NANYANG TECHNOLOGICAL UNIVERSITY

CE4042 Neural Networks and Deep Learning

Assignment 2

Team Members:

Chockalingam Kasi U1920428E

Gavin Neo Jun Hui U1921265L

Tan Kah Heng Darryl U1921321H

Content Page

Introduction

Neural Networks is a set of networks or circuits of neurons to mimic the biological neural network found in the human body. The name and the structure are inspired by the human brain. The artificial neural networks comprises node layers, starting with input layer, hidden layers and output layer. Each artificial neuron is connected to another neuron and has its own weight and threshold determined by its activation function. Neural Networks and deep learning can be used for regression analysis, classification and data processing. Neural networks can be built from scratch and the parameters can be fine-tuned until it is functional. Another approach would be transfer learning, where a pre-trained model can be used with further modifications made to published architectures.

Problem Statement

This project is tasked to build an automatic gender classification model to be used in image analysis that can be used for social media platforms and other areas. We are required to classify the gender of the faces in an image. We can use Convolutional Neural Network (CNNs) to tackle the image classification task in this project. This is a binary classification problem where we will determine whether the label is either 'male' or 'female'. Most datasets are acquired with controlled lighting, lack of blurriness and face aligned to the camera. However, in practicality, photos may not always be captured in a controlled environment. As such we will be training our model using the Adience dataset [1] that include images captured in real-world imaging conditions. The images have varying poses, lighting and varying appearance. The dataset consists of 26000 real-life images with separate files containing information such as image id, gender, age range and angle information. We will be using Jupyter Notebook and Google Colab Pro to train our model. The artificial neural networks will be using TensorFlow a free and open-source software library for machine learning and Artificial Intelligence. We will be using this method to build a model to determine the gender and age of the face on the image.

Project Requirements:

1) CNN model for simultaneous gender classification and face age estimation using the image
2) Implement Transfer Learning using a previously published architecture
3) modify the parameters of the architecture or increase depth of the network

Data Preparation

Preprocessing is an important step in the CNN training where we will have to ensure that the data used in the training of the model meets the necessary requirements. The Author states that the Adience dataset consists of 26000 images. After downloading we have around 19370 images that are labeled. In the raw data form, we have about 5 Text files containing the image information and images split into multiple folders. The data has originally been built for 5 folds used for k-fold cross validation to measure the model performance. The images are differently sized and hence they need to be resized into a standard form. Using Jupyter notebook, we use pandas to merge the text files and keep key columns of the information such as original_image, gender and age that are important for this project. We can save the data to a csv file for further usage.

Since we are building two models, for classifying gender and ages, we will keep 2 copies of the data separately for training of each model. For the gender classification model, we will sort the images based on the **gender into two folders** named male_face and female_face. For the Age classification model, we will sort the images based on the age groups. We have **7 age categories** of age groups consisting of ages (0 to 3), (4 to 6), (8 to 13), (15 to 23), (34 to 54) and (55 to 100). Out of the available 19370 images **only about 12670 images are properly labeled**. Using the image ids we were able to find that some images had both male and female labels and some of the rows were empty. Therefore, after analyzing, there are 12670 valid images. These images will be resized into 256 x 256 pixels and they will be split into training and validation data. The code used for data preprocessing is attached to this submission as well.

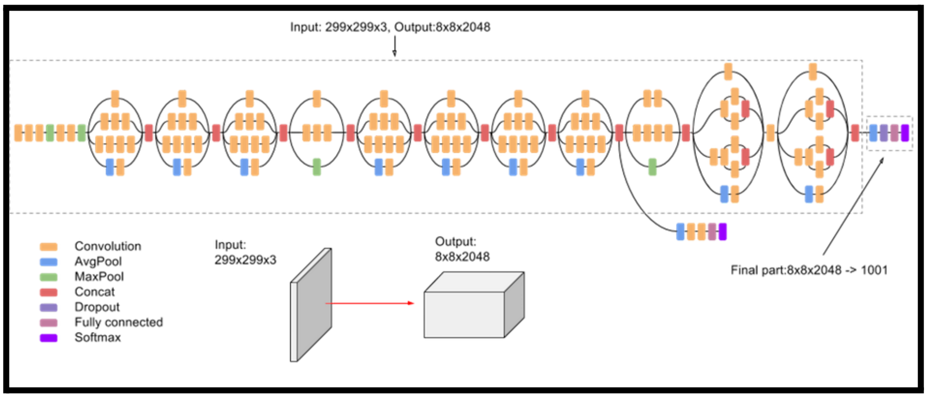
Review of Popular CNN Architectures

The introduction of Convolution Neural Networks (CNN) helps to advance computer vision tasks and improve performance in various computer vision tasks, like object detection, image classification and semantic segmentation. The first CNN that gained huge attention was the AlexNet, developed in 2012, which won the ImageNet Large Scale Visual Recognition Challenge for that year. It was developed using 5 convolution layers with ReLU activation functions used for each layer. 2 year later, VGG-16 [2] and Inception-V1(GoogLeNet) [3] were developed, resulting in better performance than AlexNet. The focus of the VGG-16 architecture was to stack more convolution layers on top of each other while the focus of the Inception-V1 architecture was the introduction of the 'Inception modules', which were developed from the Network in Network idea proposed by [4] and will be further explained later. Around this time, there was a common consensus that deeper modules would result in greater performance. However, deeper models often face a problem in the form of vanishing gradients. This hinders the training effectiveness for the lower layers of a huge network. Therefore, the ResNet architecture [5] was created to tackle the vanishing gradient problem using skip connections. From then on, many CNN models were improvements of these 3 models (VGG-16, Inception and ResNet), like the Inception-V3 model [6], the Inception-ResNet-V2 model [7] and the ResNeXt model [8]. As we are still new in learning and exploring Neural Networks, we have decided to review and understand one of these models, which will be the Inception-V3 architecture model.

Convolutional Neural Networks with Transfer Learning

Description of Methods

This project will implement a convolutional neural network using Inception-v3 to classify the gender and age. Inception-v3 was trained using the original ImageNet [9] dataset, consisting of 14 million images from 1000 classes. We'll use Inception-v3 as our base architecture and modify the published architecture and train with the Adience dataset which contains about 12000 properly labeled images.



Inception-v3 Architecture [6]

The inception v3 was a game-changing technique in the development of CNN classifiers. Most CNN architectures were built by stacking convolution layers one after another to achieve better performance. Hence, in order to learn more about the network, we decided to use the network for our gender classification task. Inception module A uses factorizing convolutions with two 3x3 convolutions replacing 5x5 convolutions to reduce the parameters, module B factorizes into asymmetric convolutions with 3x1 filter followed by 1x3 filter and module C is proposed for high dimensional representations. An auxiliary classifier is used for regularization of the data. The grid size is reduced efficiently by convolutions with stride 2. 320 feature maps are obtained by max pooling. Two of the 320 feature maps are concatenated together. It stacks 5 module A, 4 module B and 2 module C with grid size reduction layers in between the modules. The last layer would be an auxiliary classifier. This base architecture would be further modified to perform better in classifying gender.

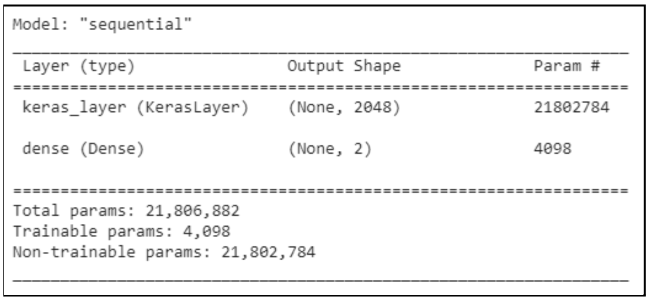
Methods to improve accuracy of Inception-v3 using modifications to the published architectures:

1. Data augmentation: Increase the diversity of the images by flipping images horizontally and random rotations to add modified copies of existing data and create new data from the existing data. We'll not use transformations based on color as they may not be relevant for this gender classification.

2. Adding Dropouts: Dropout layers prevent overfitting of training data. It is a regularization method to train the neural network with different architectures by randomly ignoring outputs of the neurons. Our dropouts will have a value of 0.2.

3. Dense Layer: Dense layers are added as part of the hidden neural networks with a relu activation function. The dense layer is a neural network layer receiving input from neurons in the previous layer. It also contains an activation function which will determine whether to transmit further into the network.

Basic Inception_v3 architecture

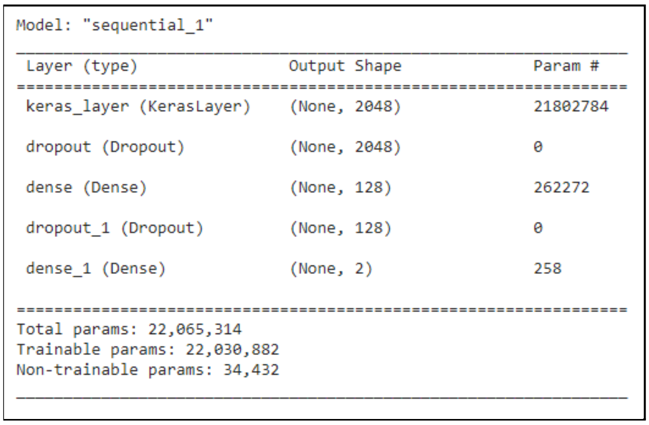


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 2048)              21802784

 dense (Dense)               (None, 2)                 4098

=================================================================
Total params: 21,806,882
Trainable params: 4,098
Non-trainable params: 21,802,784
_________________________________________________________________
```

Modified Inception_v3 Architecture



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 2048)              21802784

 dropout (Dropout)           (None, 2048)              0

 dense (Dense)               (None, 128)               262272

 dropout_1 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 2)                 258

=================================================================
Total params: 22,065,314
Trainable params: 22,030,882
Non-trainable params: 34,432
_________________________________________________________________
```

The steps involved in training the neural network model would start by importing libraries and steps will be taken to prepare the pre-processed dataset images. The second step would be stacking the layers sequentially in the model. The model is compiled with an early stopping feature to monitor the cross validation loss. When the accuracy of the model has converged, it will monitor for 10 epochs following which it would stop the model fit process. Callbacks are used to save the best weights by monitoring the val_loss feature.

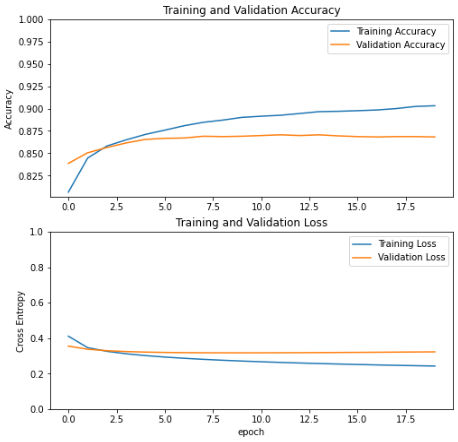
The basic architecture will be built using the Inception-v3 model that has pre-trained weights. This architecture is used to understand the raw accuracy without modifying the model. In order to improve the retrieved model, we will carry out data augmentation, add dropouts and include dense layers with 128 hidden neurons. We will use the same model for gender classification and face age estimation. Males and females have varying facial features such as length of hair, facial hair and other facial features that are noticeable. Age estimation is also an interesting image analysis task, which is used to determine the age of the person from the set of images. People from different age subgroups have different appearances at each stage of their life.

Experiment and Results

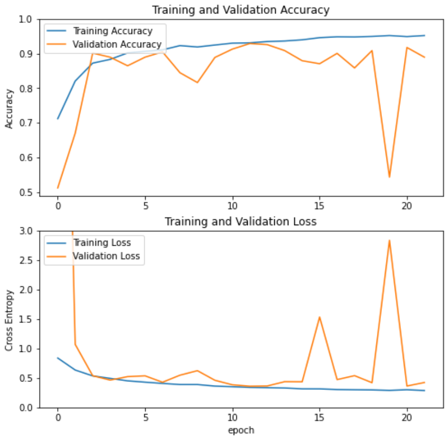
For Gender classification, the modified inception architecture has a validation accuracy of 0.92, which performs better than the basic inception architecture with validation accuracy of 0.86. We can notice that the additional layers added helps to improve the classification of genders. We can also notice improvement in the age classification from 0.55 to 0.64 in validation accuracy.

Basic Inception_v3 Architecture	**Modified Inception_v3 Architecture**

Gender Classification

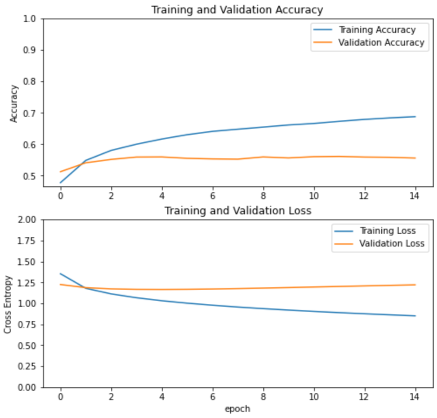


Val_loss: 0.3173 Val_acc: 0.8690

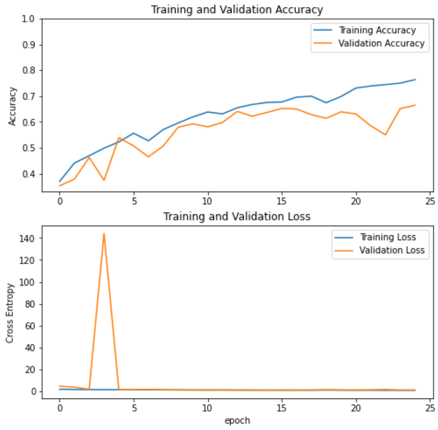


Val_loss: 0.3596 Val_acc: **0.9290**

Age Classification

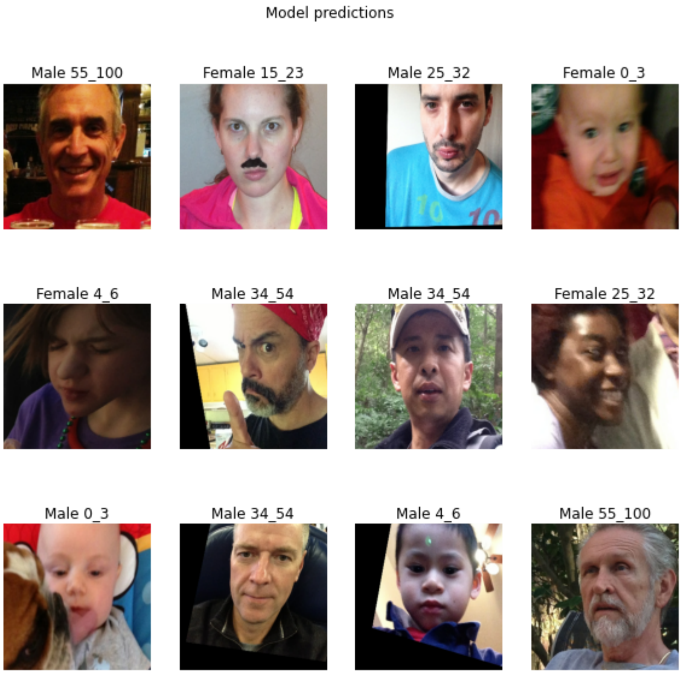


Val_loss: 1.1653 val_acc: 0.5597



Val_loss: 1.1579 Val_acc: **0.6368**

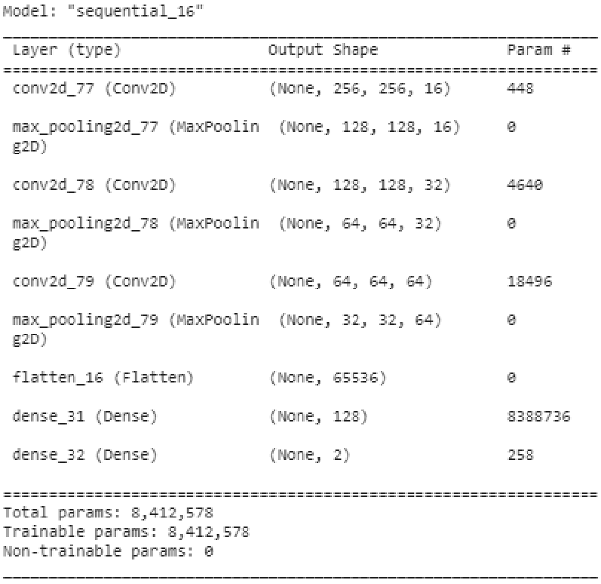
The images below are the predicted gender and age classification by the modified Inception_v3 architecture.



Gender classification without transfer learning

Description of Methods

In this part, we want to compare the results of using transfer learning and building the model from scratch. We first designed a simple model, to kickstart the classification of genders.



```
Model: "sequential_16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_77 (Conv2D)          (None, 256, 256, 16)      448

 max_pooling2d_77 (MaxPoolin  (None, 128, 128, 16)     0
 g2D)

 conv2d_78 (Conv2D)          (None, 128, 128, 32)      4640

 max_pooling2d_78 (MaxPoolin  (None, 64, 64, 32)       0
 g2D)

 conv2d_79 (Conv2D)          (None, 64, 64, 64)        18496

 max_pooling2d_79 (MaxPoolin  (None, 32, 32, 64)       0
 g2D)

 flatten_16 (Flatten)        (None, 65536)             0

 dense_31 (Dense)            (None, 128)               8388736

 dense_32 (Dense)            (None, 2)                 258

=================================================================
Total params: 8,412,578
Trainable params: 8,412,578
Non-trainable params: 0
_________________________________________________________________
```

Basic Architecture from scratch

The model consists of 3 convolution layers, 3 max pooling layers and 2 dense layers.

Data Augmentation

The input images undergo data augmentation. This is to increase the input samples, and also to reduce overfitting.

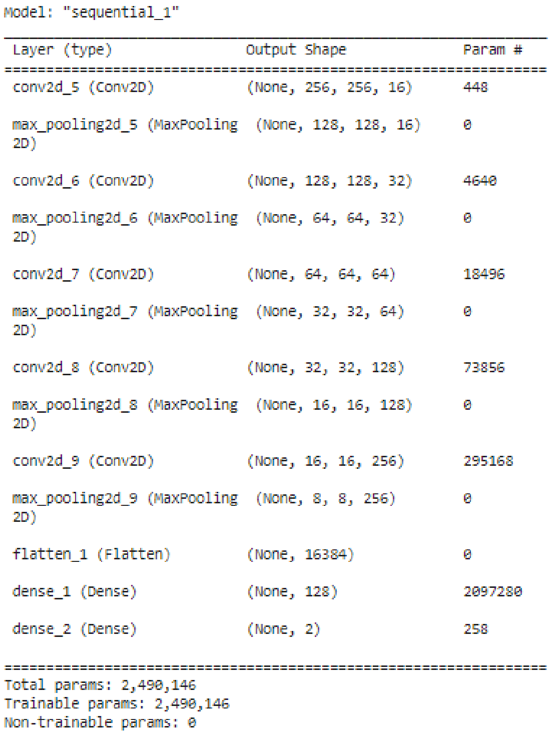
Optimization of gender classification

To improve the performance, we optimize the model. For this, we decided to increase the number of convolution layers. For each convolution layer added, we added 1 max pooling layer after each convolution.

Number of convolution layers	3	4	5
Validation loss	0.313	0.274	**0.242**
Validation accuracy	0.871	0.893	**0.903**

From the table, having 5 convolution layers results in the lowest validation loss and highest accuracy, and thus we decided to keep the model with 5 convolution layers for gender classification.

Age classification without transfer learning



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_5 (Conv2D)           (None, 256, 256, 16)      448       
                                                                 
 max_pooling2d_5 (MaxPooling  (None, 128, 128, 16)     0         
 2D)                                                             
                                                                 
 conv2d_6 (Conv2D)           (None, 128, 128, 32)      4640      
                                                                 
 max_pooling2d_6 (MaxPooling  (None, 64, 64, 32)       0         
 2D)                                                             
                                                                 
 conv2d_7 (Conv2D)           (None, 64, 64, 64)        18496     
                                                                 
 max_pooling2d_7 (MaxPooling  (None, 32, 32, 64)       0         
 2D)                                                             
                                                                 
 conv2d_8 (Conv2D)           (None, 32, 32, 128)       73856     
                                                                 
 max_pooling2d_8 (MaxPooling  (None, 16, 16, 128)      0         
 2D)                                                             
                                                                 
 conv2d_9 (Conv2D)           (None, 16, 16, 256)       295168    
                                                                 
 max_pooling2d_9 (MaxPooling  (None, 8, 8, 256)        0         
 2D)                                                             
                                                                 
 flatten_1 (Flatten)         (None, 16384)             0         
                                                                 
 dense_1 (Dense)             (None, 128)               2097280   
                                                                 
 dense_2 (Dense)             (None, 2)                 258       
                                                                 
=================================================================
Total params: 2,490,146
Trainable params: 2,490,146
Non-trainable params: 0
```

Description of Methods

Using the best model from gender classification, the model performs age classification. The performance has a lowest validation loss of 1.06, and validation accuracy of 0.598. The performance is worse compared to gender classification. This is due to the fact that there are a total of 7 classes to classify the age group of the images, as compared to binary classification of gender.

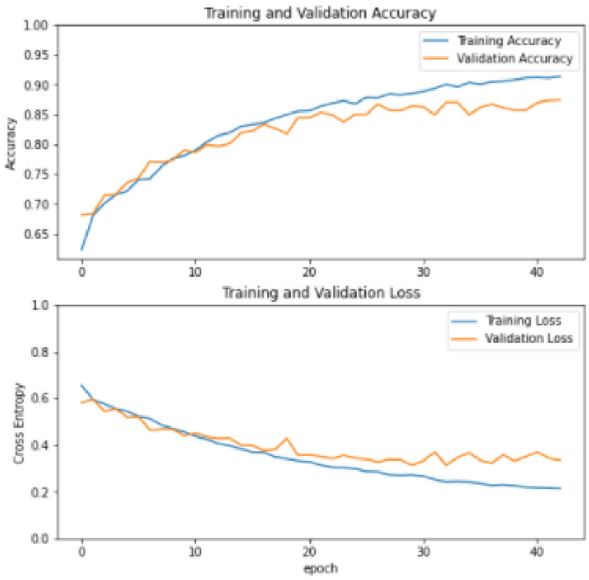
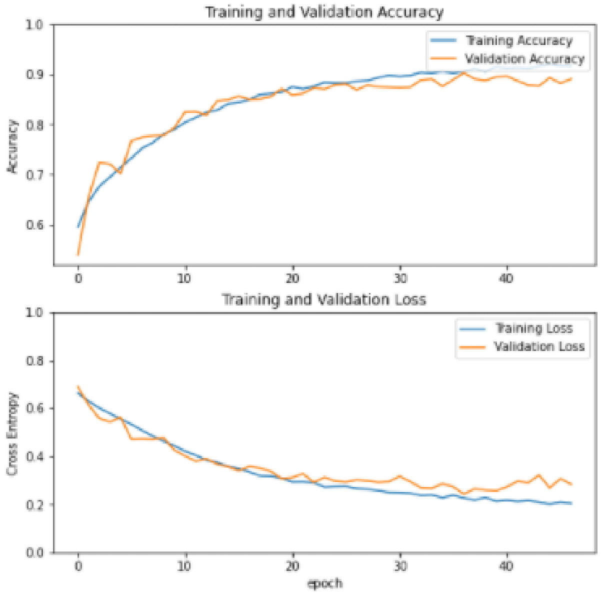
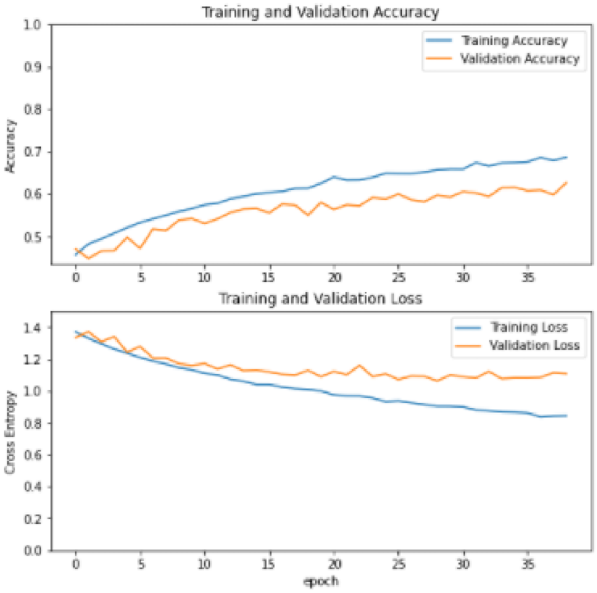
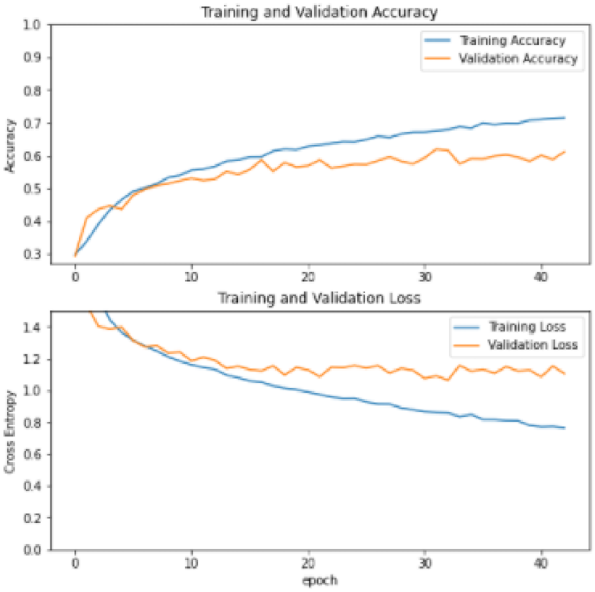
Optimization of age classification

For this classification, we decided to increase the number of hidden neurons for the fully connected layer.

Number of neurons	128	256	512
Validation Loss	1.066	**1.047**	1.080
Validation accuracy	0.6040	0.6245	**0.6340**

From the table, we decided to use 256 as the number of hidden neurons, as it results in the lowest validation loss.

Experiment and Results

Basic Architecture from Scratch	Optimized Architecture from Scratch
Gender Classification	
Val_loss: 0.313 Val_acc: 0.871	Val_loss: 0.242 Val_acc: **0.903**
	
Age Classification	
val_loss: 1.06 val_acc: 0.598	val_loss: 1.06 val_acc: **0.616**
	

Discussion

We will now evaluate the Inception-V3 model and our own proposed model performance against the performance of the model proposed in [10].

Gender Classification		**Age Group Classification**	
Model	**Validation Accuracy**	**Model**	**Validation Accuracy**
Levi-Hassner Model [10]	86.8	Levi-Hassner Model [10]	50.7
Inception-V3	86.9	Inception-V3	56.0
Modified Inception-V3	**92.9**	Modified Inception-V3	**63.6**
Proposed Model	87.1	Proposed Model	59.8
Tuned Proposed Model	90.3	Tuned Proposed Model	61.6

It is important to note that we are doing binary classification hence the model only has to classify between 2 classes. Hence for Gender classification we are able to achieve an accuracy of 92% but for the accuracy for the face age estimation does not have a similar level of accuracy as there are more classes(age groups) and each class has a limited number of images. In order to improve the face age estimation, we need to increase the size of the dataset. A possible idea for analyzing images would be cropping the face out of the full image and training the model purely based on the facial features as male and females have different facial constructs.

It is interesting to note that the Inception-V3 performs better for both gender and age classification as compared to the model by Levi & Hassner. This might be due to the fact that the Inception-V3 model has a deeper architecture and performs better. It was also developed later than the model by Levi & Hassner.

Conclusion

In this project, we have explored popular CNN architectures and used one of them - Inception-V3 to compare the performance of it against the model proposed in [10], with and without modifications done to suit the Adience dataset. We have also tried to define our own CNN model and discover the right hyperparameters to achieve the best performance and compare it to the model in [10]. As this is the team’s first time in learning Neural Networks, there were a lot of hurdles to overcome and a huge portion of the time was spent in understanding the models and translating them in TensorFlow code. If given more time and resources, we would be able to explore other advanced architectures, like ResNet and VGG. We would also be able to find out how to tune hyperparameters effectively without a wastage in time running models iteratively. Overall, through this project, the team now has a deeper appreciation and understanding of the recent developments of CNN in the field of computer vision and has sparked greater passion in delving deeper into deep neural networks.

References

[1] E. Eidinger, R. Enbar, and T. Hassner, “Age and Gender Estimation of Unfiltered Faces,” *Face Recognition in the Wild*. [Online]. Available: https://talhassner.github.io/home/projects/Adience/Adience-data.html#agegender

[2] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In CVPR, 2015.

[4] M. Lin, Q. Chen, and S. Yan. Network in network. arXiv:1312.4400, 2013.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In CVPR, 2016

[7] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. CoRR, abs/1602.07261, 2016

[8] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In CVPR, 2017.

[9] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[10] G. Levi and T. Hassner, “Age and gender classification using convolutional neural networks.” in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) workshops, 2015