Age Classification Using Convolutional Neural Networks

Yushan Wang, Qichen Huang, Shanshan Gong
Email: *yuw688@ucsd.edu, *qhuang5@uw.edu, *shanshangong.2000@gmail.com

***Abstract*—Automatic age classification has become relevant to an increasing amount of applications, but the performance of existing methods on real-world images is still significantly lacking, especially when compared to the tremendous leaps in performance recently reported for the related task of face recognition.**
This project aims to show that by learning representation through the use of different deep-convolutional neural networks, the performance shall increase. We propose to experiment with different convolutional net architectures that can be used when the data is lacking. This project focuses on the performance of each Convolutional Neural Network on the Adience Benchmark Project. The project contains multi-labeled faces images which are used in the training and testing phases of this project.
We examined the architecture that was developed by Gil Levi and Tal Hassner [1] , They used a 3-layer fully connected convolutional neural network, and can basically provide a prediction to each human faces. However, they are assuming the input images to their models are faces, and the accuracy was not exactly satisfying. Furthermore, we also examined other Convolutional neural networks. We produced another CNN architecture that could slightly outperform Gil and Tal's model. Then we combined our model with other face-detection pre-trained models, then we can have a input identification before making the age predictions. Later on, We developed our model through a webcam, which could be used in real-time age classification.

I. Introduction

Nowadays, the role of facial recognition and identification has significantly increased. Furthermore, Age has an important part of identification. People, in certain different ages, would choose their vocabulary differently, would have different hand gestures and other little habits. Even it was not a incredibly different task for human to recognize other people's approximate age, but for computer vision technology, this ability is still fundamental. The ability to automatically estimate the accurate age from certain facial images is the main training we are going to provide in this project.

More difficultly to our project is the lacking of certain face images with clearly labeled with age. Because with the increasingly strict privacy policy. It is increasingly difficult to manually fetch or mine such a face images data set, One might wonder why cannot we use the faces from public figure? To be entirely fair, we cannot restrict our origin of the data set to be celebrities only. This will certainly neglects the issue of lack of diversity. Moreover, celebrities tend to more care about their daily skin care, and would not ages as normal people do. Matthew Perry is 50 years old in the below image and Jennifer Aniston is 51 years old. In comparison, Jennifer certainly looks younger than Matthew if we only the knowledge from the picture.

Even though, we might able to predict the accurate age just from the picture, we still might able to know both of them are look like 50s. In this project, we aim to develop one CNN-based model for age classification, that would be able to give a general range of age from the image input.

II. Research Problems

By far, there are a few applications of age classifications. However, the demand for such techniques are bountiful, Websites with 18 age plus requirement, bues for the elders who are 60 or older, auto-payment with age identification. Our investigation is to focus on how to make this prediction could be more accurate based on the current baseline, Firstly, what types of the data are we need to gathering? We basically reviewed two solutions: 1. use current built data set which we found the Adience Landmark project, 2. Use artificial network to automatically generate human-faces, which we propose use GAN [2] [3] to automatically generate faces which could be facades, and could also generate partial of faces, faces wearing sunglasses,etc. The details of the Adience Benchmark project would be carefully explained in the Solution part. The Face generation by using GAN [4], we are putting into the section "Next Step", because in the current phase of our project, we are still using present built data set.

After the selection of data set, we then encounter the question of face detection. We did not have extremely high reliance on this model, Face detection is not the same definition of face recognition. Face detection is to detect whether there is a face present in the photo or in the video feed. If there is present one face or multiple faces, face detection need to localize where is the face or where are the faces. Face recognition, on the other

hand, not only detect the face, but also need to tell the identity of the face. In other words, face detection is to know where is the face, and recognition is to know where and who is the face. Thus, in this project, we will need a face detection tool to tell us where is the face in the video feed, then we could use that face as input to predict the age of that face.

Once we localize the region of the face, then we need to have a model to input the image and output the corresponding age range with certain accuracy. This is the part where we say "is lacking" in the previous introduction. The question is to use what architecture or which classification model to achieve this task?

III. Related Work and Literature Review

A. Adience Benchmark

By far, the open data set for face image is the Adience Benchmark Project [5] with the highest authority. Gil Levi and Tal Hassner used the same data set when they were developing their own age and gender classification models. The Adience Benchmark published in 2014, contains 26,580 photos across 2,284 subjects with a binary gender label and one label from eight different age groups, partitioned into five splits. The data set is not designed for predicting the accurate age, since in certain age the images are not plenty. It could be useful in predicting a range of age. The Adience data set provides 8 groups of the age, which are (0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60-). Indeed, the data set has plenty subjects of human faces, but most of the images are facade. The data set does not include the extreme conditions, such as only one side of a face. That could might be an influence during the testing our model after we used data set for training.

B. Age Classification

Demand for detecting the ages from the attributes from the facial images has increased in recent years and many methods have been developed. Gil Levi and Tal Hassner conducted a rather comprehensive survey about the early methods were used for the age estimation.

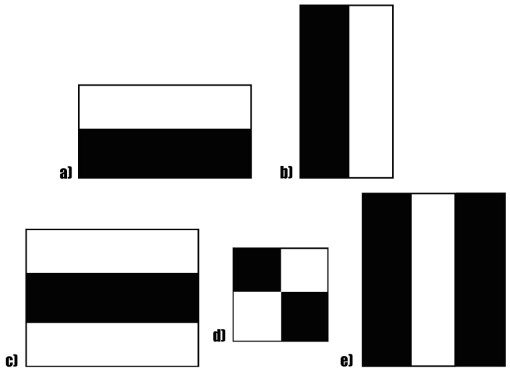
Age estimation methods in the early stage are based on the different measurement of facial feature landmark ratios. Facial landmarks include eyes, noses, mouth, even forehead. [6] [7] Once the landmarks are localized, then the ratios between each feature could be calculated and be used in the later classification.

Recently, regression and deep learning models have been deployed, Tal Hassner deployed a deep convolutional neural network as a model for the classification. Convolutional Neural Network has proved its productity in the computer vision task, and for MINIST data set, CIFAR-10 data set, the multi-labled classification task on these data sets were performed by the CNN model.

C. Face Detection

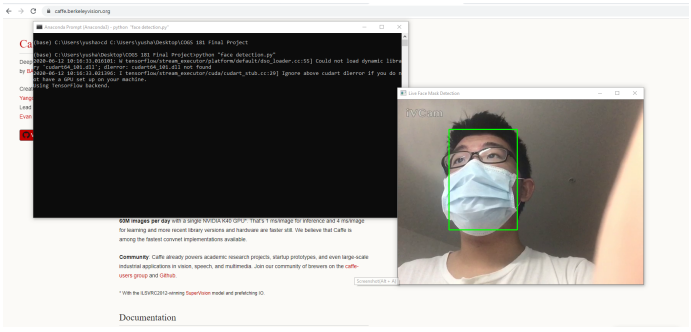
Face Detection, a widely popular subject with a huge range of applications. Mostly common used example would be the facial authentication. Most of the cellphones and laptops are installed in-built face detection softwares. Face photoshop, Facial movement recogntion, blinking tests, there are many apps have the capability to detect, capture, and process a face.

Haar Cascades Classifier Viola and Jones proposed the first object [8] [9] detection model in the real-time video footage, and later on was known as the Haar Cascades. It was originally designed for object detection. By calculating the Haar features, it could realize the detection. Haar features are on the image makes it easy to find out the edges or the lines in the image, or to pick areas where there is a sudden change in the intensities of the pixels. The figure below is one demonstration of the Haar Cascade.



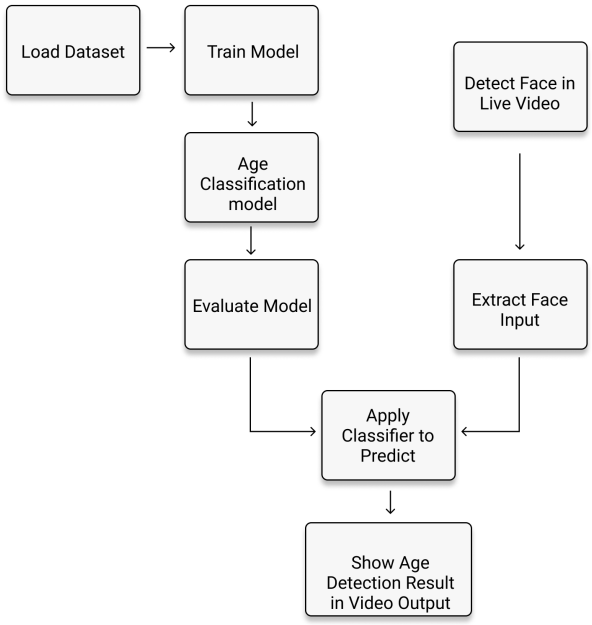
Many computer vision or deep learning would have pre-trained haar cascade model for certain object detection. However, one of the biggest deficit of those pre-trianed models, is that they cannot detect the object when there is only pieces or parts of an object. For instance, if you wear a mask, or a sunglasses, then there exists high probability that the Haar Cascade pre-trained face detection model cannot find your partial face.

Deep Neural Network (DNN) It is a Caffe model which is based on the Single Shot-Multibox Detector (SSD) and uses ResNet-10 architecture as its backbone. The model in the real-world deployment works the same as the Haar-Cascade classifier in the normal condition: Both of them could detect a clear face when a frontal face is present in the input footage. What makes the caffe model actually stands out is the detecting ability in the extreme condition. The model still could recognize the face when the test object is wearing a facial coverings or deliberately showing slight part of the face. Below is one demonstration of the caffe-DNN working in the extreme condition.



IV. Method (Solution Design)

This project is going to develop a Real-time Age classification application. Thus, we need to come up with design about how to handle Real-Time Video input, and how to produce the result with minimum delay. We also certainly need to develop an age classification model, to classify the age from the input images and videos. Then the whole solution design could be separated into the following parts: Face Detection, Age Classification, real-time video capture and processing. As divide and conquer, after we separate the model into several parts, we then just need to come up with details of how to achieve each part. Furthermore, because our target is to build an age classification model with a prototype. Thus, it is critical to have a general design baseline of the entire project. Below the figure is the general process of our design,



As one can notice, on the left-hand side is the process of creating an age classification model. This is the basic process of generating a trained neural network model. One would need to find the proper data set and load the data, complete the data processing and do appropriate data cleaning. After all the procedures with data, then we need to build our baseline for our convolutional neural network model, then start training, validation and testing with the loaded data set. On the right-hand side is the process of face detection. At the bottom is to deploy both models, and produce the output result.

A. CNN for Age Classification

First of all, we are going to reproduce the deep convolutional neural network architecture presented by Gil and Tal Hassner. Then we are going to experiment with a few different architectures to see if could outperform the current result. As they mentioned, the CNN architecture is 3 convolutional layers which they also frankly admit that this architecture is not overly complicated. We then will try to improve their models by slightly amend the architectures and try to develop a few well-known architectures. Then we will train and test the model, if the accuracy is acceptable, then we are going to deploy that to our final application. If the accuracy is not acceptable, then we are going to figure out which part causes the low accuracy. This is just the assumption, but it is generally not highly likely, since we are reproducing Tal Hassner's model since their accuracy is fine, then the outcome should not be too surprising. Once we created the age classification model, then we could deploy and use for predictions.

B. Face detection

There are a lot of pre-trained face detection models, mostly known as the four listed in below. The Haar Cascade pre-trained face detection classifier, Dlib Frontal Face Detector, MTCNN, and caffe [10] DNN Face Detector model. The reason we are choosing to use pre-trained model instead of building our own model is that the pre-trained model could perform properly in our project, and building our own face detection model would be time consuming, Our plan is to experiment the above four pre-trained models, and test which one is the optimal, and to see if one could qualify for the project. Because there are a lot of demonstrations for each model, our presumption is that the Haar Cascade and the caffe-DNN model shall outperform the rest of the two in this task. The details of which model we choose as final result would be revealed in the implementation section.

For the final Deployment, once we have our age classification model, and picked our face detection model, we need to combine the two models and deploy them to an interface. We come up three different deployments to achieve real-time age classification.

Webcam Since most the laptops are equipped with webcam camera, then this could be one of the main input for real-time video. Since our age classification model and face detection models are all developed and finished on our local machine, then it is not hard to complete all those stuff on a local machine which is equipped a camera.

IOS deivce such as Iphone After we deployed the application on PC and laptops, we then can place our focus on how to deploy that on IOS device. Iphones and ipads are now both have cameras. We could use that as our image input. We then could wrote an app to deploy our age classification model to the IOS device.

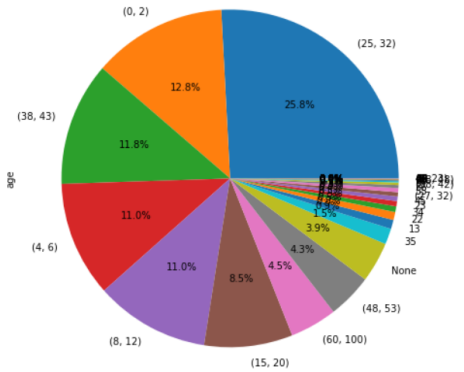
Cloud Configuration If we could deploy our model on cloud, then most of users could avoid spend extra time and resource to reproduce our result. They could just use our model on the Cloud. So far, we came up with the idea of by using Docker to deploy the model, and everyone has the docker access would have access to the code. Later on, we hope to deploy the application onto the cloud platform such as Azure.

This is basically the whole design for the project. The most crucial part lies in the age classification.

V. Implementation

Following the structure as we described in the Method (Solution Design) section, we are going to implement each part step by steps. First of all, we are going to implement the Age classification model.

Adience Benchmark includes 26,000 images of 2,284 subjects, and we created a Pie chart to visualize the age distribution.



and after we classify all the edge cases into the belonging age ranges. Below is the number of images we have for each label.

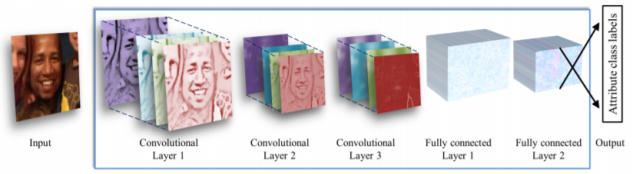
Age	0-2	4-6	8-13	15-20	25-32	38-43	48-53	60-	Total
Image	2509	2140	2292	1792	5296	2776	916	901	17721
Comment									

What to be noticed here is the total images might not match the total number of images as we described the data set early on. The reason for this difference is because the 26k images were the statistic number is the measure at first during the publish. At the moment, we used Adience, the original data set contains 19k images and after we excluded "none" images, (the image does not have a age label), what we got is the number of 17,721.

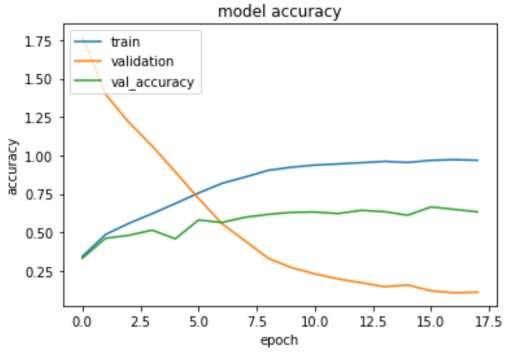
Data Processing We performed a center cropping with a uniform size of 227x227. This is actually follows what Tal did for feeding the network with the face image, cropped to 227 × 227 around the face center. Below is an example of the face image without cropping.

After the data processing, we come to the part of building our architecture.

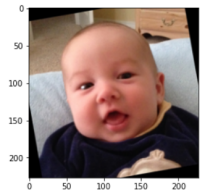
Simplified AlexNet AlexNet is a 5 fully connected convolutional layer network, published by Alex Krizhevsky. The whole structure resembles to the first CNN net - Le-Net. The reason we name our first model to be simplied AlexNet is because a complete AlexNet is 5 layer but the architecture published by Tal is 3-layers. Below is the original architecture of the Tal's model.



We then reproduce the CNN architecture in TensorFlow. We divided the whole data with a batch size of 32. We monitored the training loss, training accuracy, validation loss, validation accuracy. The training loss decreased sharply in the first few epochs, so does the sharp increase of the training accuracy. On the other hand, the validation accuracy is basically not improving after epoch of 8. The architecture presented by Tal has the issue of slightly over-fitting. Below is the graph we monitored the trend of the accuracy.



Training and Testing Since we did not use pre-trained model in this case, our simplified AlexNet was trained from scratch all the params were not pre-set before the training, all the labels are one-hot encoded to a binary vector with a size of 8(since there are 8 classes of age.) Also, at this stage, we also realize that if we used a face recognition pre-trained model to augment on the ability for age detection. There might be the possibility to outperform a model which was trained from the beginning.



```
predictions = AlexNet.predict(test_sample.reshape((-1,227,227,3)))
print('predicted probability: ' + str(predictions))

predicted_label = AlexNet.predict_classes(test_sample.reshape((-1,227,227,3)))
print("prediction labels: " , str(predicted_label))
```

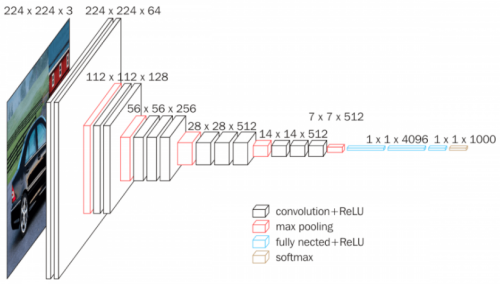
```
predicted probability: [[9.9880815e-01 1.3608072e-04 1.9162569e-04 2.9957542e-04 2.6140598e-04
  1.3245844e-04 4.3267220e-05 1.2751762e-04]]
prediction labels:  [0]
```

```
encoder.inverse_transform(predicted_label)
```

```
array(['0-2'], dtype=object)
```

Then we tried the complete AlexNet to see if it could works slightly better than the Tal's model, with 2 more Convolutional Layer, the model converges faster than the Tal's model, but the overall accuracy is basically the same. When testing the model, the eventual test accuracy was between sixty percent and sixty-five percent.

VGG-16 [11] proposed by K. Simonyan and A. Zisserman from the University of Oxford, we then use to cross-compare the result of our CNN classifier. VGG-16 replaces large size kernel filters with a stack of 3 x 3 filters, while each uses stride size of 1 in convolution layer. Furthermore, the spatial padding is 1 with a stride size of 1 for the 3x3 convolution layers, while the padding in 2x2 max pooling layers has a stride size of 2. Each hidden layer in the VGG-16 structure include a ReLu layer to deal with non-linearity. VGG-16 was mainly used for image classifications and detection, so we deployed this model to perform a transfer learning.



In the Convolution Neural Networks that we constructed to detect the age range, there are two hidden layers in the structure, in which we used ReLu layer and Max Pooling layer. The choice of activation function we selected in the end is the softmax function.

The Rectified Linear Unit serves the purpose of breaking up the linearity during image processing. The convolution operations might impose linearity on images that are essentially non-linear, which would undermine the learning ability of the model. ReLu removes all the black elements in the image, leaving only the white and grey colors, so that we can observe the change in the patterns more abruptly after this operation. On the other hand, the max pooling layer is used to teach the convolution neural networks regardless of the background and texture differences that may affect the accuracy of the model. Similar construction of the layers are being specified when trying to train the models, and we have achieved considerable accuracy using this strategy. The accuracy was slightly better than the AlexNet.

VI. Lesson Learned and Next Step

Lesson Learned We reproduced the results of Tal Hassner published in the 2015 CVPR. We noticed what improvements could be done, and what kind of architecture would be useful to implement in order to elevate the accuracy. We acquired a lot useful knowledge from the Adience Benchmark Project, we solved the problem that the original model cannot work properly in the extreme condition by using a more complicated architecture and tested with more extreme conditions. We also solved the problem that the original model relies on the presumption that the input image would be a face by add a face detection model in the baseline before the age classification model. Furthermore, we acquire much useful experience in image processing when we are handling the image inputs: the oversampling and the center crop. We acknowledged not only handle the image input but also capable to handle recorded and real-time vide input. The OpenCV indeed has many handful built-in functions could be deployed, but more importantly is to understand how and why to use which functions. We used the knowledge from computer vision and supervised learning, and neural network. We provided a quite simple solution to tackle a real-world problem, and we planned for improving our current solution.

Nest Step The current prediction model still exists space for improvement, The accuracy is not overall satisfying, We are hoping to experiment more CNN net architecture and also to see if there is any improvements with the image pre-processing. In this project, we only experiment two models. First the model presented by Tal, and the second is VGG-16. VGG-16 is a well-known CNN architecture, but the performance on this task is not extremely satisfying. We hope to increase the accuracy further on. Also for the data processing part, we only encoding the labels and only performed center crop on each image. To further improve our models, certainly there are more pre-processing procedure could be done. Since the methods in the early stage were relying on the facial landmark, maybe the Gaussian Filter to extract the outline of the image could be useful in improving the predictions. The outline extracted from the facial images would be easier to detect each facial features; in other words, easier to for our model to notice which part is the mouth which part is the chin,

The cost of the experiment also could be decreased. This project is running on A local machine with a 3840 CUDA cores and 16 Gigabytes of video memory. Traing each network is about 10-15 minutes. Even the time cost is minute, the cost for memory is gigantic. Because the Adience data set is about 1.6 GB and it is tolerable on our training machine, it does not mean this could be optimized. Prediction running times can conceivably be substantially improved by running the network on image batches. The 1.6 GB images, on our process was divided into 32 batches. If the memory on another tester is less than 16 Gigabytes, divided into more batches would in some way solve the issue.

Furthermore, there are huge spaces of improvement could be done for the deployment. So far, we have only deployed the model on the laptop by using the default webcam. The next step is to deploy our model on IOS devices and Android devices. We could develop an application on the IOS devices, and Android devices. We could use CoreML to transform our pre-trained age classification model into the IOS devices. CoreML has been used for object detection on IOS devices. User could use the pre-trianed models come with the app, or also developed their own model for different purpose. With the access to the camera. For the cloud deployment, the simplest way for the next step is just to create a docker to let user could

access our code throught the docker platform, or we could deploy that as IOT on the cloud platform such as Azure.

Since the user interface is not the main target for this project. We basically designed nothing to cope with potential users or customers. A simple website could be designed. Speaking of which, we are currently using django to create a website for age estimation. There are two webpage in the design, one is for the image upload, the other is for the real-time video feed with access to your camera, The UI is still in designing process. Furthermore, if the app portal is going to be deployed. Then we also need both front-end and back-end engineering for our IOS app designs. For the cloud deployment, the configuration could just use our pre-trained model, instead of using extra cost of time and human resource to train a completely new model. This is our expecting configuration so far. The actual deployment still has much details in discussion. What we can imply from what we have created for now, this application does have a potential future.

VII. Conclusion

We finished building the age classification model, and successfully deployed to our webcam application. We examined the CNN model architecture developed by the Gil and Tal's model. The accuracy was between sixty to sixty-five percent and we found a little over-fitting of that model. Our model based on the VGG-16 eventually could reach an accuracy between sixty-seven to seventy-two percent. Our model outperform the convolutional neural network architecture in the Gil and Tal's model. This also proved the expecting conclusion mentioned by Tal, " more elaborate systems using more training data may well be capable of substantially improving results beyond those reported here." Indeed, the VGG-16 based model outperforms the simple three layered architecture. On the other hand, we realize both of the architectures have the shortcomings when predicting the age, even the accuracy is slightly improved, our model still could be improved in the next step development.

For the deployment part, we now have a real-time age detection application by using webcam model. As it has been fully explained in the implementation section. Though many methods have addressed the age estimation, but rarely any of them ever mentioned the real-world deployment. So far, our simple deployment showed its solution on laptop through webcam. As mentioned in the Next Step Section, we are planning to use CoreML to implant our model onto IOS device. It is also could be an useful IOT to deploy on multiple cloud platform. Our solution now can show the utilities of age estimation. The simple deployment we used in this project also implies that with more deliberate and careful user interface designs and user feedback, the age estimation application could be wide-ranging.

References

[1] G. Levi and T. Hassner, "Age and gender classification using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[2] Y. Huang, F. Juefei-Xu, R. Wang, Q. Guo, X. Xie, L. Ma, J. Li, W. Miao, Y. Liu, and G. Pu, "Fakelocator: Robust localization of gan-based face manipulations," 2020.

[3] H. Zou, K. E. Ak, and A. A. Kassim, "Edge-gan: Edge conditioned multi-view face image generation," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2401–2405.

[4] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," 2015.

[5] S. Lapuschkin, A. Binder, K.-R. Müller, and W. Samek, "Understanding and comparing deep neural networks for age and gender classification," 2017.

[6] X. Geng, Z.-H. Zhou, and K. Smith-Miles, "Automatic age estimation based on facial aging patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2234–2240, 2007.

[7] G. Guo, G. Mu, Y. Fu, and T. S. Huang, "Human age estimation using bio-inspired features," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 112–119.

[8] H. Yang and X. A. Wang, "Cascade classifier for face detection," *Journal of Algorithms & Computational Technology*, vol. 10, no. 3, pp. 187–197, 2016.

[9] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

Appendices

The code for this project could be found in the below link, or if you view this project in Electronic form, you can also click here.

Github Link to the project:
https://github.com/yushan1089/ST8601_Age_detection

More details of this project could be found on Yushan Wang's personal website:
https://yushan1089.github.io/projects.html