Facial Keypoints Detection

Sara Kartalovic
sara.kartalovic@studenti.unipd.it

Sercan Albut
sercan.albut@studenti.unipd.it

Abstract

Facial keypoints detection is a machine learning and computer vision problem whose solutions have many different applications. It can be used to track and capture human faces in images and videos, to recognize faces as well as facial expressions. It is a crucial component for face recognition technologies. The problem we face includes the prediction of 15 facial keypoint coordinates on a given input image of different human faces. Facial keypoints recognition is challenging in the sense of machine learning because of the fact that people have different facial features. With the rise of Deep Learning features the problem became more interesting because Convolutional Neural Networks are proven to be very effective when it comes to solving facial recognition problems. With this being said, in this project we are introducing and sampling state-of-the-art deep convolutional neural network approaches starting from AlexNet on the task of facial keypoint detection problem. Models are trained and tested on 96x96 grayscale images and in order to detect the facial keypoints, deep learning techniques have been used.

1. Introduction

The scope of this project is facial keypoints detection on images of human faces. Facial keypoints recognition task has become crucial in today's world as it is used widely on CCTV, imageNet searches, basically involving everything which takes into account the extraction of keypoints on visuals. The reason why this detection problem is challenging is because people have different face features; moreover, the image itself can make it challenging depending how, when and where it was taken. Some of the challenges that are present not only in this problem, but in many computer vision problems, are: illumination, scale, viewpoint, and position. To deal with the problem, deep convolutional neural networks models will be used in order to solve the problem and find the mappings between the image pixels and facial keypoints. Different face recognition and visual key point extraction methods and applications will be briefly explained before experimentation. The implementation will start with a basic model which is considered as a cornerstone of the process: AlexNet. Later on, it will be continued by introducing more complex and deeper models by adopting some VGGnet properties. The experiments, implementations and improvements will be completed on a publicly available data set. The process of obtaining the deep convolutional neural networks will be explained as it has been used as layerwise and basic structures of itself. At the same time there will be a process of the evaluation of the data and model. Since the task is widely used and still in process of improvement, the experiments that have been conducted in this paper will have different approaches with the aim of increasing the accuracy and decreasing the mean average error of the model and detection.

Components of Network	Number and specifications
Convolutional	10 layers Filters: 32,32,64,64, 96,96, 128,128,256,512 Kernel: (3,3) Padding: 'same' Activation: Leaky ReLU (alpha=0.1)
Dense	2 layers: 512, activation: 'relu' 30, no activation function
Batch normalization	Applied after each Convolutional layer
Max pooling layer	5 layers: (2,2)
Dropout layer	1 layer before the last Dense layer dropout rate = 0.1

Table 1. Description of the architecture of the network

Table 1 shows the structure and architecture of the model that we have obtained after conducting numerous tests. This model showed a good performance when it comes to predicting the coordinates of the facial keypoints during our experiments and was the model that provided the highest accuracy as well as the lowest mean average error. The path of obtaining this model is explained in this paper.

2. Related work

Facial keypoints recognition has been a topic of research for a while. There are numerous models and applications that

are related to this task. Previous work includes approaches using deep learning and CNN which is the structure that we are going to exploit as well. In this section, we describe two applications and different artificial neural networks that have been used to recognise faces and facial features.

2.1 FaceReader application

An app FaceReader, for example, is used for emotion analysis. Without any bias, this app records the primary reactions of the test subject and provides a validated data analysis of facial expressions. The unique feature of this application distinguishes the intensity of the active muscles by using the groups of facial muscles called 'action units'. This is performed for the left and the right side of the face separately.

The application follows the 4 main steps. At the first step it uses Viola-Jones algorithm to find a face in the picture. Next, it uses 500 facial keypoints in order to make precise 3D modeling of the face. At the third step, it uses Artificial Intelligence and Deep Learning to analyze the face in detail. Furthermore,at the last step, an artificial neural network is used for classifying the emotional facial expression.[1]

It is reported that FaceReader 6, according to their validation study, achieves the accuracy of 88%, whereas a newer version of the application FaceReader 8.1 achieves much higher accuracy of 96%. According to Google Scholar, so far FaceReader is the most cited facial expression recognition software in nearly 1,300 publications since 2005. Moreover, more than 1,000 universities (including 6 out of 8 Ivy League universities), research institutes, and companies in many markets such as psychology, consumer research, user experience, human factors, and neuromarketing use this application.

2.2 Luxand FaceSDK application

FaceSDK is another application that uses facial keypoint detection for different purposes. "FaceSDK is a high-performance, multi-platform face recognition, identification and facial feature detection solution."[**2**] It is used for face recognition, live video recognition, face detection as well as mask-on face detection, facial keypoints recognition, facial expression recognition, gender and age recognition, IP camera support and thermal face detection.

When it comes to 70 facial points recognition, the SDK processes the image in order to find a human face on it. Then, it returns the coordinates of the 70 unique facial keypoints. These points include the eyes, eye contours, eyebrows, lip contours, nose tip, and others. The detection of keypoints works in real-time on desktop or mobile, which allows this software to be used on video for tracking and transformations of facial features.

2.3 Convolutional neural networks (CNN)

One sort of artificial deep neural network that is used particularly for image recognition, classification and object detection is CNN. Because our dataset consists of images and CNN's have proven that they can be successfully used for solving the kind of problem that we are facing, we decided to exploit this structure and use CNN in order to correctly recognize facial keypoints in a given image. One of the reasons why CNN's are so successful is because they extract features directly from the image and these features are learned while the training is performed. For instance, the Alex Net, LeNet, VGG, GoogleNet, ZFNet and ResNet have been used to solve the facial keypoints detection problem. [3] Using AlexNet and VGGNet as examples, we try to create our own CNN architecture to detect keypoints in the facial images.

"Network framed by Alex Krizhevsky, et al. AlexNet has been trained and tested for classification of high-resolution images, with results better than conventional feature extraction methods like SIFT." [3] The part of AlexNet has been used, for example, to improve vehicle image detection and classification [4]. Also, more recently, AlexNet features are applied in order to get better classification of scenes. [5] This network is composed of 5 convolutional layers, 5 relu layers, and 5 max pooling layers which are followed by 2 fully connected dense layers. [3] The first convolution layer accepts the images that are of size 224*224*3 with 96 kernels of size 11*11*3 and stride 4. The output of the layer is 55*5*96 while activation function is rectified linear unit (ReLU) . This activation along with max pooling is used in order to reduce overfitting and add nonlinearity where now the output of the layer is 27*27*96. The second convolution layer has 128 kernels of size 5*5, stride 1*1, and provides the output 27*27*128. The following 3 convolution layers are composed of 384, 384 and 256 filters each of size 3*3 each and stride 1*1 each where after each convolution layer there are max pooling, dropout and padding layers applied. At the end, we have two fully connected layers which give 4096 features. "Layer1 gives edge and blob of the input image, layer2 performs the conjunctions of these edges or responds to corners and other edge or color conjunctions, layer3 output is texture of a image, Layer5 identifies object parts", where the last two fully connected layers give the image features and are used as feature descriptors where region around the keypoint of image is detected using Scale-invariant feature transformer or shortly SIFT.[3] The scale-invariant feature transform is a feature detection algorithm to detect and describe local features in images.

Karen Simonyan and Andrew Zisserman proposed VGGNet in their paper *Very deep convolutional networks for large-scale image recognition* [6]. Their model won first and second place in 2014 on the competition *ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)*. Since it has been introduced to the public, it is still accepted as a state-of-art model for specifically image classification. VGG, which is short for Visual Geometry Group, or another known name as oxford model, has a key value over the definition and repeated usage of VGG-blocks to clarify the image filtering. The concept is basically starting with as small as 3x3 pixel filtering followed by max pooling layer. In their research

paper of Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014, the process has been basically defined as:"The image is passed through a stack of convolutional (conv.) layers, where we use filters with a very small receptive field: 3 x 3 (which is the smallest size to capture the notion of left/right, up/down, center). […] Max-pooling is performed over a 2 x 2 pixel window, with stride 2.".The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. The model has made improvements over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. When it is mentioned VGG, it rather means VGG Net-16 which consists of 16 convolutional layers and it is considered appealing because of its very uniform architecture. In truth VGGNet consists of 138 million parameters, which at first could be hard to handle. To give an idea of its basic structure it follows the routine of two convolutional layers ending with max pooling and the last 3 layers are fully connected. The first convolutional layer accepts the images that are of size 224*224*3 then continues 13 of them with max pooling in between. It finishes the process with softmax.

3. Dataset

The dataset used for the project is available on Kaggle and comes from the Facial Keypoints Detection challenge where the goal is to detect the location of keypoints on face images. [7] The dataset is split into train and test sets which include lists of 7049 and 1783 images respectively. The images used only one channel and were grayscale where each image was 96*96 pixels as shown in Figure 1. They were composed of a list of pixels which are ordered by row, as integers in range (0,255). For the training dataset, the coordinates for 15 keypoints are also provided along with images. As shown on Table 2, those 15 keypoints represent different elements of the human face such as: center and corners of both eyes, inner and outer ends of the eyebrows, tip of the nose, corners of the mouth and center of the top and bottom lips. On the Table 1 and in the dataset, the left and right are the sides from the view of the subject. Because these 15 coordinate features are in the form of (x, y), there are 30 points features for each image of the face.

The facial keypoints	
Left eye center	Right eye center
Left eye inner corner	Right eye inner corner
Left eye outer corner	Right eye outer corner
Left eyebrow inner end	Right eyebrow inner end
Left eyebrow outerend	Right eyebrow outer end
Left mouth corner	Right mouth corner
Top lip mouth center	Bottom lip mouth center
Nose tip	

Table 2. The 15 facial features

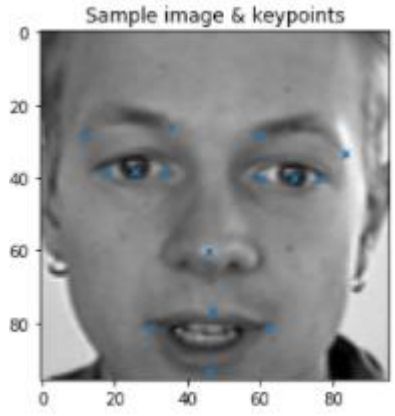

Figure 1. Sample of training image of size 96x96 and corresponding 15 facial keypoints

Immediately after loading the datasets, the images were scaled. The training dataset had numerous missing facial keypoints where 4909 rows have at least one missing keypoint. After removing the data with missing keypoints coordinates we were left with less than half of the training dataset which now consisted of 2140 images that include all 15 keypoints.

In computer vision and machine learning tasks it is almost always better to have more data to experiment with, but we were left with only 2140 images and corresponding key points to experiment with as a training dataset. We modified our original dataset in order to make it more flexible to variations and obtain more training samples. Firstly, we flipped pictures horizontally from left to right which doubled our training dataset. This way the x coordinates were flipped, while y coordinates remain the same. Then, we rotated pictures along with the keypoints for 12 degrees to the left and continued with changing the brightness of the pictures. The brightness was increased by factor 1.5 and decreased by factor 0.8. The images and corresponding key points were shifted to bottom, up, right and left and this way 6350 new images were obtained. Lastly we added the random normal noise to the

original training images which provided us additional 2140 samples to work with. After all alterations we ended up with a much larger data set for training which included 48970 images and corresponding keypoint features.

4. Methods

After thoroughly analyzing the dataset and creating more data to train our model on, we proceeded by exploring the work that was done in order to solve this problem. Because it has been proven that deep convolutional neural networks work well when applied to computer vision problems, we decided to take this approach. Next, because we had 48970 samples in our training dataset, as a starting point AlexNet was being explored and few adoptions were taken from VGGnet. Also, it has been considered to use deeper VGGnet as our starting point; however, it was impossible to know if we would obtain better results and prediction by using exactly 13 convolutional layers. By trying to find the middle between complexity and accuracy while keeping prediction loss low, we started with a less complex model. Eventually, to obtain better results, we explored and created a deeper and more complex neural network.

At the beginning, for the sake of comparison, all models were trained using 75 epochs. Moreover, 80% of the data was used for training, while the remaining 20% were used for the validation purposes. When improved models were found, the number of epochs was doubled so the models could benefit from an increased number of epochs.

Towards the end, we ran into an issue which was the high mean average error. Part of our goal was to minimize it, but we were not able to find a way to decrease it, so we had to change the approach and do more research on the models that were made for solving the same or similar tasks. We quickly concluded that almost all other more successful models used a different activation function. All models that performed very well used the 'Leaky ReLU' activation function. After obtaining this information, and applying it to our model, we were able to obtain better results. We found out that this activation speeds up the training process and is good for models that suffer from sparse gradients or exploding gradients. This 'Leaky ReLU' activation function is used as an alternative and to face the problem called 'dying ReLU' where the neuron is considered to be 'dead'. This neuron is 'dead' if it always gives the output 0 and is stuck on the negative side.

5.Experiments

The main goal of the experiment is minimization of the loss of the facial keypoint coordinates and high accuracy. In our experiments we create models by using Keras library and start with a neural network model that consists of 5 convolutional layers followed by 3 dense layers which is modeled after AlexNet. Testing different batch sizes did not influence the model very much, so we decided to use a smaller batch of size 64. The optimizer used for all models is 'adam' which is "an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments"[8]The loss function for regression used is 'mean squared error loss' because we are comparing the predicted coordinates of the keypoints with the actual coordinates. The activation function used is 'ReLU' and during the experiment the dropout layer is added before the fully connected layers with dropout rate 0.2 in order to reduce overfitting. Activation is not applied on the final dense layer because we need directly the coordinates of our facial features keypoints. In the experiments we also used a technique called callbacks.ReduceLROnPlateau; it simply inherits from callbacks and reduces the learning rate when a metric has stopped learning. Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced. Also, another technique called early stopping is used. Early stopping is a method that allows you to train a large number of training epochs and stop training once the model performance stops improving on a hold out validation dataset. It is used to determine the choice of the number of training epochs to be used. After testing the models with different kernel sizes, the model that had small improvement used kernels 3x3 on each convolution layer which is also a characteristic that we adopted from VGGnet architecture. In order to stabilize the learning process, batch normalization is added to standardize the input to each layer for each mini-batch. Adding batch normalization after every convolution layer improved our model accuracy, but the results were still not satisfying so we continued our experimentation.

The next step was to create a deeper network, so we started by adding more convolutional layers. To check if this is actually going to work, we firstly added only 2 convolutional layers with 384 and 256 filters respectively and kernels of size 3x3. The result did not improve and there was overfitting, so the dropout layer was added before the last dense layer again to reduce overfitting. Without using dropout layer models show overfitting, smaller accuracy and bigger loss no matter how many convolutional layers they have, so the dropout layer with dropout rate 0.2 is adopted and is not removed from the future model's experiments.

Before creating a deeper network, maximum pooling layers are added after each convolutional layer. After this, our accuracy dropped and loss increased so we tested the model with the maximum pooling layer after every other convolutional layer. This gave worse results, so for now we continue with using only 2 max pooling layers after the first and second convolutional layers. We continue with creation of a deeper network and progressively add more convolutional layers while monitoring the loss, mean average error, and accuracy until we reach a similar architecture as VGGnet that is explained in section 2.3.

The accuracy was at its peak point while using 10 convolutional layers and dropped significantly after

switching to 11 or 12 convolutional layers. It dropped more than 6% when using 12 convolutional layers, and mean average error increased after using more than 10 convolutional layers. We tried adding max pooling layers, but decided to stop adding more convolutional layers because the models were more complex but were not improving for solving a given task with our dataset. With this being said, we did not reach VGGnet architecture and we explored thoroughly the model that used 10 convolutional layers.

After a lot of experiments, the architecture of the neural network that is exploited consists of 10 convolutional layers followed by 3 dense layers. In order to get the best results, we experiment with the number of max pooling layers and dropout layers, as well as filter and kernel size. Additionally, we change the number of epochs from 75 (the number of epochs that all our previous models used) to 150.

First, we started by adding max pooling layers, but there was not much improvement, so we decided to change the size of the filters. Because we started with the architecture of AlexNet which accepted pictures of size 224x224, and our pictures were only 96x96, we decided to reduce the number of filters. We significantly reduced the number of filters at our dense layers from 4096 to 512. This helped to reduce the mean squared error and loss, but our accuracy dropped a little as well. The mean squared error was decreased by removing one dense layer which means that now we are left with two dense layers,one of size 512 and the other of size 30. Finally, the activation on each convolution layer was changed from 'ReLU' to 'Leaky ReLu', while the activation on the first dense layer remained the same.

5.1 Results

Table 3 provides results of some of the main and major experiments that were conducted after we decided that we will use the neural network that includes 10 convolutional layers. It provides information such as the validation mean squared error, validation loss, and validation accuracy. It shows that the best model we conducted, next to using 10 convolutional layers, also uses 5 maximum pooling layers, 2 dense layers and 1 dropout layer. The model that obtained slightly better accuracy of 87.42% uses a dropout rate of 0.1 while the model that has the lowest mean average error of 1.9691 uses the dropout rate 0.2. However, the model with dropout rate 0.1 has much higher loss, so we decide to proceed with a model using dropout rate 0.2 and test it.

Figure 2 shows 3 different plots of comparison of training and validation sets of our best model mentioned above. First plot compares mean average error, second plot compares loss and third plot compares accuracy. As it can be seen from the graphs that mean average error and loss decreases as the accuracy increases. The distance between curves states that adjusted learning rate is fit for the model. Furthermore, the very small difference between loss curves indicates that there is no overfitting present. The training is relatively stable due to the batch normalization after each convolution layer.

Figure 3 shows the final model used for prediction on the test set. Even though the model had relatively stable accuracy on the validation set, it can be seen that the test accuracy curve is very unstable which may mean that we need a deeper model.

6. Conclusion

The focus of this project was prediction of the coordinates of 15 facial keypoints on raw facial images. Particularly, we would map the coordinates of keypoints to the raw pixels on 96x96 grayscale pictures. So, the purpose is to find out how well this deep convolutional neural network will learn facial keypoints and detect them on unseen data. The AlexNet architecture was used as a baseline, but since there was a need for a deeper model, we experimented with different deeper CNN structures. We also experimented with different hyperparameters such as dropout rate, filter size, kernel size, maximum pooling, and activation functions. Different data augmentation techniques were performed such as scaling, rotating, flipping, shifting, changing the brightness of the picture and adding noise.

Starting from a neural network that has 5 convolutional layers and 3 dense layers, we provided a deeper neural network which has 10 convolutional layers, 5 maximum pooling layers, 2 dense layers and 1 dropout layer. After experimentation and through many processes, the results proved that deeper convolutional neural networks dealt better with detection and prediction of facial keypoints which was our main objective. In future, deeper and different architectures of convolutional neural networks should be explored and tested in order to improve the overall performance and solve the challenge accurately.

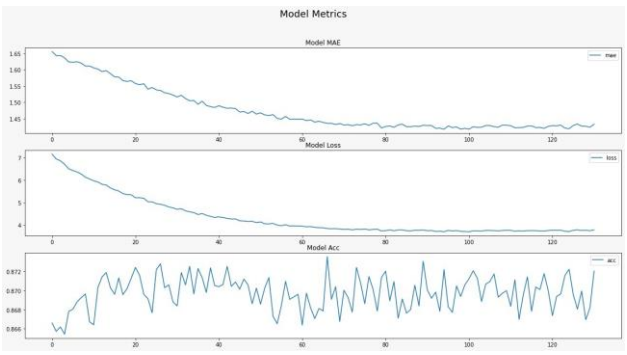

Figure 3. Test mean average error, test loss, and test accuracy curves plotted against the number of epochs

Model Description (10 convolutional layers)	Val_mae	Val_loss	Val_acc	Avg_time/epoch
2 max pool 2x2, 3 dense layers, dropout layer (0.2), 'relu'	5.2647	103.2744	0.8353	4 sec
2 max pool 1x1, 2 dense layers, 'relu'	5.1434	86.0458	0.7952	4 sec
3 max pool 1x1, 3 dense layers, dropout layer (0.2), 'relu'	4.9871	81.9883	0.7765	11 sec
4 max pool 1x1, 3 dense layers, dropout layer (0.2), 'relu'	4.8663	82.8976	0.806	11 sec
4 max pool 2x2, 3 dense lay, dropout layer (0.2), 'relu', reduced size of filters	4.3257	64.2515	0.7633	3 sec
4 max pool 2x2, 2 dense lay, dropout layer (0.2), 'relu', reduced size of filters	4.3647	47.6951	0.7965	5 sec
5 max pool 2x2, 2 dense lay, dropout layer (0.2), 'relu'	3.0534	25.3422	0.8111	14 sec
4 max pool 2x2, 2 dense lay, dropout layer (0.2), 'leaky relu'	3.182	20.459	0.862	14 sec
4 max pool 2x2, 2 dense lay, dropout layer (0.1), 'leaky relu'	2.393	17.1444	0.8708	14 sec
5 max pool 2x2, 2 dense lay, dropout layer (0.1), 'leaky relu'	2.3327	16.2562	0.8742	14 sec
5 max pool 2x2, 2 dense lay, dropout layer (0.2), 'leaky relu'	1.9691	11.6214	0.8718	14 sec

Table 3. Validation mean average error, validation loss, validation accuracy, and average time per epoch obtained after conducting experiments on convolutional neural network containing 10 convolutional layers

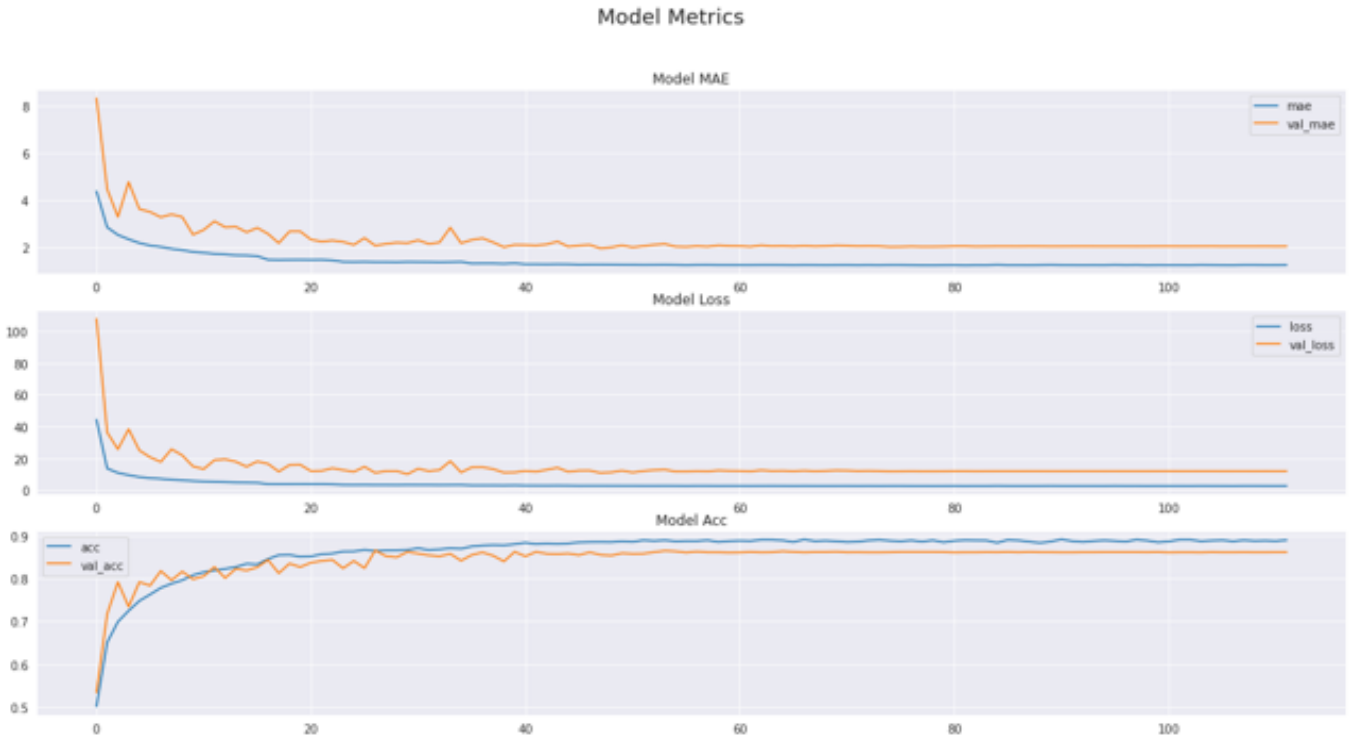

Figure 2. Comparison plots of different curves for training and validation sets

References

[1] *Facial expression recognition software: FaceReader*. Facial expression recognition software FaceReader.(n.d.).https://www.noldus.com/facereader.

[2] *Detect and Recognize Faces and Facial Features with Luxand FaceSDK*. Luxand. (n.d.). https://www.luxand.com/facesdk/.

[3] K.Kavitha, B. Thirumala Rao, & B. Sandhya. (2016). *Evaluation of Distance Measures for Feature based Image Registration using AlexNet*., Vol. 9, No. 10

[4] Yiren Zhou, Hossein Nejati, Thanh-ToanDo, Nagai-Man Cheung, Lynette Cheah, *Image-based Vehicle analysis using Deep Neural network: A systematic Study*, IEEE International conf. on digital signal processing, 2016.

[5] Lisha Xiao, Qin Yan, Shuyu Deng, *Scene Classification with improved AlexNet Model*, 12th international conf. on intelligent systems and knowledge engineering, 2017.

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

[7] A. Krizhevsky, I. Sutskever, G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 2012, pp. 1097–1105.

[8] Gao, R., & Liu, Q. (2018). Facial Keypoints Detection with Deep Learning. Journal of Computers, 13, 1403–1410. https://doi.org/http://www.jcomputers.us/vol13/jcp1312-07.pdf