

TABLE IV: Simple VGGnet

Layer	Name	Size
0	input	$1 \times 96 \times 96$
1	conv2d1	$64 \times 94 \times 94$
2	conv2d2	$64 \times 92 \times 92$
3	maxpool2d3	$64 \times 46 \times 46$
4	conv2d4	$128 \times 44 \times 44$
5	conv2d5	$128 \times 42 \times 42$
6	maxpool2d6	$128 \times 21 \times 21$
7	conv2d7	$256 \times 19 \times 19$
8	conv2d8	$256 \times 17 \times 17$
9	maxpool2d9	$256 \times 8 \times 8$
10	dense10	512
11	dense11	512
12	dense12	512
13	dense13	512
14	output	30

keypoints for faces detected from the webcam input data.

III. Discussion

The facial keypoints were predicted by training several models such as simple CNN, LeNet-5 layer, LeNet with dropout and simple VGGnet. The model use mean squared error (MSE) to measure the average of the squares of the errors. The average squared difference between estimated values and what is estimated. All the models are evaluated with the activation function rectified linear unit (ReLU). The metric used for the model is accuracy. Adam [7] optimizer, a combination between root mean square propagation (RMSProp) and stochastic gradient descent with momentum is used as an optimizer method. Adam optimizer uses squared gradients to scale the learning rate and the moving average of the gradient. The models are implemented and analyzed with a learning rate from 0.1 to 0.001 values. The models were trained with 200 epochs with 100 batch size. The simple CNN gives 218,300 total parameters and a trainable parameter. The loss function plot is depicted with model loss across epochs. Figure 3 for simple CNN for 0.01 and 0.001 shows mean squared loss for different learning rates. The simple CNN for learning rate better accuracy with reduced loss.

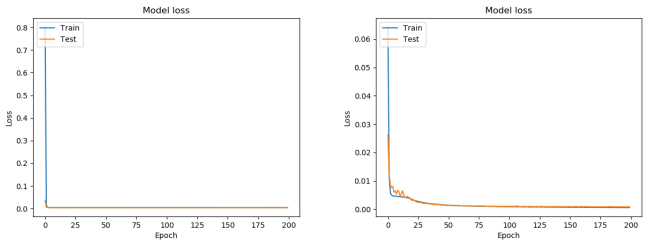

Fig. 3: Simple CNN with learning rate 0.01 and 0.001

Figure 4 shows the keypoint predicted with simple CNN. The model predicted approximate facial key points on the face when the user is looking front. The keypoint detection is not accurate when the user changes their head orientation.

The model LeNet with 5 convolution layer has 1,287,350 total and trainable parameters. The loss function for learning

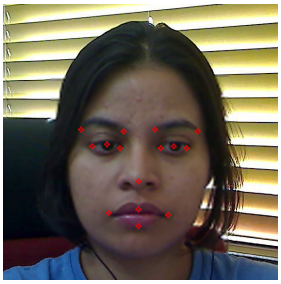

Fig. 4: Simple CNN predicted facial keypoint

rate 0.1 and 0.01 has similar values for loss across epoch plot. The learning rate of 0.001 shows reducing loss across epoch. Figure 5 depicts loss function.

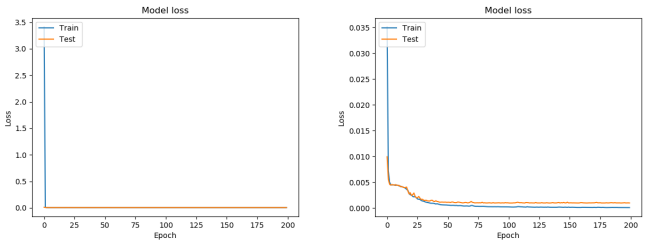

Fig. 5: LeNet-5 layer with learning rate 0.01 and 0.001

The facial keypoints detected on webcam input are approximate than simple CNN for head orientation with better accuracy. Figure 6 shows keypoint detection using LeNet-5 layer.

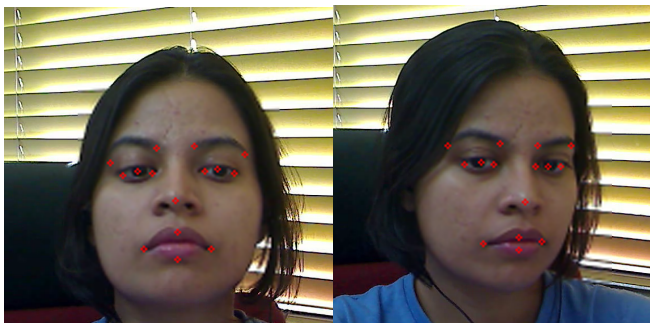

Fig. 6: LeNet-5 layer predicted keypoints

LeNet with dropout implemented with 3,682,350 total parameters. The loss for LeNet with the dropout layer shows a drop in the model loss for learning rate 0.01 for test data. Figure 7 for model loss plots for different learning rates across epochs.

LeNet with dropout does not predict keypoint as well as Lenet 5. Some of the predicted key points are missed on the face and do not work well for the head orientation which is useful if the head position is estimated. The predicted keypoints are shown in figure 8.

The simple VGGnet implemented had better accuracy with consistent loss. The total parameter and trainable parameters

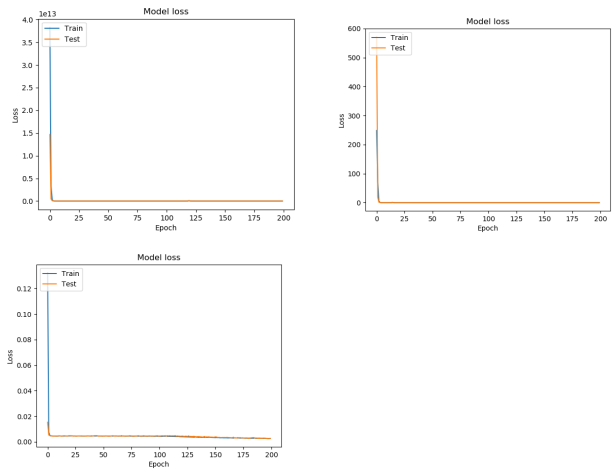

Fig. 7: Loss of LeNet with dropout with differernt learning rates

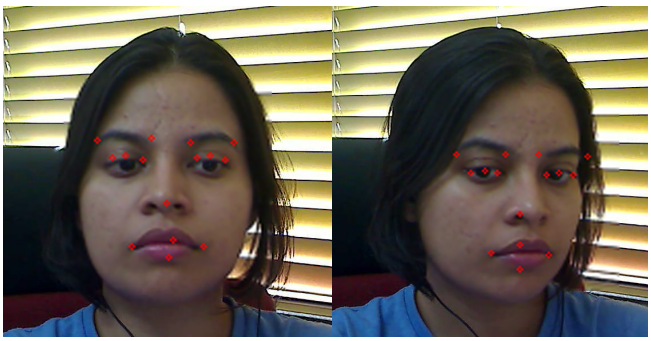

Fig. 8: Predicted keypoints with LeNet with dropout

were 9,811,422. The model loss plot for simple VGGNet for learning rate 0.001 since it showed a better result than other learning rates. Figure 9 shows loss plotted against epoch

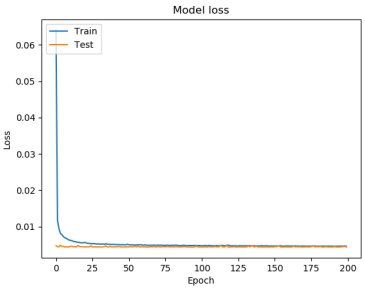

Fig. 9: Simple VGGnet loss

The models for VGG with different layers were implemented but the simple VGGnet showed better facial keypoint detection for learning rate 0.001. Figure 10 shows the detection of facial keypoints with webcam input data with simple VGGnet.

The models when trained with lower learning rates, the keypoint detection had better results and loss was reduced

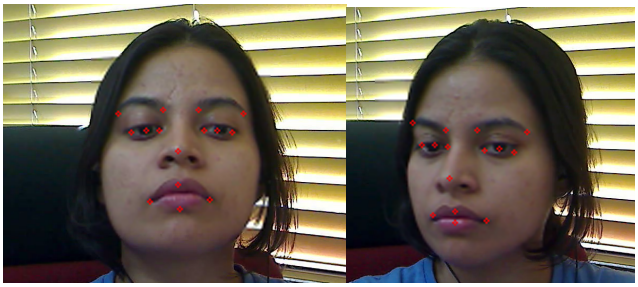

Fig. 10: Predicted keypoints with simple VGGnet

significantly.

IV. Conclusion

In this paper, different models such as simple CNN, LeNet-5, LeNet with dropout and simple VGGnet are trained using facial keypoint detection dataset by Kaggle. The model predicts the key points which are mapped onto the webcam input data. The comparison of different models is discussed by taking parameters such as learning rate, optimizer, loss function, and accuracy. For a certain model, the keypoint detection was not accurate with the corresponding learning rate. The future work is concentrated on building a better model considering different cases. Using a better model, we can analyze the head position of the user. Data augmentation can be applied to improve the performance of the models.

Acknowledgment

This work was supported by the Industrial Strategic Technology Development Program-Development of Vehicle ICT convergence advanced driving assistance system and service for safe driving of longterm driving drivers (20003519) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea)

References

[1] S. Shi, "Facial Keypoints Detection," arXiv preprint arXiv:1710.05279, 2017.
[2] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in European conference on computer vision, 2014, pp. 94-108.
[3] U. Özaydın, T. Georgiou, and M. Lew, "A Comparison of CNN and Classic Features for Image Retrieval," in 2019 International Conference on Content-Based Multimedia Indexing (CBMI), 2019, pp. 1-4.
[4] "Kaggle Competition - Facial Key Points Detection," 7 May 2013. [Online]. Available: https://www.kaggle.com/c/facial-keypoints-detection.
[5] Y. LeCun, "LeNet-5, convolutional neural networks," URL: http://yann. lecun. com/exdb/lenet, vol. 20, p. 5, 2015.
[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.