A Direct Regression based CNN approach to Face Alignment

CandNo: 215865

Introduction

In this report, we present a reasonable approach for facial landmark detection, face alignment. We then outline our approach for image segmentation and present our graphical effect to demonstrate the potential uses for the face alignment model. Similar to previous regression-based approaches, the model directly predicts the landmark co-ordinates after training on a large training data set (2811 images) with labelled key points (46 facial landmarks).

Methods

Pre-Processing

Before data augmentation and pre-processing, a train/validation split with the ratio 70:30 to allow evaluation of the model. Figure 1 shows the process of pre-processing the data.

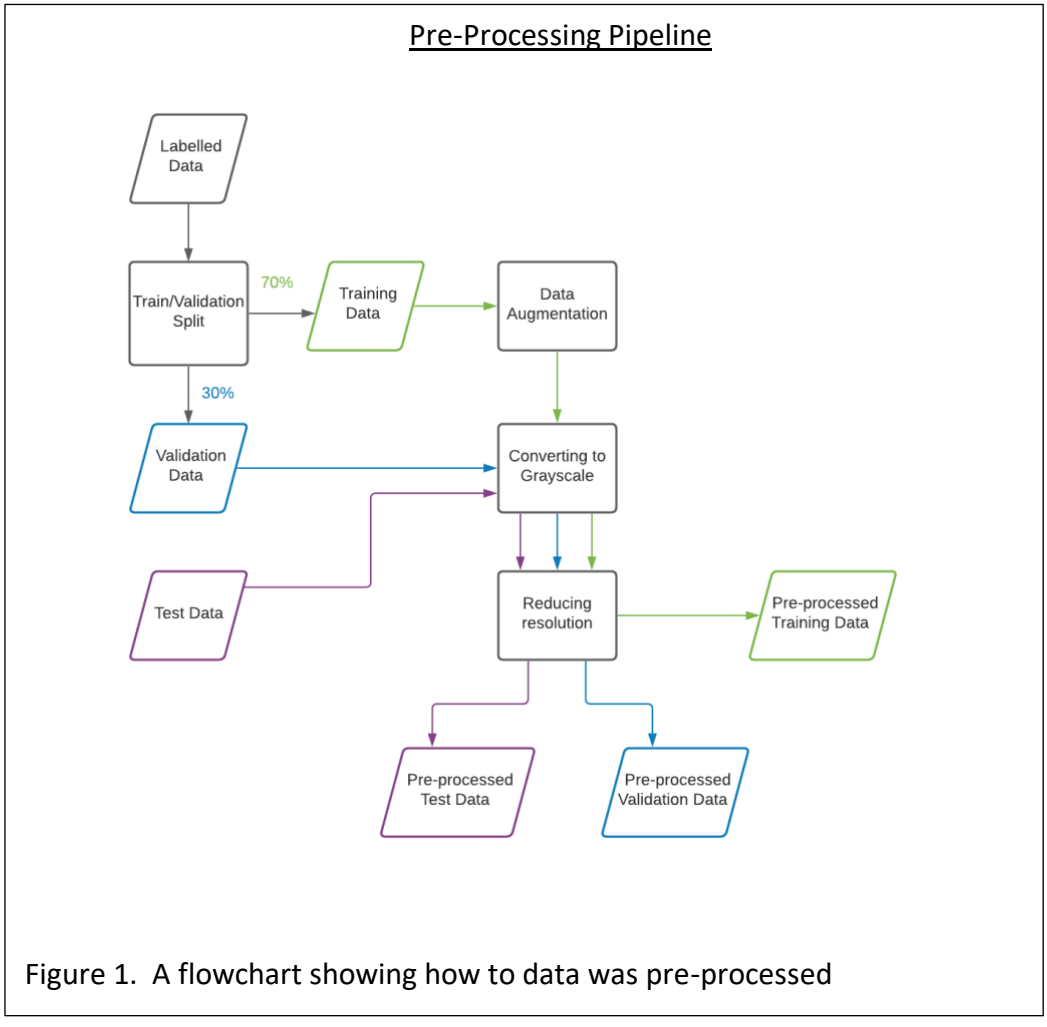


Figure 1. A flowchart showing how to data was pre-processed

Data Augmentation

We augmented the training data using an offline augmentation approach. To do this, each image and corresponding labelled points were flipped horizontally to form the augmented dataset along with the original images, as seen in Figure 3. Images were flipped using cv.flip. To amend the key points to line up with the flipped image, we used a simple transformation width-x-1 for the x values of the pixel. Data Augmentation improved the robustness of the model and doubled the number of training images.

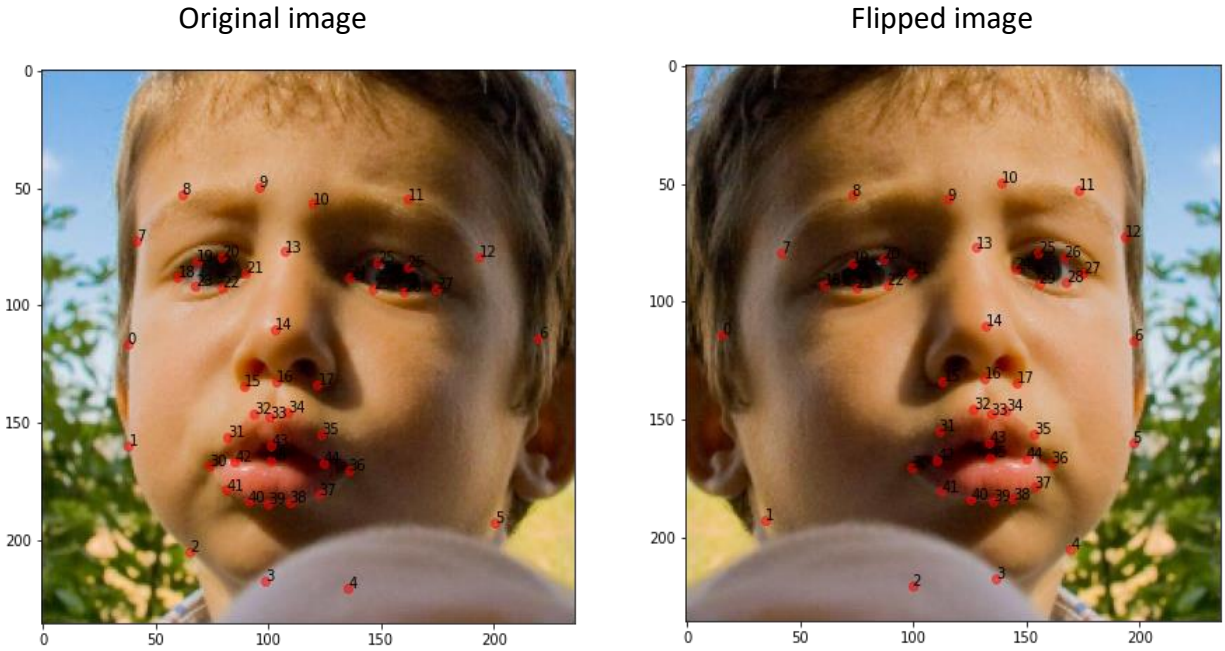


Figure 2. The effects of horizontal flipping and the location of transformed landmarks

The pre-processing techniques used on the data were converting to grayscale and reducing the resolution. Using grayscale images reduced the dimensions of the data, and therefore increase the speed of the model. We reduced the resolution of the images to 96x96 to further reduce the dimensions and speed up the processing of images. Following, the annotated landmarks were reshaped to align with the transformed images.

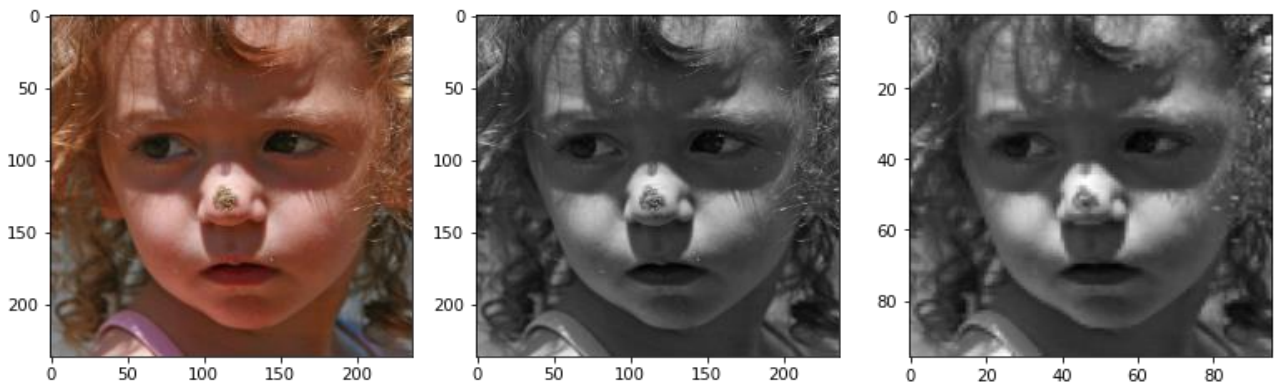


Figure 3. The effects of pre-processing on the training images

Model Implementation Details

We have used a Sequential Convolutional Neural Network from Keras as the architecture for the regression model. We use four 2D convolutional layers with a ReLU activation function and 5x5 kernels for the first 4 layers and 3x3 kernel on the fourth, using a normal kernel initializer. The number of filters for each convolutional layer is as follows: 64, 128, 256, 512. After each convolutional layer is a max pooling layer with 2x2 pool size and 2x2 strides. Following these is a fully connected layer with 256 filters on it, and also ReLU activation. The output layer is fully connected with no activation.

We employ the stochastic optimization algorithm Adam (Kingma et al,. 2014) with an initial learning rate of 0.0001 to learn the parameters of the neural network. The mini-batch size is set to 64, using 100 epochs. We used the Root Mean Squared Error (RMSE) as the loss metric.
Previous versions of the approach used the SGD optimizer, however as seen in Figure 4, the prediction error on validation images was significantly higher than using Adam with the same initial learning rate. Figure 4 also demonstrates how 0.0001 was a suitable learning rate for this problem, as it kept the prediction error under 5 for almost all of the validation set predictions. Max Pooling also helped minimize the network's prediction errors, as seen in Figure 5 as it helps reduce overfitting.
Although we experimented using regularization, as seen in Figure 5, it was not employed for the final version of this model as it damaged the predictions. Figure 6 also demonstrates this, as the predicted points from the regularized model are not aligned well with the face.

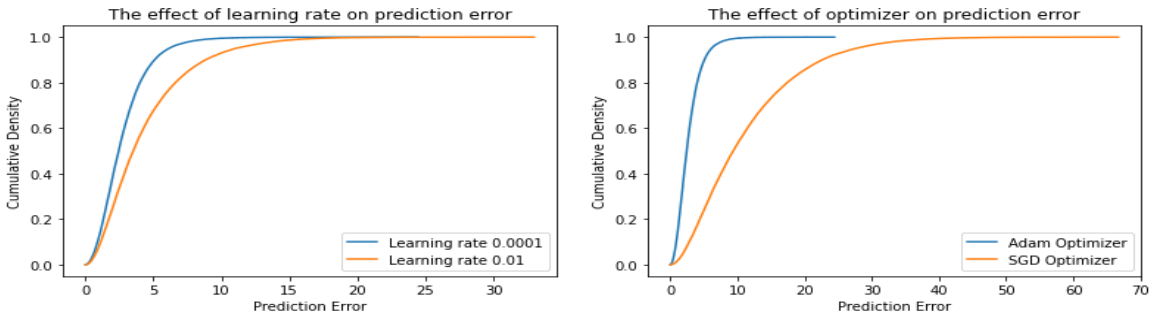


Figure 4. Cumulative Density Plots showing training parameter choice effect on prediction error

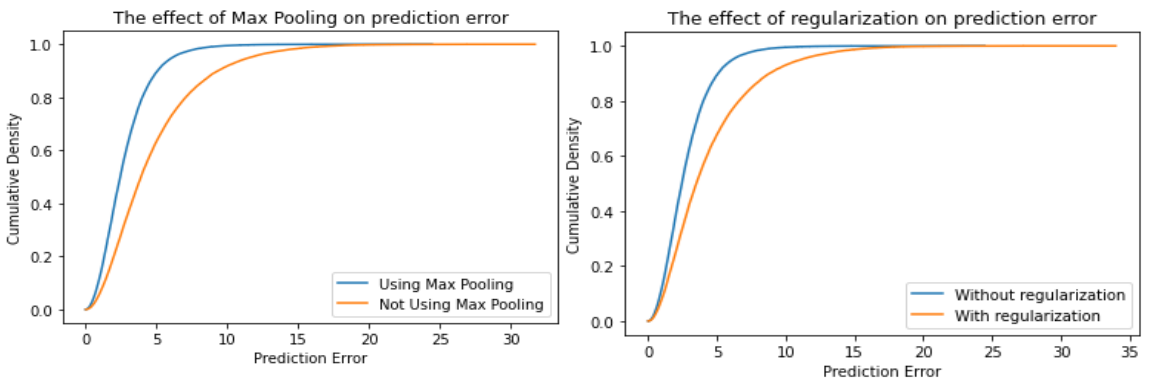


Figure 5. Cumulative Density Plots showing training parameter choice effect on prediction error

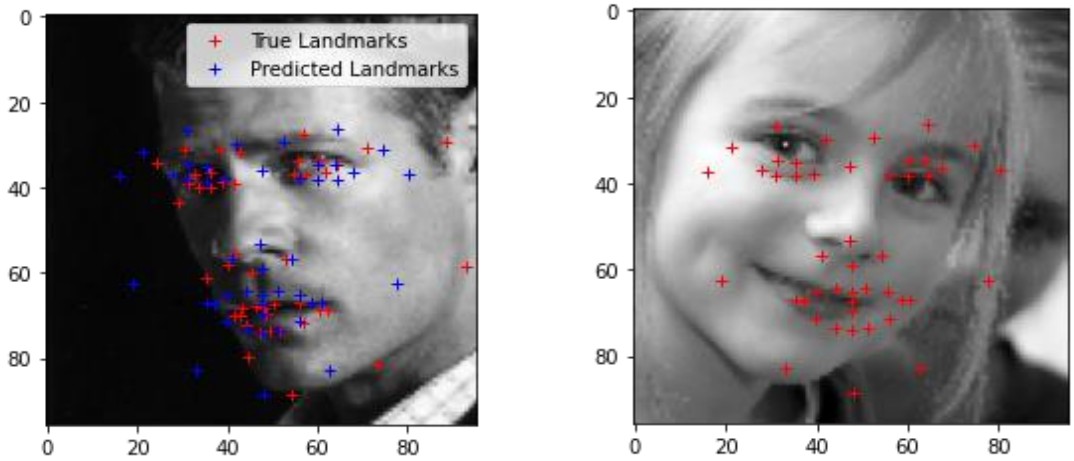


Figure 6. Predicted points on a validation image (left) and test image (right) using a model with regularization

Results

The neural network had a final R^2 score of 0.98 on both the training and validation datasets, although this doesn't provide a very useful description of performance. As seen in Figures 4 and 5, almost all of the predictions had a prediction error of less than 5. The notebook attached to this report shows the mean of the prediction errors was 2.5, and the standard deviation is 1.8, which show the success of the model.

The model performs relatively well on most images, including in a variety on poses. Although the ground truth landmarks and predicted landmarks do not align perfectly on the validation images, see Figure 7, the model makes reasonable attempts to adjust the landmarks based on a variation in pose.

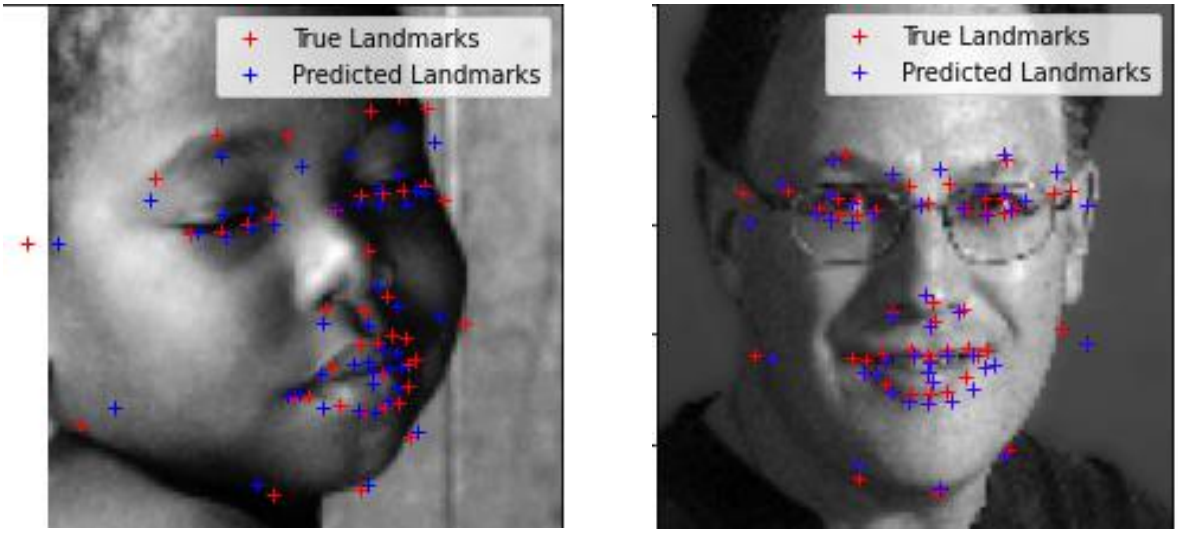


Figure 7. Comparison between the ground truth and predicted landmarks

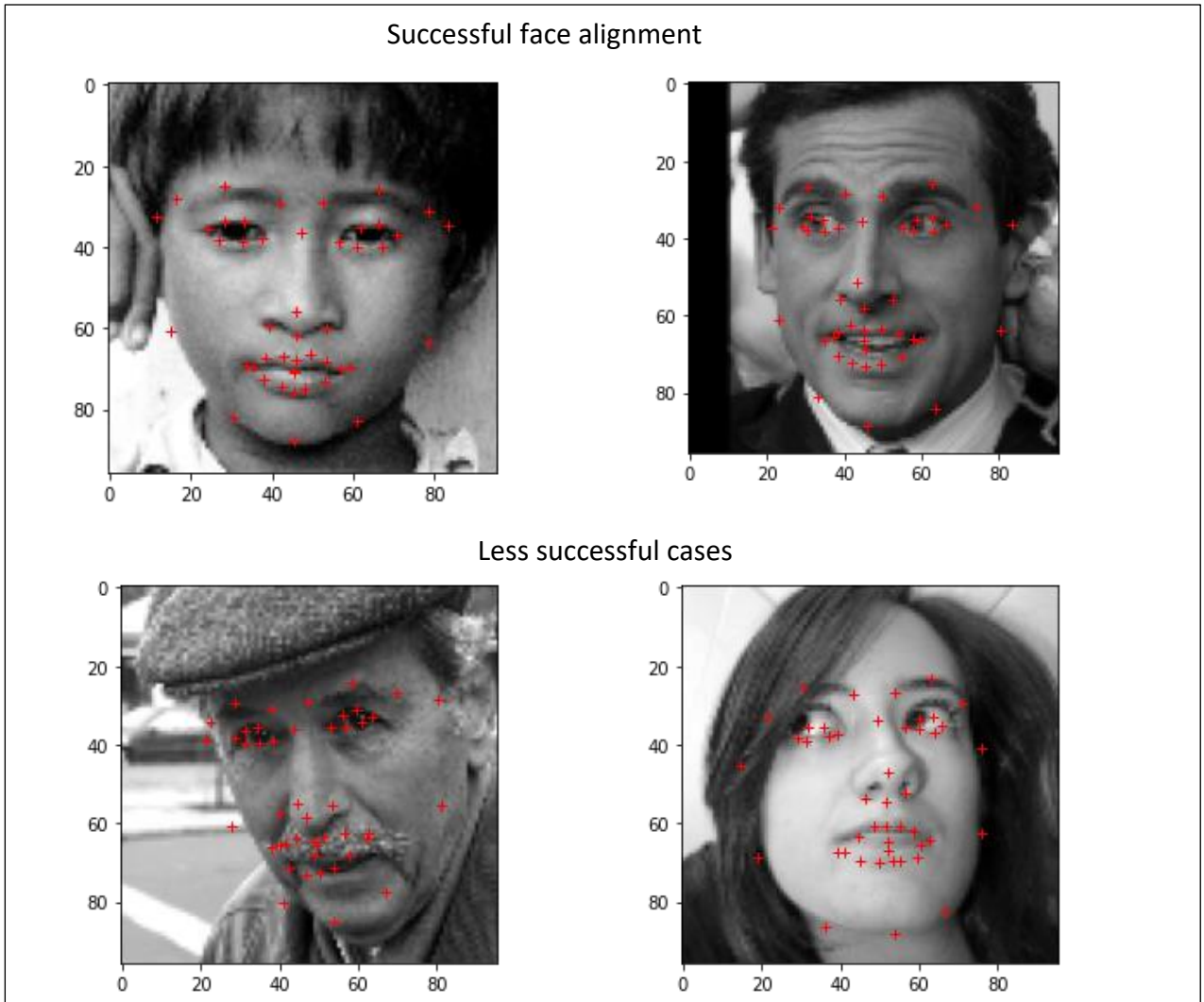


Figure 8. Predicted points on the test images showing successful face alignment (top) and less successful alignment (below).

Figure 8 demonstrates some successful facial landmark detection which capture the expression of the face well. The second half of images in Figure 8 show a reasonable but not perfect attempt at landmark detection. The first of these (bottom left), demonstrates the model's struggle to align faces with facial hair as his moustache has been incorrectly detected as the mouth. The second (bottom right) image highlights an instability in predicting facial landmarks in a wide range of poses, specifically here from beneath since the eyes are not correctly outlined.

Segmentation

We used a simple line segmentation approach using the labelled key-points as edges for the facial objects. This worked relatively well for the eyes and mouth, as these points had a higher concentration of key points than the nose and eyebrows. The eyebrows only had 3 points to align with, and this results in less than optimal segmentation. Similarly, the nose had 4 points outlining the shape of the nose, which also fails to include a variety of nose shapes, especially when the faces are at angles in the image.

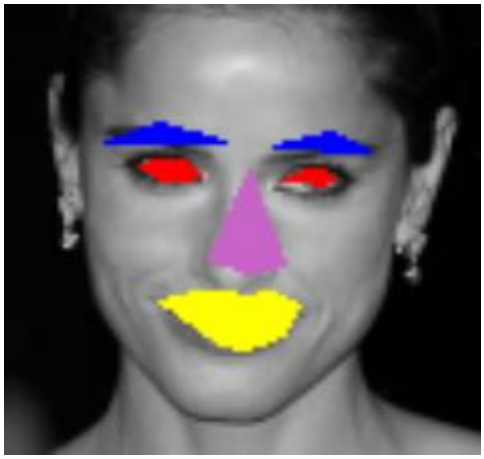
Segmentation

Legend for segmentation

Figure 9. Face segmentation using labelled training points and training images.

Figure 10 shows the segmentation applied to predicted landmarks in the test dataset, the first image (left) demonstrates a relatively successful approach to segmentation as the labelled areas are in the right vicinity, but do not align perfectly. The failure case (right) shows a poor attempt at segmenting the face.

Successful segmentation



Fail case

Figure 10. Face segmentation using predicted landmarks on the test dataset.

Graphical Effect

For the graphical effect, I first used MakeSense, a website which allows you to assign labelled points to an image, which I labelled using the indices from the facial landmarks in my model. This allowed me to align the points on the filter to the points in the face. Unfortunately, since the filter points didn't correspond well to facial landmarks, such as the tongue needing to originate inside the mouth, the ears on top of the forehead, the effect did not produce very successful results. The distance between the dog nose and start of tongue was also not desirable for my model. When the face in the image was facing the camera and the person was smiling, as such decreasing the distance between nose and mouth, there were some reasonable attempts. As seen in Figure 11, the effect did not perform well on any other variation in pose.

Reasonable performance

Fail case

Figure 11. Graphical effect on images using hand labelled landmarks

Evaluation

While our approach shows some success in correctly predicting facial landmarks, there are a number of areas where the model could be improved.
The reason for some of the failure cases may be where the whole face does not fall within the boundaries of the images, and the model cannot extrapolate. Perhaps the model would work better with the full images, and a bounding box face detection process first to ensure that all points fall within the image. There were also errors in prediction when the face had a significant amount of facial hair, which could be improved by collecting more labelled training images of people with facial hair. Although 100 epochs seemed to show good performance, using a check pointer to determine the optimal number of epochs using early stopping could have improved the performance and efficiency of the model further. The model also seems to struggle performing facial landmark detection on images of people with dark skin tones. This could be due to an imbalance in the dataset, or the fact the images are grayscale and do not capture the skin as well. Further pre-processing on the images, potentially changing the contrast and brightness, or collecting a larger dataset of POC may help avoid this. Architectural changes may also substantially improve the performance of the model, such as cascaded regression trees or an active appearance model.

References

1. https://www.kdnuggets.com/2017/09/detecting-facial-features-deep-learning.html
2. https://towardsdatascience.com/face-landmark-detection-with-cnns-tensorflow-cf4d191d2f0
3. https://www.makesense.ai
4. https://github.com/Spidy20/Insta_filters_with_python
5. X. Miao, X. Zhen, X. Liu, C. Deng, V. Athitsos and H. Huang, "Direct Shape Regression Networks for End-to-End Face Alignment," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 5040-5049, doi: 10.1109/CVPR.2018.00529.