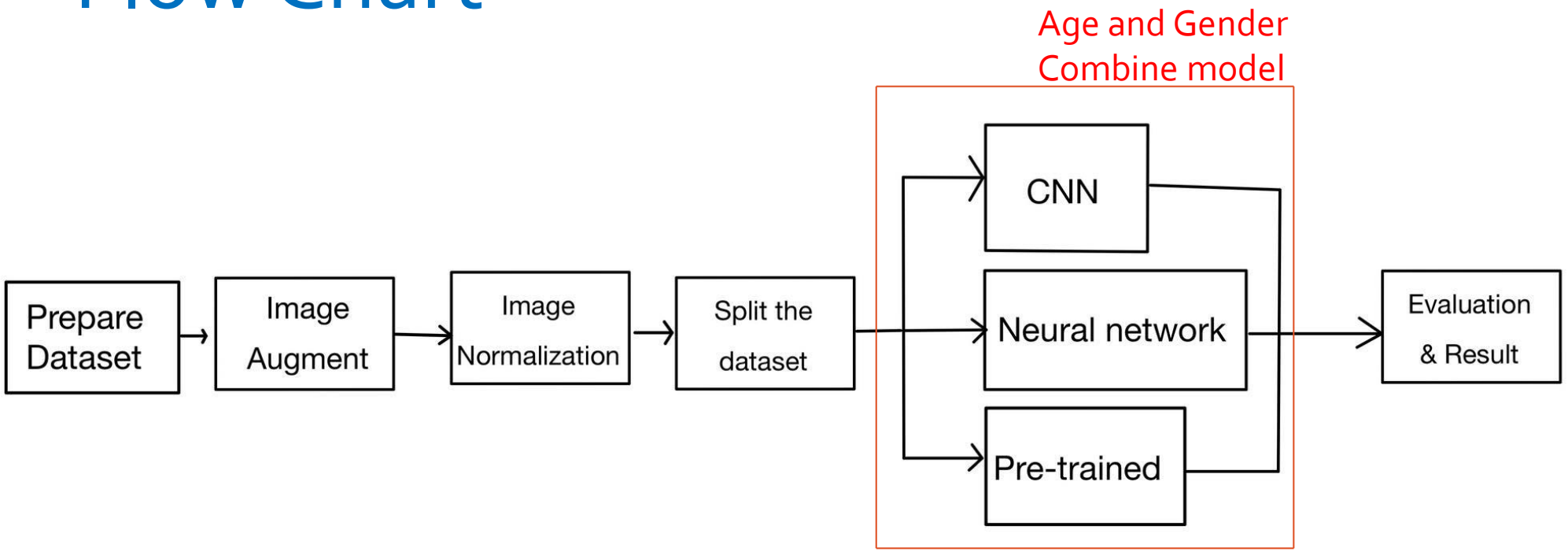
AGE AND GENDER PREDICTION

Jakkaphob Kongthanarith
6210554725

Flow Chart

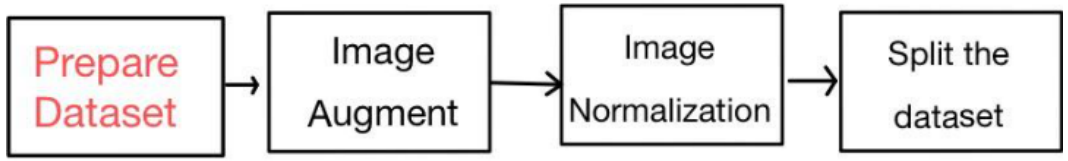

Age and Gender
Combine model
Prepare Dataset
Image Augment
Image Normalization
Split the dataset
CNN
Neural network
Pre-trained
Evaluation & Result

Data sets

k

UTK Face downloaded from Kaggle
Content: Face images
Age range: 0 to 116
Gender: Male and Female
Number of images : 23,708 images

https://www.kaggle.com/datasets/jangedoo/utkface-new



Number of files per gender: 25
Age range: 1-79

Age_Gender_Ethnicity_Filename.jpg.chip.jpg

25_0_0_20170104214616710.jpg
.chip

32_1_0_20170105002717213.jpg
.chip

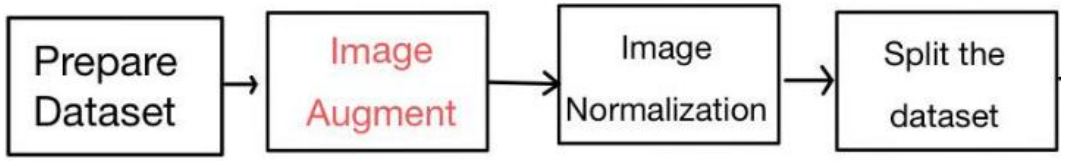
Age = 25
Gender = 0 -> Male

Age = 32
Gender = 1 -> Female

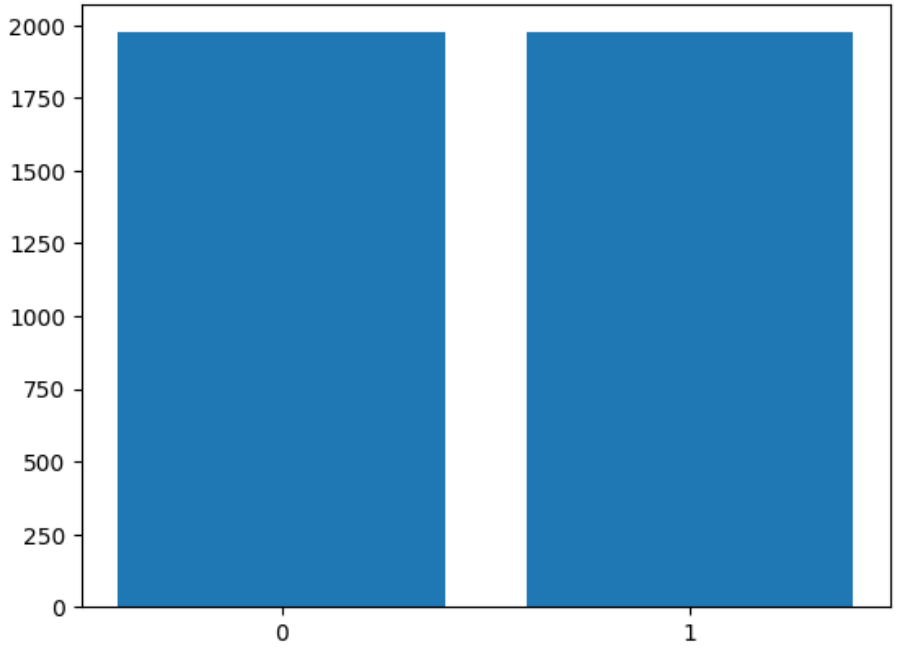
Source: https://susanqq.github.io/UTKFace/.

https://www.iconfinder.com/icons/4373210/kaggle_logo_logos_icon

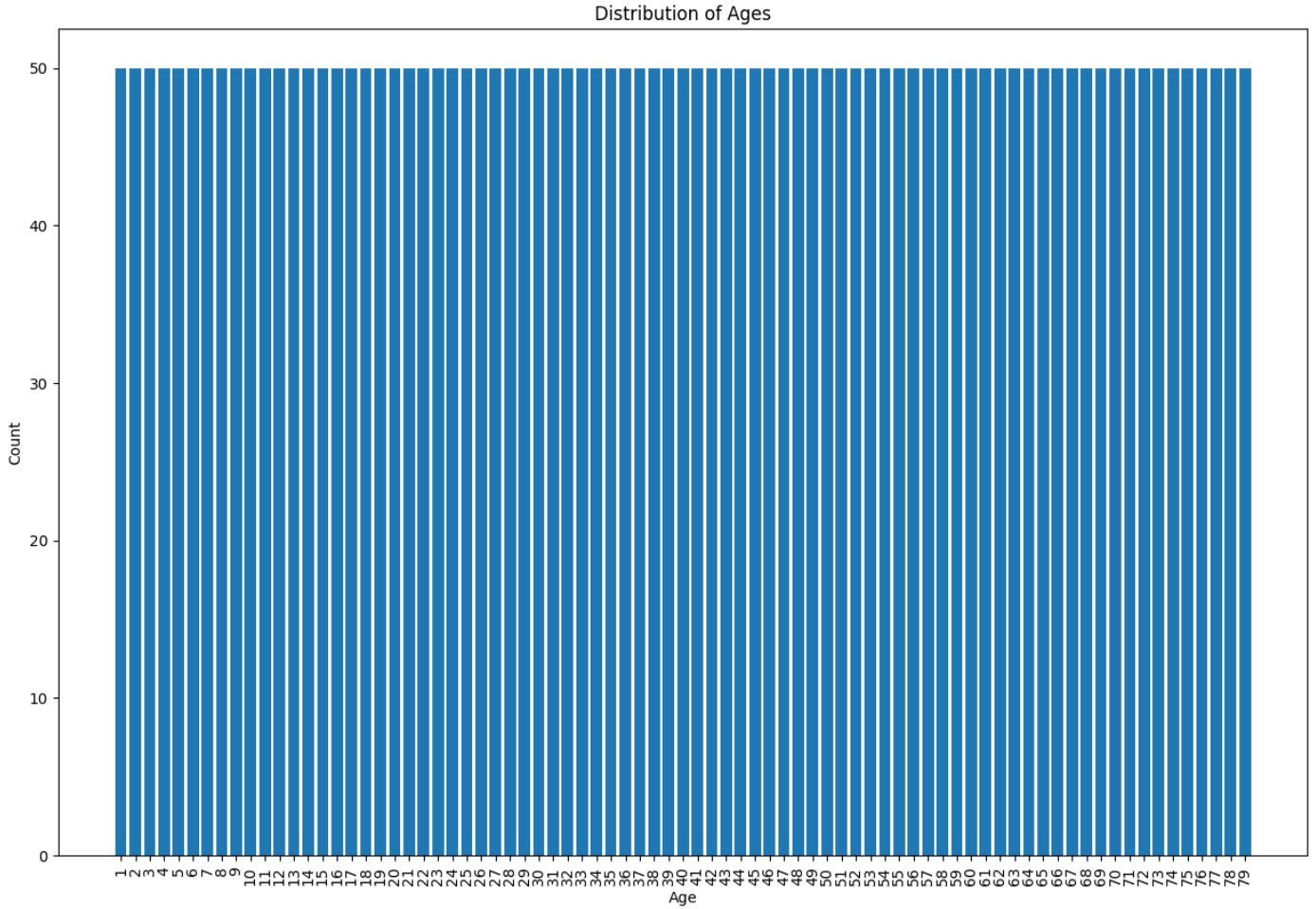
Image Augmentation



If the number of Images for each gender did not reach the maximum count (25) perform an augmentation
Augmentation method: Enhance contrast, brightness
Total Samples: 3950



0 = Male | 1 = Female



Min Value = 1 | Max value = 79

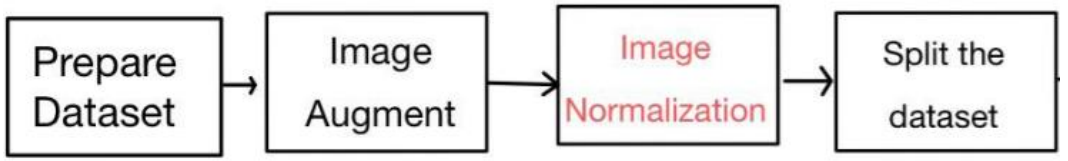
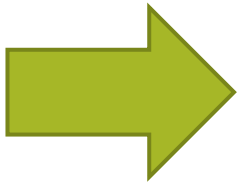


Image Normalization

Normalize the image to be used for Dataset's features by divide all of pixel values by 255

- Faster processing time
- better accuracy

```
[ 20.,  22.,  52.]
[ 12.,  14.,  44.]
[  8.,  10.,  40.]
```

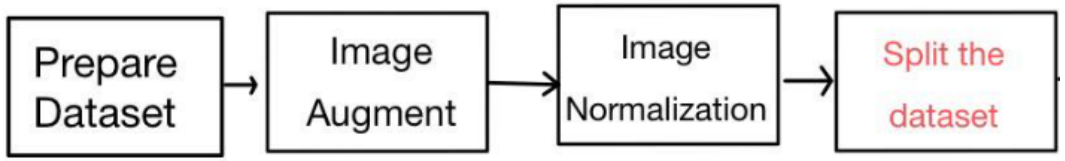


```
[0.07843138, 0.08627451, 0.20392157]
[0.04705882, 0.05490196, 0.17254902]
[0.03137255, 0.03921569, 0.15686275]
```

A part of pixel values before Normalization

A part of pixel values After Normalization

Dataset for Model training



Total images: 3950
Split into Training and Testing Dataset
- Train: 3160
- Test: 790
- Data type: numpy.ndarray

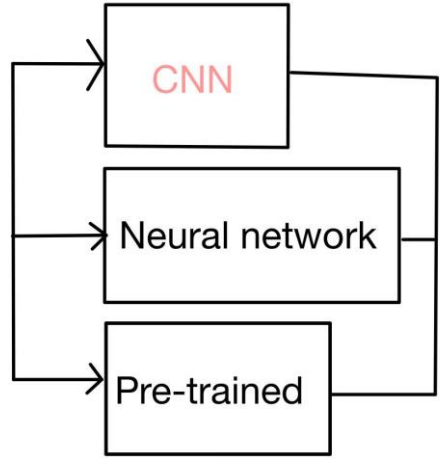
x train
(Image of an individual person)

Actual age: 21

y train
(Age of an individual person)

Actual age: 46

Age and Gender Model (CNN)



Age Model
Input layer : (224,224,1)
Output layer : ReLU

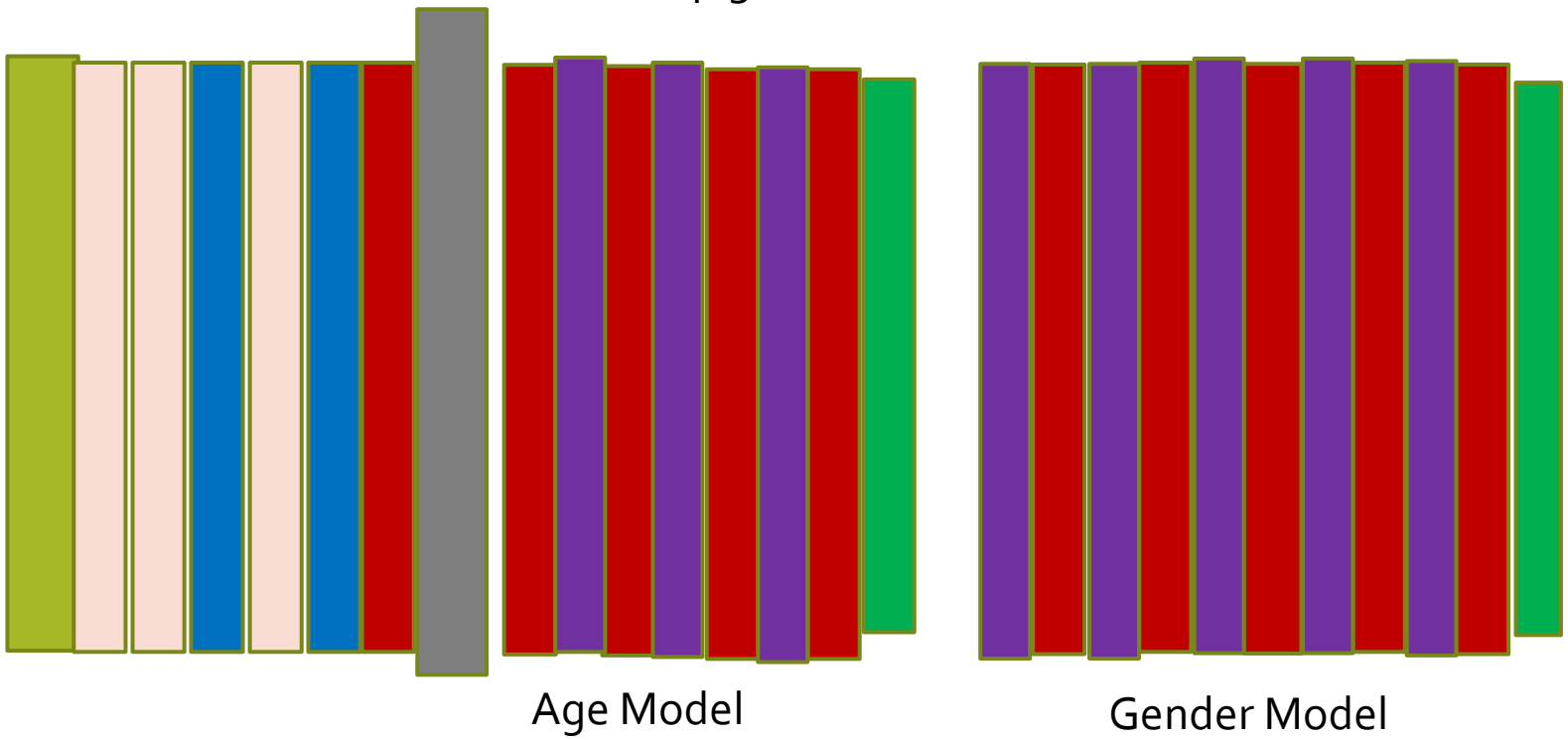
Epoch = 100
Batch size = 80

Gender Model
Input layer : (224,224,1)
Output layer : Sigmoid

Conv2D Layer :
32>64>128

Age Layer :
128>64>32>1

Gender Layer :
128>64>32>16>8>1



Age Model

Gender Model

= Input

= Convolutional2D

= Max Pooling

= Dropout

= Flatten

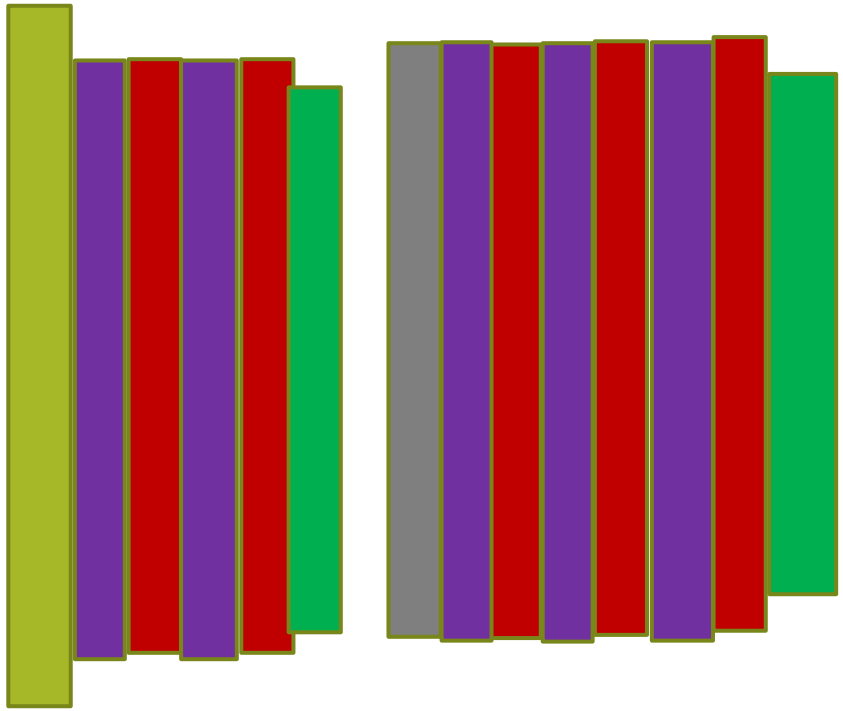
= Dense with ReLU

= Output

Age and Gender Model (Neural Network)

Age Model
Input layer : (224,224,1)
Output layer : Linear

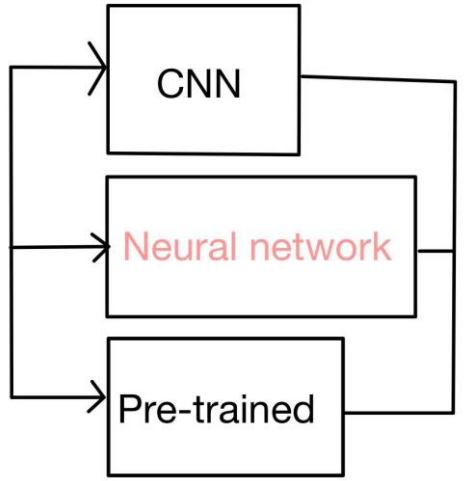
Gender Model
Input layer : (224,224,1)
Output layer : Sigmoid



Epoch = 60
Batch size = 32

Age Dense:
128>64>1

Gender Dense:
128>64>1



= Input

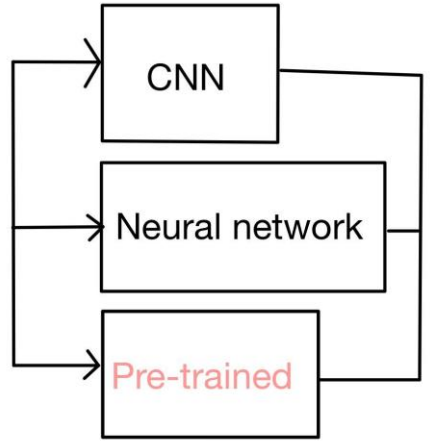
= Dropout

= Flatten

= Dense with ReLU

= Output

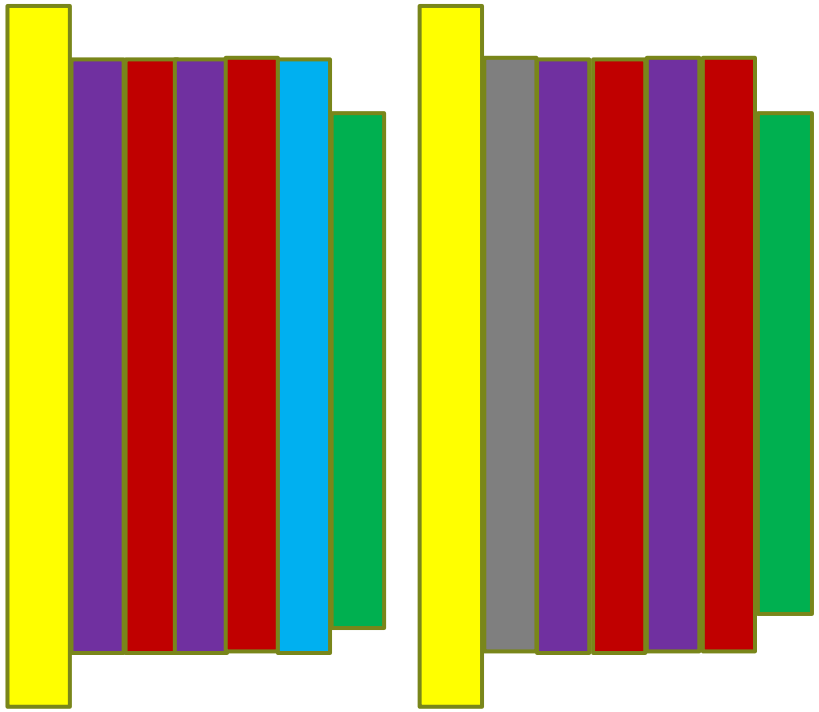
Age and Gender Model (Pretrained+Finetuned)



Age Model
Input layer : (224,224,3)
Output layer : ReLU

Gender Model
Input layer : (224,224,3)
Output layer : Sigmoid

Epoch = 100
Batch size = 128



Age Layer :
128>64>1

Gender Layer
: 256>128>1

= Pretrained model output

= Dropout

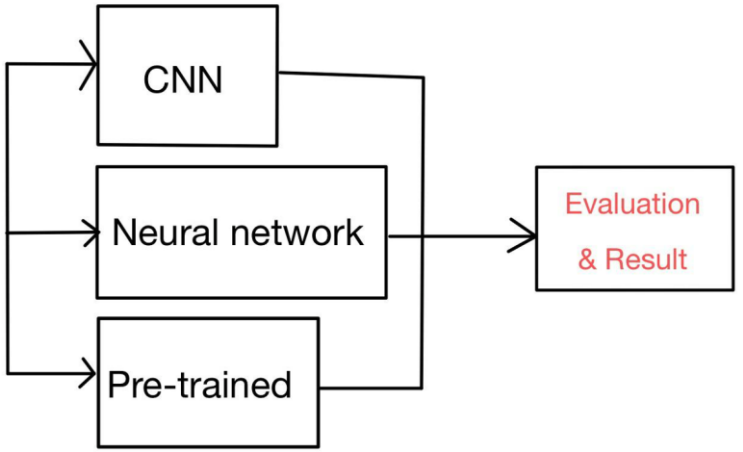
= Batch Normalization

= Flatten

= Dense with ReLU

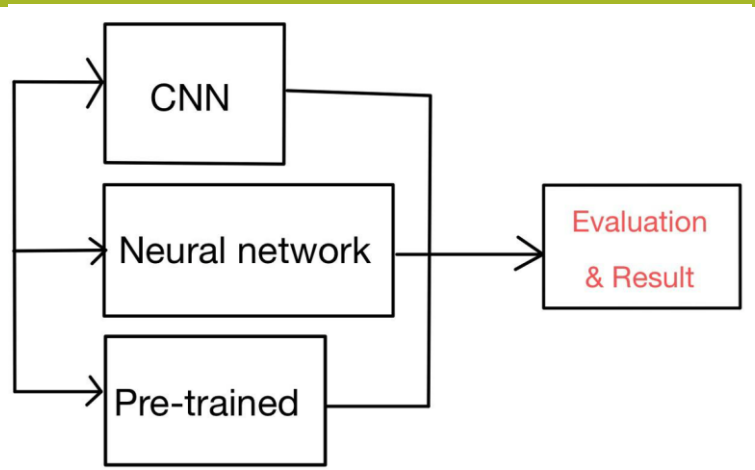
= Output

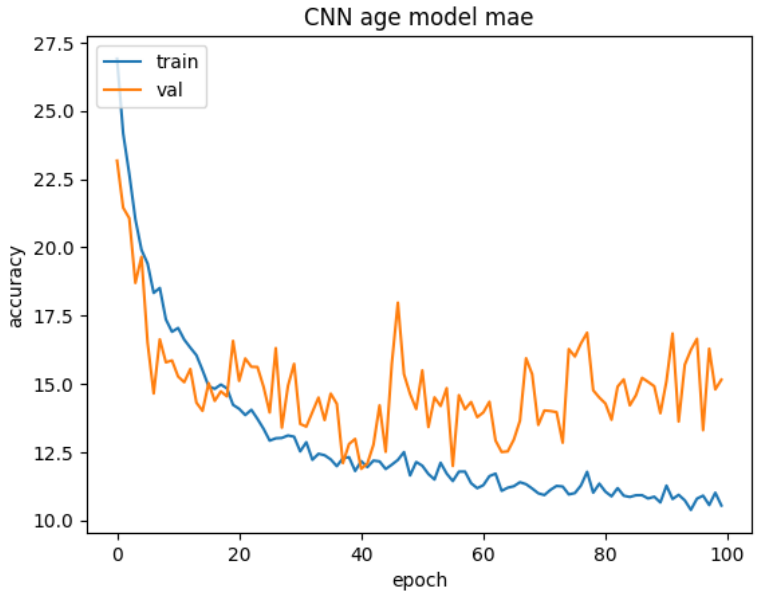
Age Model Evaluation

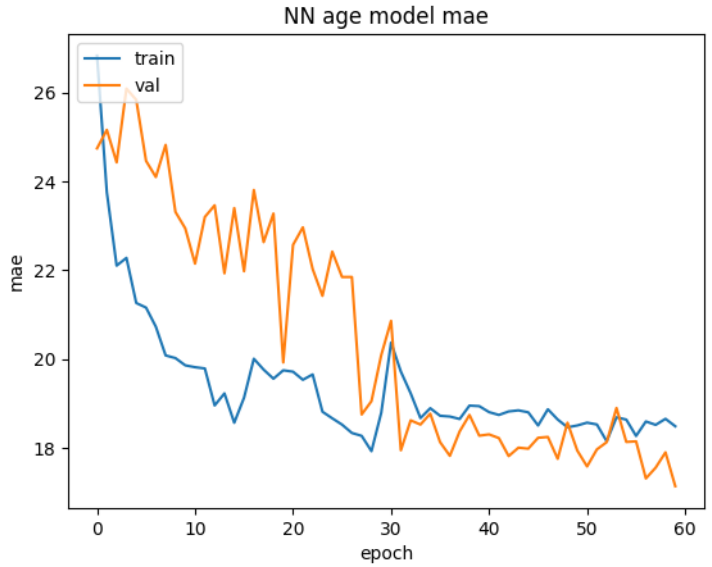


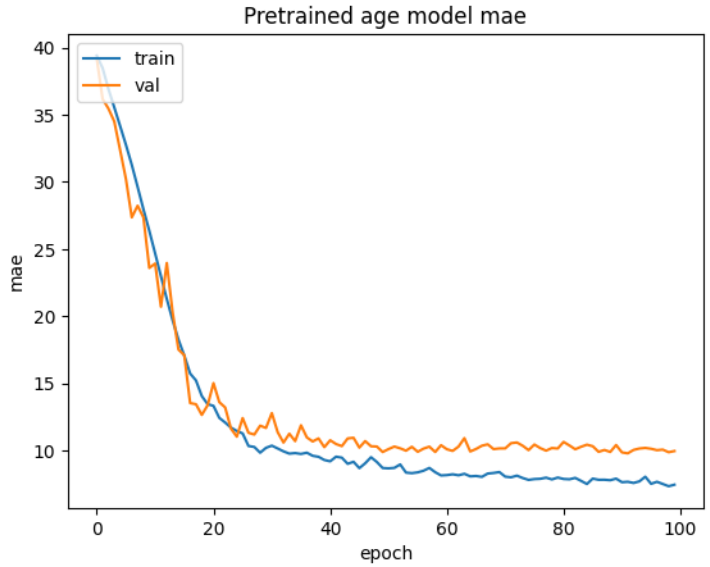
Model	Train MAE	Test MAE	Train Loss	Test Loss
CNN	10.5386	15.1492	197.7404	357.4378
Neural Network	18.4853	17.1355	484.3796	406.8231
VGG 16 Pretrained + fine-tuning model	7.4472	9.9541	95.2519	164.0147

Age Model Evaluation Graph





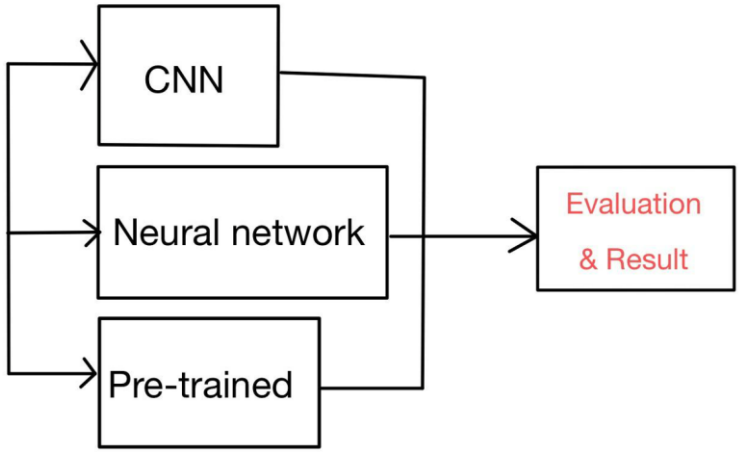




Percentage error of the model

```
def error_count(error):
  error1 = 0
  error2 = 0
  error3 = 0
  error4 = 0
  if error < 5:
    error1 = error1+1
  elif error <= 10:
    error2 = error2+1
  elif error <= 15:
    error3 = error3+1
  elif error > 15:
    error4 = error4+1
  return error1,error2,error3,error4

percent_dif =  (abs(predicted_age – actual_age))/ max_age) * 100
error_count(percent_dif)
```

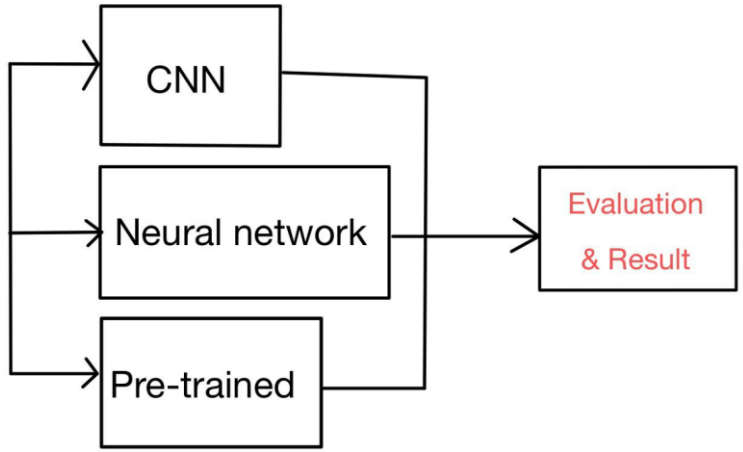


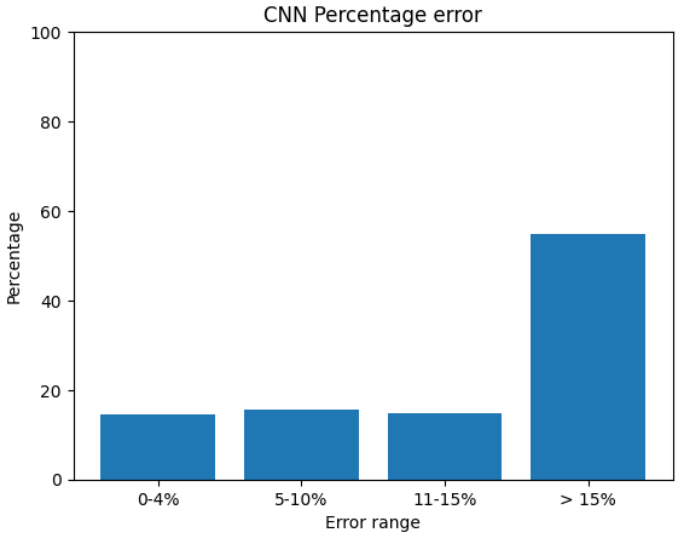
Real | Predict
76 | 47
percent difference: 32 %

Real | Predict
44 | 40
percent difference: 4 %

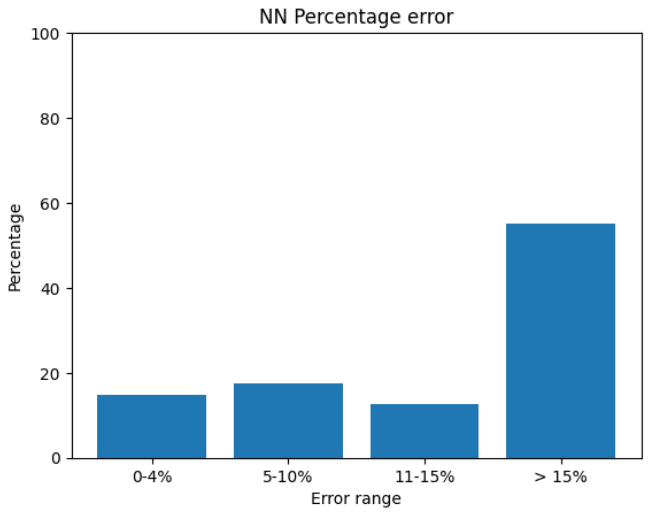
Distribution of percent error

Test size: 400 files Randomly selected from UTK Face Dataset

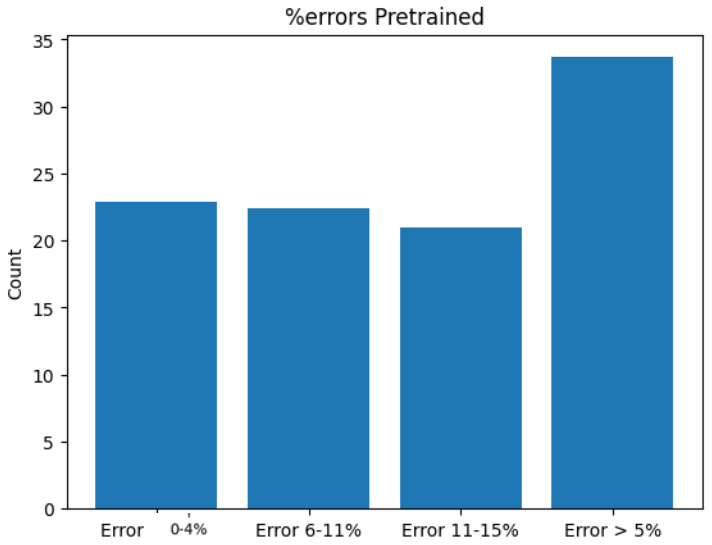




Error <5 %: 14.56 %
Error between 6- 10%: 15.70 %
Error between 11-15%: 14.94 %
Error > 15%: 54.81 %

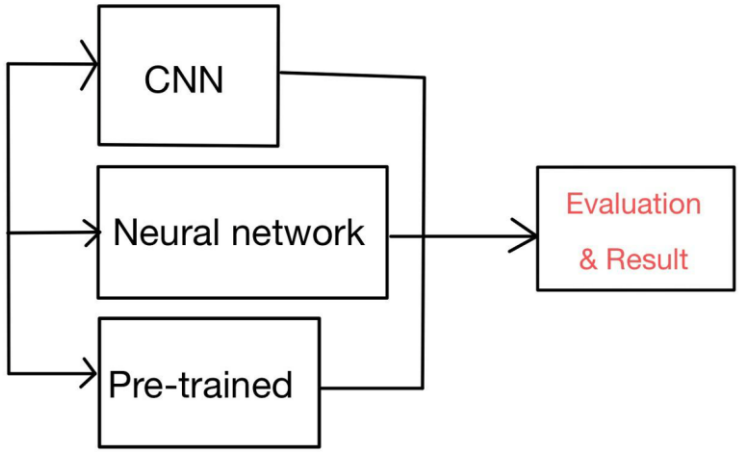


Error <5 %: 14.81
Error between 6- 10%: 17.47 %
Error between 11-15%: 12.53 %
Error > 15%: 55.19 %



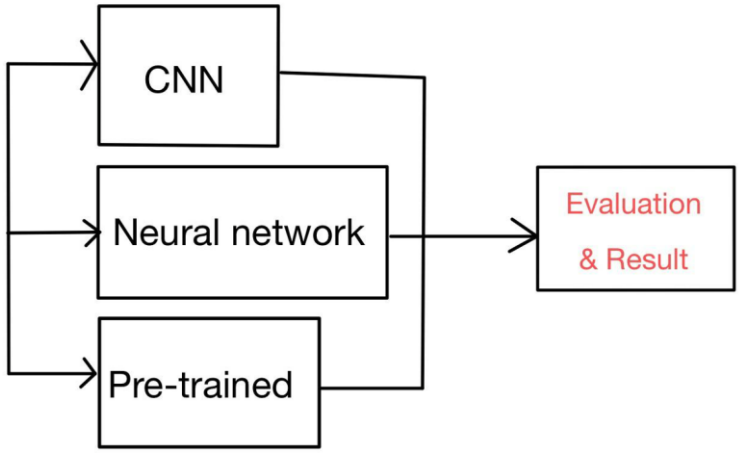
Error < 5 %: 22.91 %
Error between 6- 10%: 22.41 %
Error between 11-15%: 21.01 %
Error > 15%: 33.67 %

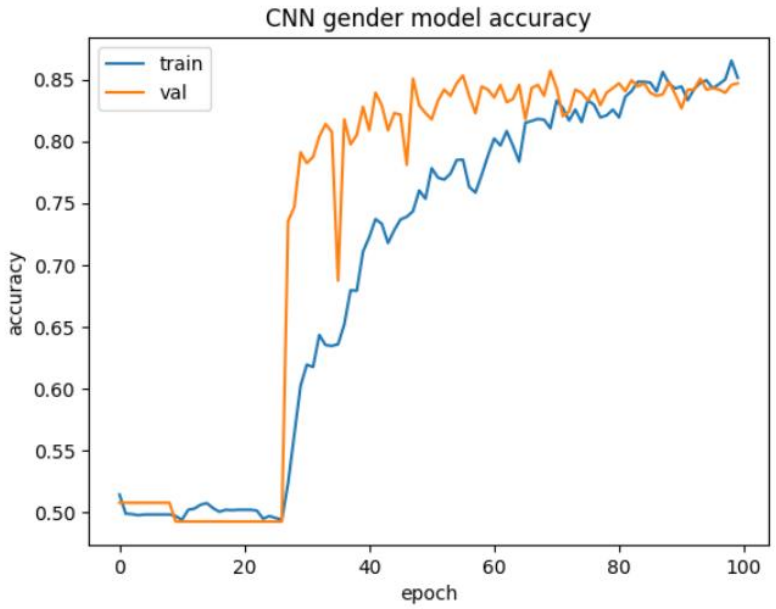
Gender Model Evaluation

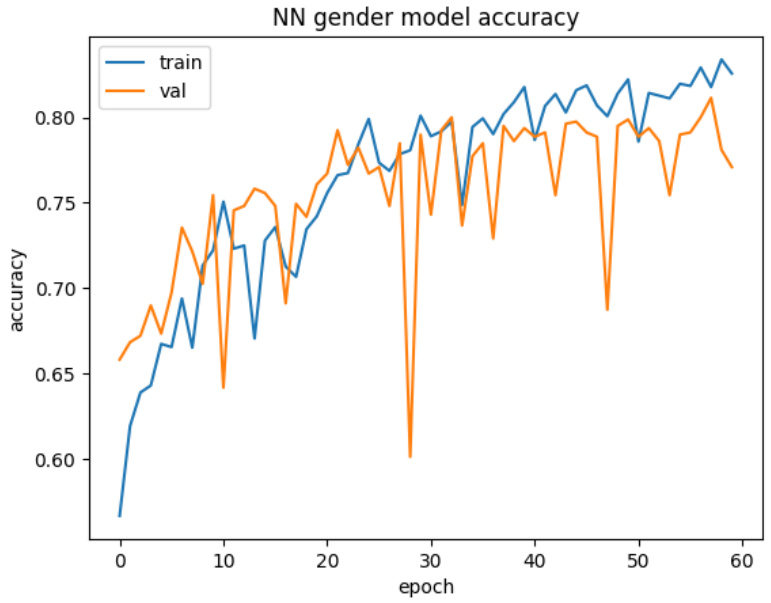


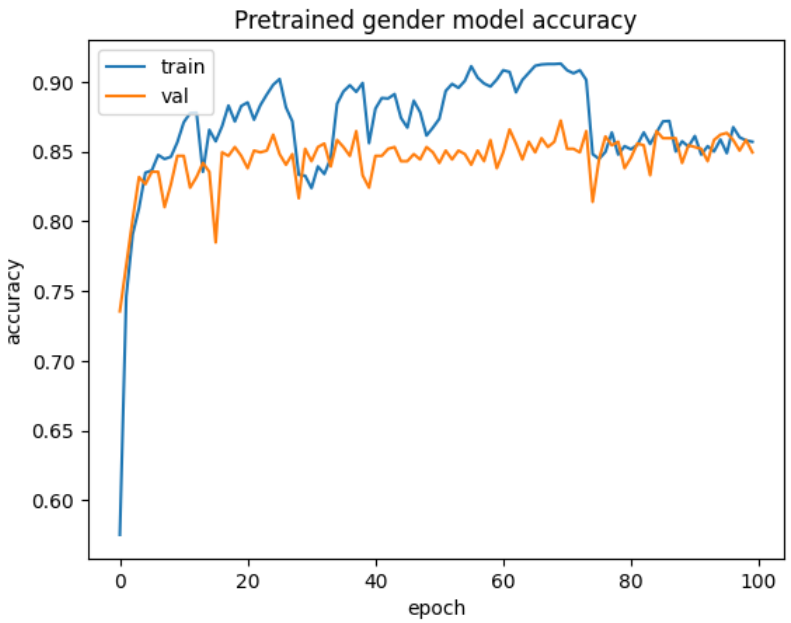
Model	Train Accuracy	Test Accuracy	Train Loss	Test Loss
CNN	0.8513	0.8468	0.3507	1.5800
Neural Network	0.8256	0.7709	0.3805	0.4442
VGG-16 Pre-trained model + fine-tuning model	0.8570	0.8494	0.2017	0.7013

Gender model Accuracy Graph

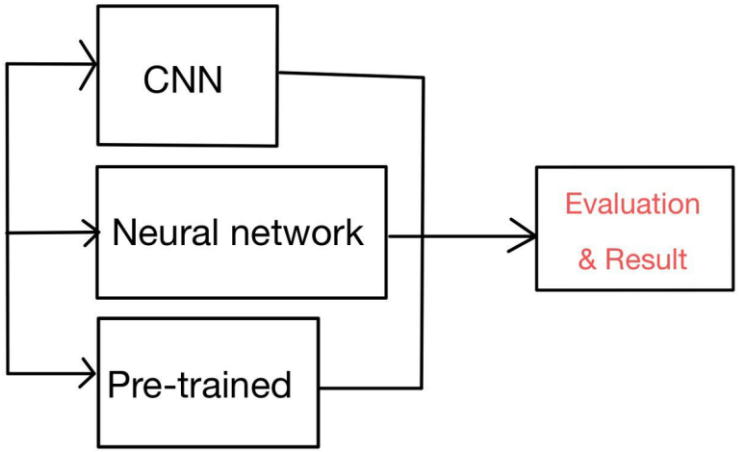


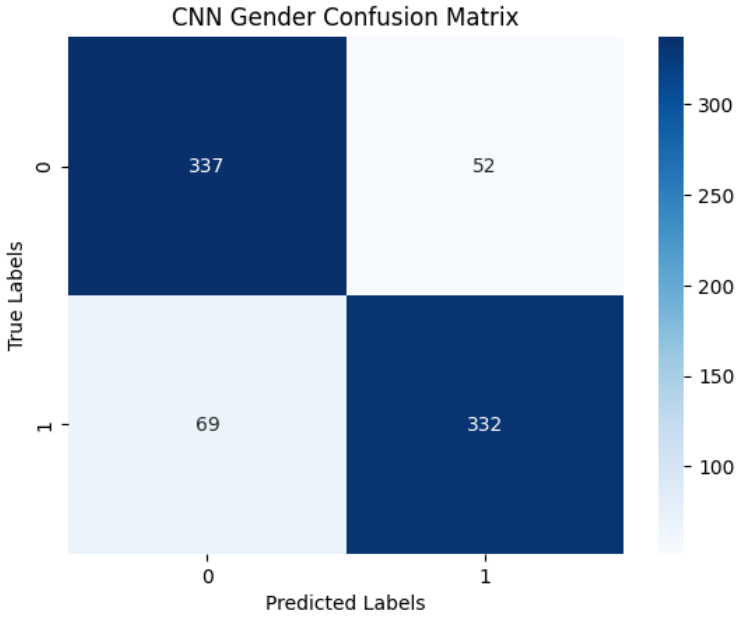


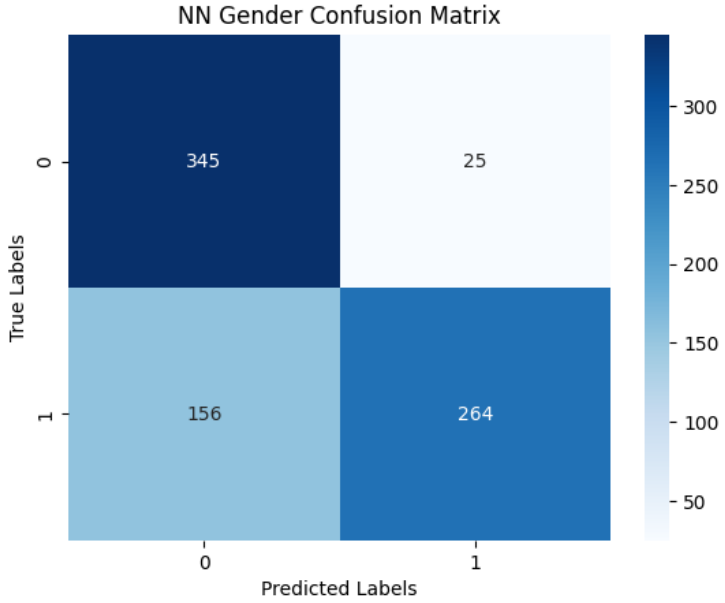


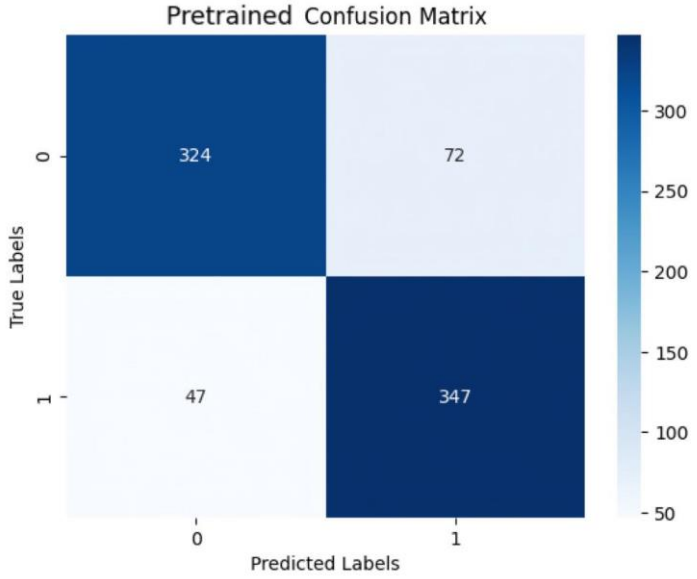


Gender Model Confusion Matrix

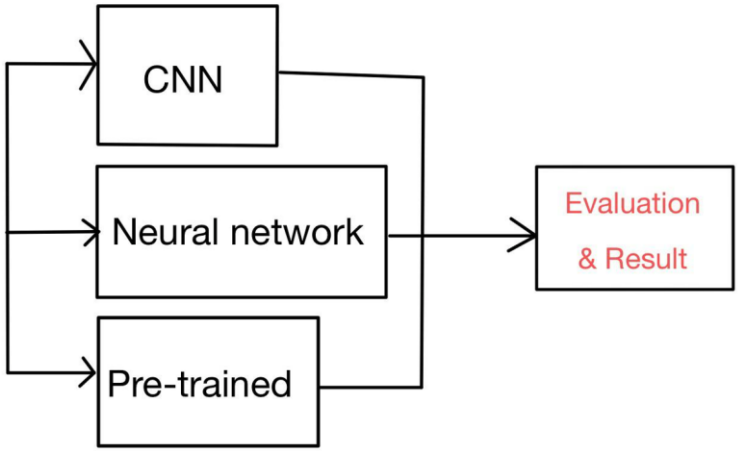








Gender Prediction



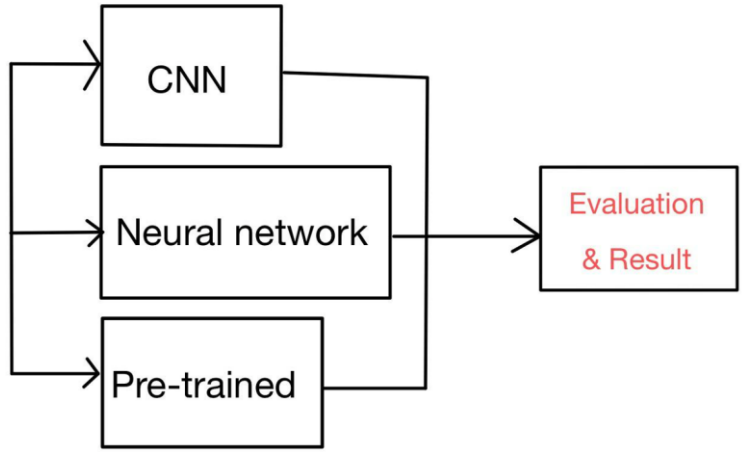

```
def get_gender(prob):
    if prob < 0.5:return "Male"
    else: return "Female"
```

- A model make a prediction value of the selected image
- Function 'get_gender()' will check that the value is higher or lower than 0.5
- If the predicted value lover than 0.5, return 'male' as an outcome
- Otherwise, return 'female' as an outcome

Real | Predict
Male | Male
Prediction Value 0.253399

Real | Predict
Female | Female
Prediction Value 0.999868

Prediction Result



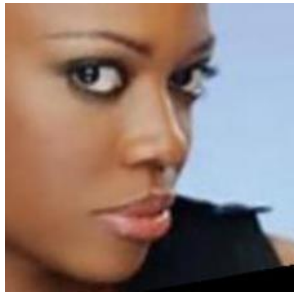
Real age | Predict age
61 | 56
Real gender | Predict gender
Male | Male

Real age | Predict age
25 | 32
Real gender | Predict gender
Female | Female

Real age | Predict age
12 | 1
Real gender | Predict gender
Male | Male

Real age | Predict age
12 | 31
Real gender | Predict gender
Female | Female

Real age | Predict age
4 | 12
Real gender | Predict gender
Male | Female



Real age | Predict age
26 | 18
Real gender | Predict gender
Female | Male