Capstone Project

Predicting Age-Range through Image Analysis

Zaini Chia - 29 Jan 2021

Problem Statement

- Deep Learning is the foundation on which many current breakthroughs are founded upon
- One of its biggest applications is in image analysis - for example, **Face ID** in the current iPhones, **Portrait photography**, and so on
- For this project, we will be looking into a simpler task: by using Deep Learning, we will train a model to i**dentify the age of a person** by analysing an image of their face

Problem Statement

Potential Applications

- Sorting through a large collection of Family photos
- Analysing customer demographic of businesses (cafes/restaurants, tourist avenues)
- Health and Fitness tracking

The Dataset

UTKFace

UTKFace

20,000+ images

	age	ethnicity	gender	img_name	pixels
0	1	2	0	20161219203650636.jpg.chip.jpg	129 128 128 126 127 130 133 135 139 142 145 14...
1	1	2	0	20161219222752047.jpg.chip.jpg	164 74 111 168 169 171 175 182 184 188 193 199...
2	1	2	0	20161219222832191.jpg.chip.jpg	67 70 71 70 69 67 70 79 90 103 116 132 145 155...
3	1	2	0	20161220144911423.jpg.chip.jpg	193 197 198 200 199 200 202 203 204 205 208 21...
4	1	2	0	20161220144914327.jpg.chip.jpg	202 205 209 210 209 209 210 211 212 214 218 21...
5	1	2	0	20161220144957407.jpg.chip.jpg	195 198 200 200 198 198 199 199 198 197 197 19...
6	1	2	0	20161220145040127.jpg.chip.jpg	208 216 217 219 222 223 222 221 220 220 221 22...
7	1	2	0	20170109191125532.jpg.chip.jpg	99 142 169 177 179 181 183 186 187 186 191 190...
8	1	2	0	20161219222749039.jpg.chip.jpg	127 127 133 140 143 148 152 157 160 165 172 17...
9	1	2	0	20170109191209991.jpg.chip.jpg	199 211 211 214 216 216 219 221 222 224 219 21...

Dataset - Issues

- 20 000 images, 120 MB
  (200 X 200 pixels each, RGB)
- Even the CSV file with descriptions is 199 MB!
- Note: Github only accepts files up to 100 MB in size
- Solution: host zip file on Dropbox

pixels
129 128 128 126 127 130 133 135 139 142 145 14...
164 74 111 168 169 171 175 182 184 188 193 199...
67 70 71 70 69 67 70 79 90 103 116 132 145 155...
193 197 198 200 199 200 202 203 204 205 208 21...

```
dataset_url = "https://www.dropbox.com/s/bdjm481gidafusd/UTKFace.tar?dl=1"
```

```
data_dir = tf.keras.utils.get_file(origin=dataset_url,
                                   fname='UTKFace',
                                   untar=True)
data_dir = pathlib.Path(data_dir)
```

```
Downloading data from https://www.dropbox.com/s/bdjm481gidafusd/UTKFace.tar?dl=1
138215424/138209280 [==============================] - 1s 0us/step
```

Exploratory Data Analysis

Your model is only as good as the Data you feed it!

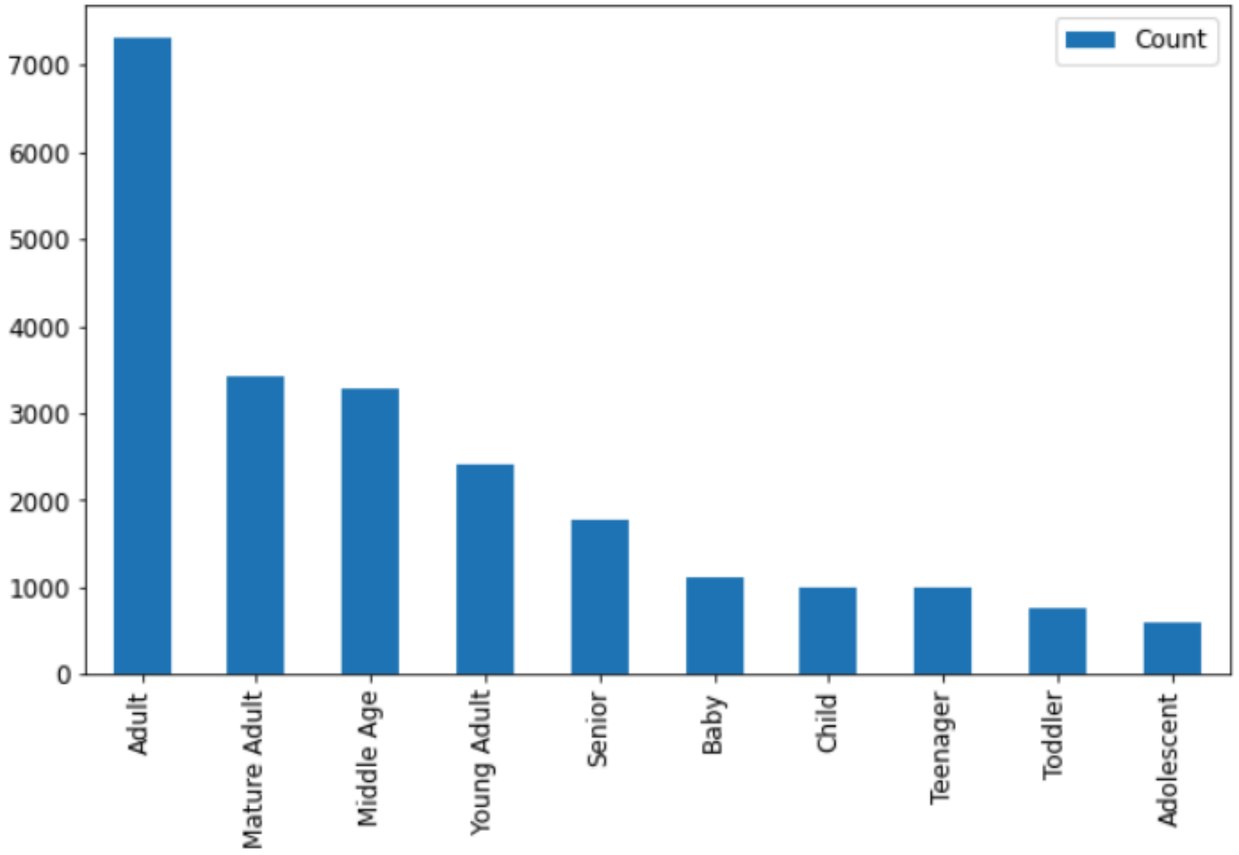
Change age to age-groups (Ordinal)

```
def age_classifier(age):
    if age == 1:
        age_group = 'Baby'
    elif age in range(2, 4):
        age_group = 'Toddler'
    elif age in range(4, 9):
        age_group = 'Child'
    elif age in range(9, 14):
        age_group = 'Adolescent'
    elif age in range(14, 19):
        age_group = 'Teenager'
```

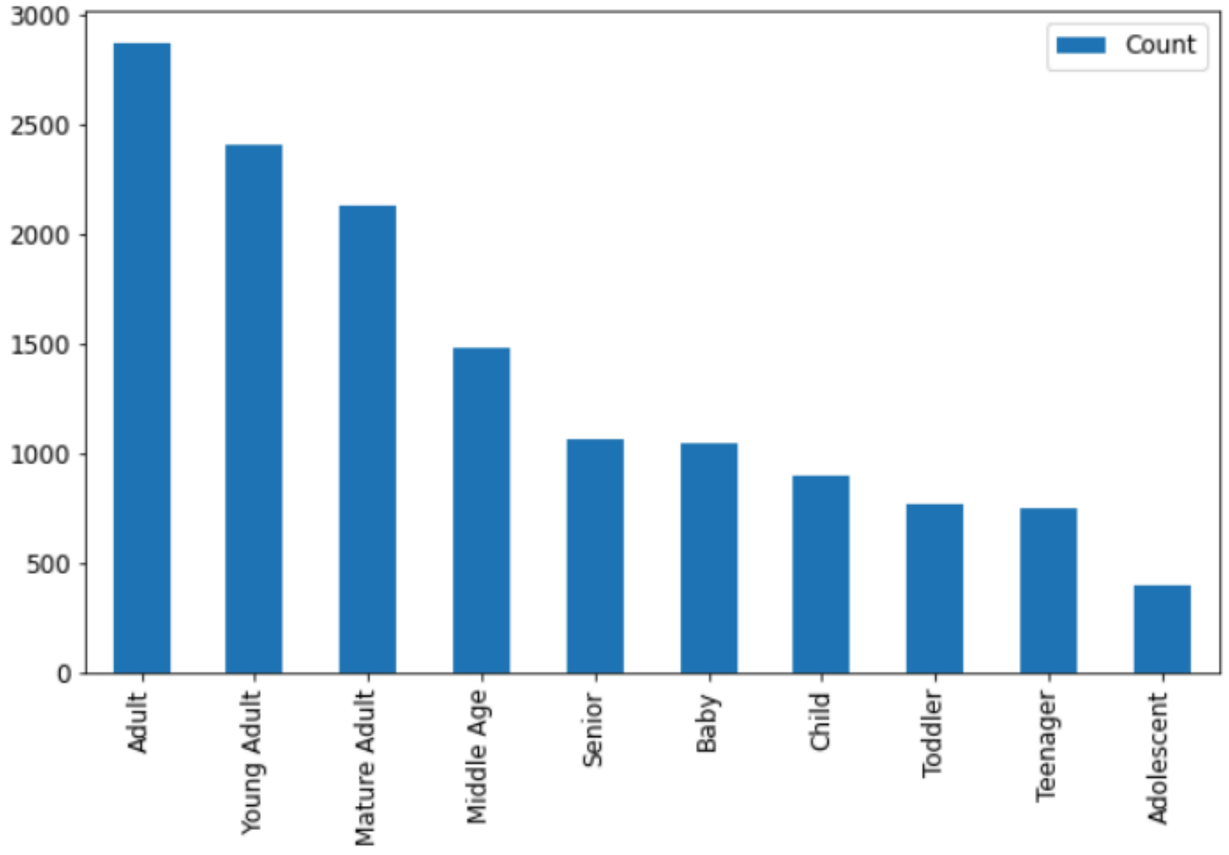
```
    elif age in range(19, 25):
        age_group = 'Young Adult'
    elif age in range(25, 35):
        age_group = 'Adult'
    elif age in range(35, 45):
        age_group = 'Mature Adult'
    elif age in range(45, 59):
        age_group = 'Middle Age'
    else:
        age_group = 'Senior'
    return age_group
```

Age-group Imbalance

Before

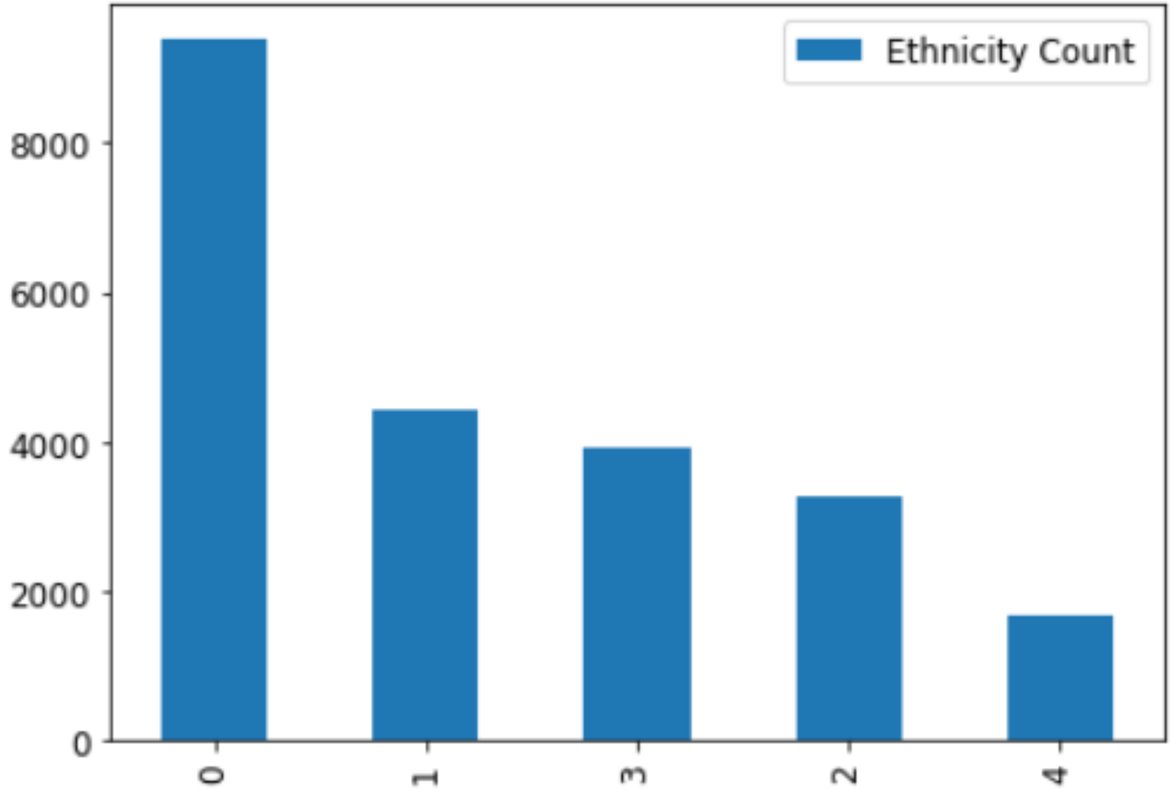


After

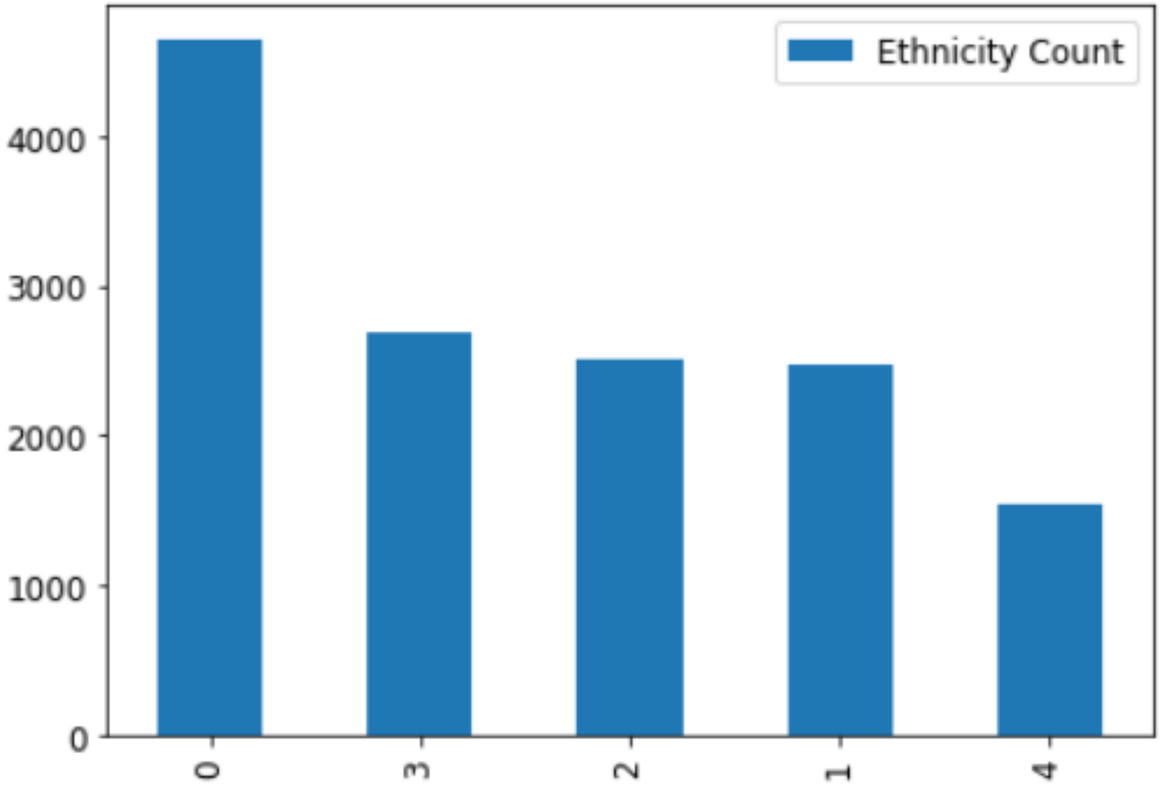


Ethnicity Imbalance

Before



After



0: White, 1: Black, 2: Asian, 3: Indian, 4: Others (Hispanic, Latino, Middle Eastern)

?

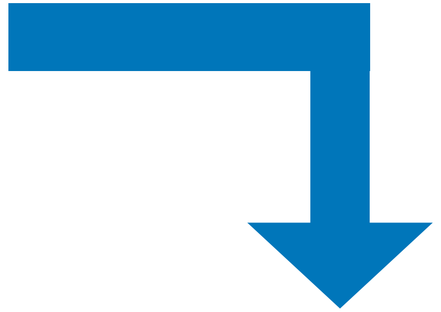
Final Dataframe

```
face_data.sample(8)
```

	age	ethnicity	gender	age group	Filename	age label
9550	40	0	1	Mature Adult	40_1_0_20170117185328025.jpg.chip.jpg	7
8221	35	0	0	Mature Adult	35_0_0_20170120220720314.jpg.chip.jpg	7
8855	37	0	0	Mature Adult	37_0_0_20170120221604091.jpg.chip.jpg	7
11278	53	3	0	Middle Age	53_0_3_20170117171834099.jpg.chip.jpg	8
2831	20	4	1	Young Adult	20_1_4_20170103230215713.jpg.chip.jpg	5
11778	58	1	0	Middle Age	58_0_1_20170117182453919.jpg.chip.jpg	8
12001	60	2	1	Senior	60_1_2_20170110151419801.jpg.chip.jpg	9
738	1	4	1	Baby	1_1_4_20161221201545377.jpg.chip.jpg	0

Setting up Tensorflow Dataset

	age	ethnicity	gender	age group	Filename	age label
35	1	2	0	Baby	1_0_2_20161219221751759.jpg.chip.jpg	0
1739	16	4	1	Teenager	16_1_4_20170103234142187.jpg.chip.jpg	4
3970	23	0	1	Young Adult	23_1_0_20170104021538021.jpg.chip.jpg	5
11751	58	2	0	Middle Age	58_0_2_20170112222056864.jpg.chip.jpg	8
5689	26	1	0	Adult	26_0_1_20170116024208336.jpg.chip.jpg	6
3121	21	4	1	Young Adult	21_1_4_20170103224431735.jpg.chip.jpg	5



```
def _parse_function(filename, label):
    image_string = tf.io.read_file(filename)
    image_decoded = tf.image.decode_jpeg(image_string,channels=3)
    image_resized = tf.image.resize(image_decoded, [224, 224])
    label = tf.one_hot(label, 10)
    return image_resized, label
```

```
train_dataset = tf.data.Dataset.from_tensor_slices((train_filenames, y_train))
train_dataset = train_dataset.map(_parse_function)
train_dataset = train_dataset.batch(32)

#we create the dataset based on the filenames in X_train
```

Young Adult

Adult

Senior

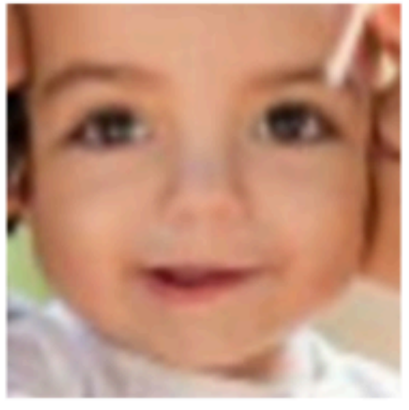
Adult

Young Adult

Young Adult

Middle Age

Baby



Mature Adult

Modelling setup

```
history = model_new.fit(train_dataset,
                        epochs=20,
                        batch_size = 32,
                        validation_data=validation_dataset,
                        callbacks=[early_stopping])
```

```
Epoch 1/20
 11/314 [>.............................] - ETA: 19:53 - loss: 6.4083 - accuracy: 0.1077
```

Erm.....

Solution: Google Colab

~ 40 X faster!

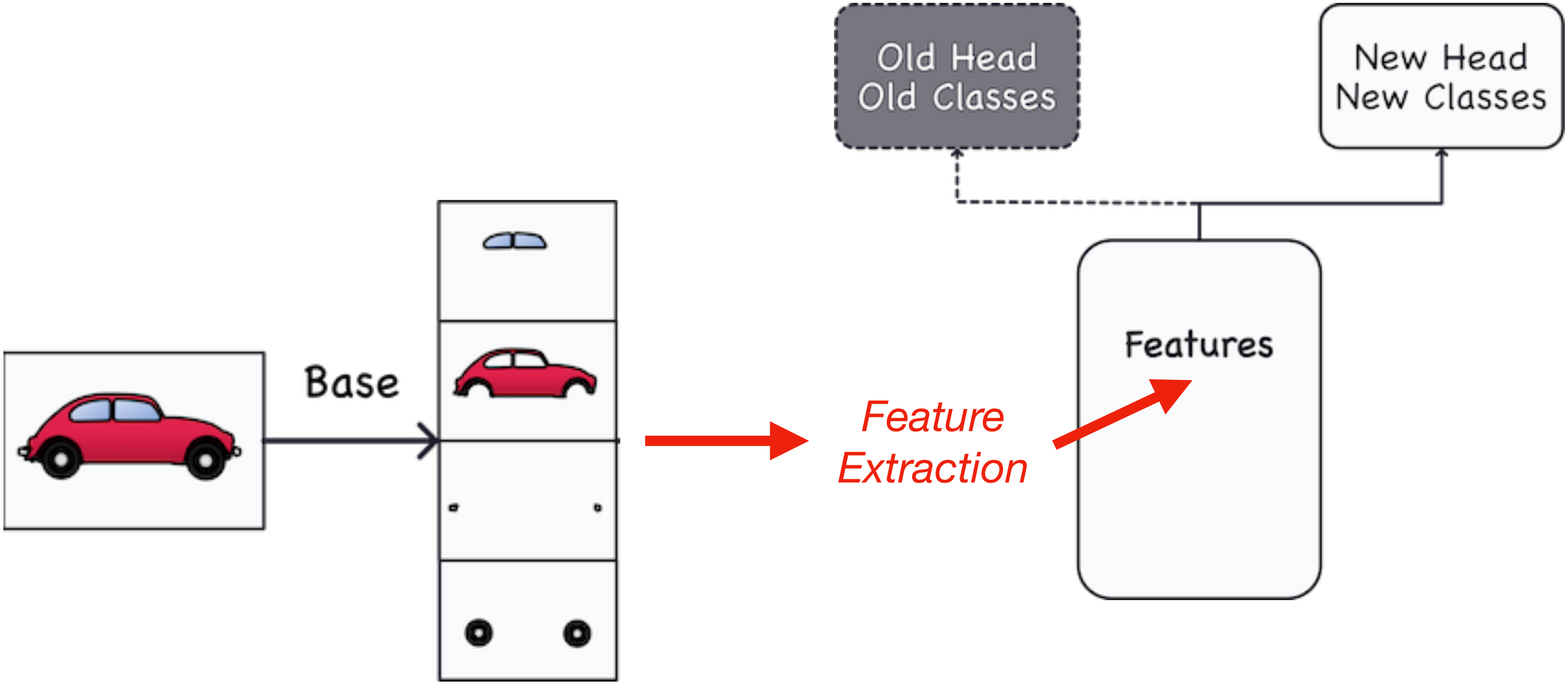
```
Epoch 1/20
314/314 [==============================] - 41s 105ms/step - loss: 2.9224 - accuracy: 0.1634
Epoch 2/20
314/314 [==============================] - 33s 105ms/step - loss: 2.1702 - accuracy: 0.1877
Epoch 3/20
314/314 [==============================] - 33s 106ms/step - loss: 2.0895 - accuracy: 0.2065
Epoch 4/20
314/314 [==============================] - 33s 105ms/step - loss: 1.9681 - accuracy: 0.2614
Epoch 5/20
314/314 [==============================] - 33s 105ms/step - loss: 1.8924 - accuracy: 0.2699
Epoch 6/20
314/314 [==============================] - 33s 106ms/step - loss: 1.8565 - accuracy: 0.2729
Epoch 7/20
314/314 [==============================] - 33s 105ms/step - loss: 1.8032 - accuracy: 0.2885
```

```
Epoch 1/20
314/314 [==============================] - 41s 105ms/step - loss: 2.9224 - accuracy: 0.1634
Epoch 2/20
314/314 [==============================] - 33s 105ms/step - loss: 2.1702 - accuracy: 0.1877
Epoch 3/20
314/314 [==============================] - 33s 106ms/step - loss: 2.0895 - accuracy: 0.2065
Epoch 4/20
314/314 [==============================] - 33s 105ms/step - loss: 1.9681 - accuracy: 0.2614
Epoch 5/20
314/314 [==============================] - 33s 105ms/step - loss: 1.8924 - accuracy: 0.2699
Epoch 6/20
314/314 [==============================] - 33s 106ms/step - loss: 1.8565 - accuracy: 0.2729
Epoch 7/20
314/314 [==============================] - 33s 105ms/step - loss: 1.8032 - accuracy: 0.2885
```

Some issues:

- Data import/export?
- Latency during editing

Convolutional Neural Network: (In 4 slides!)

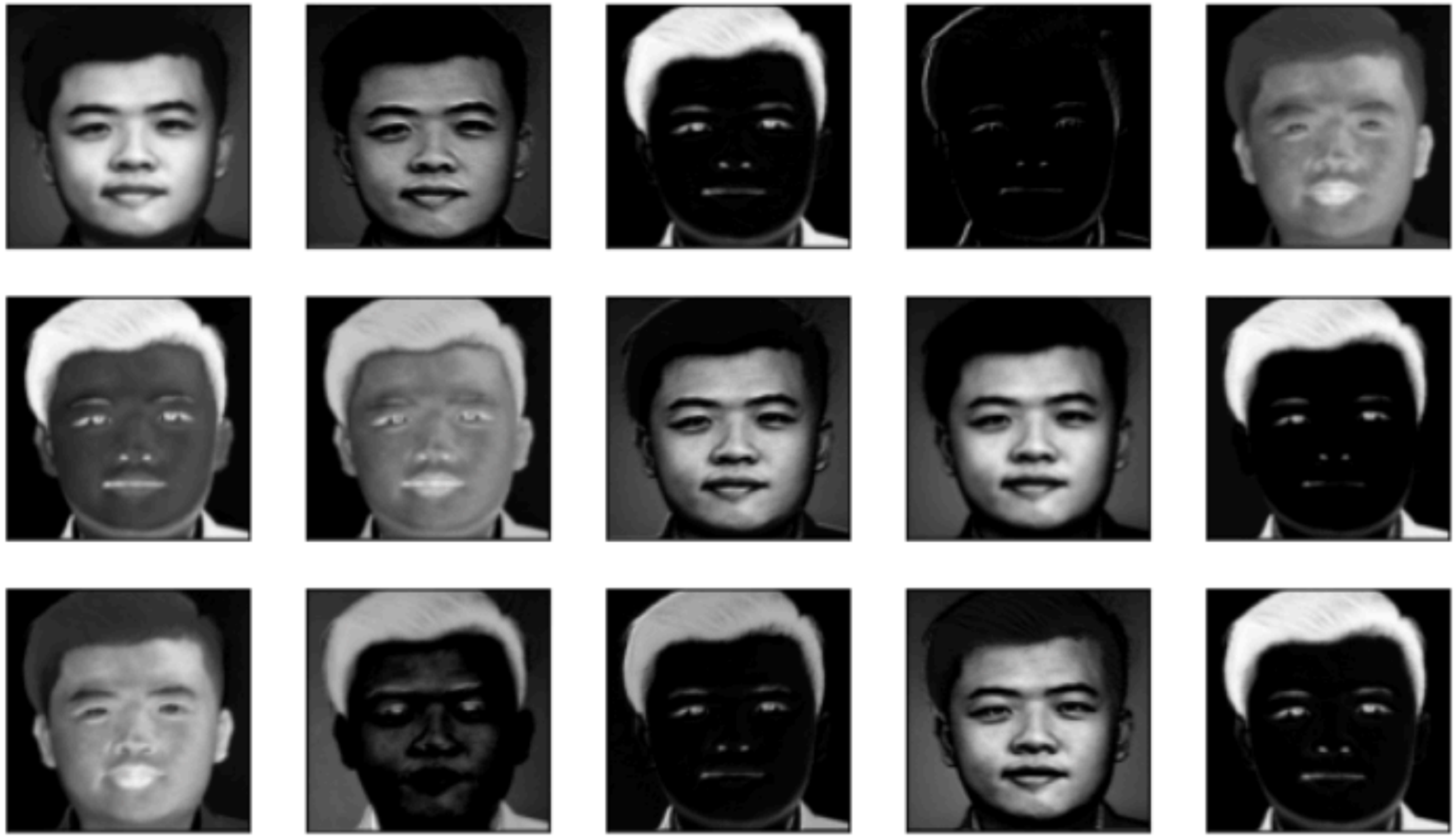

Old Head
Old Classes
New Head
New Classes
Features
Base
Feature
Extraction

First Convolutional Block

Texture

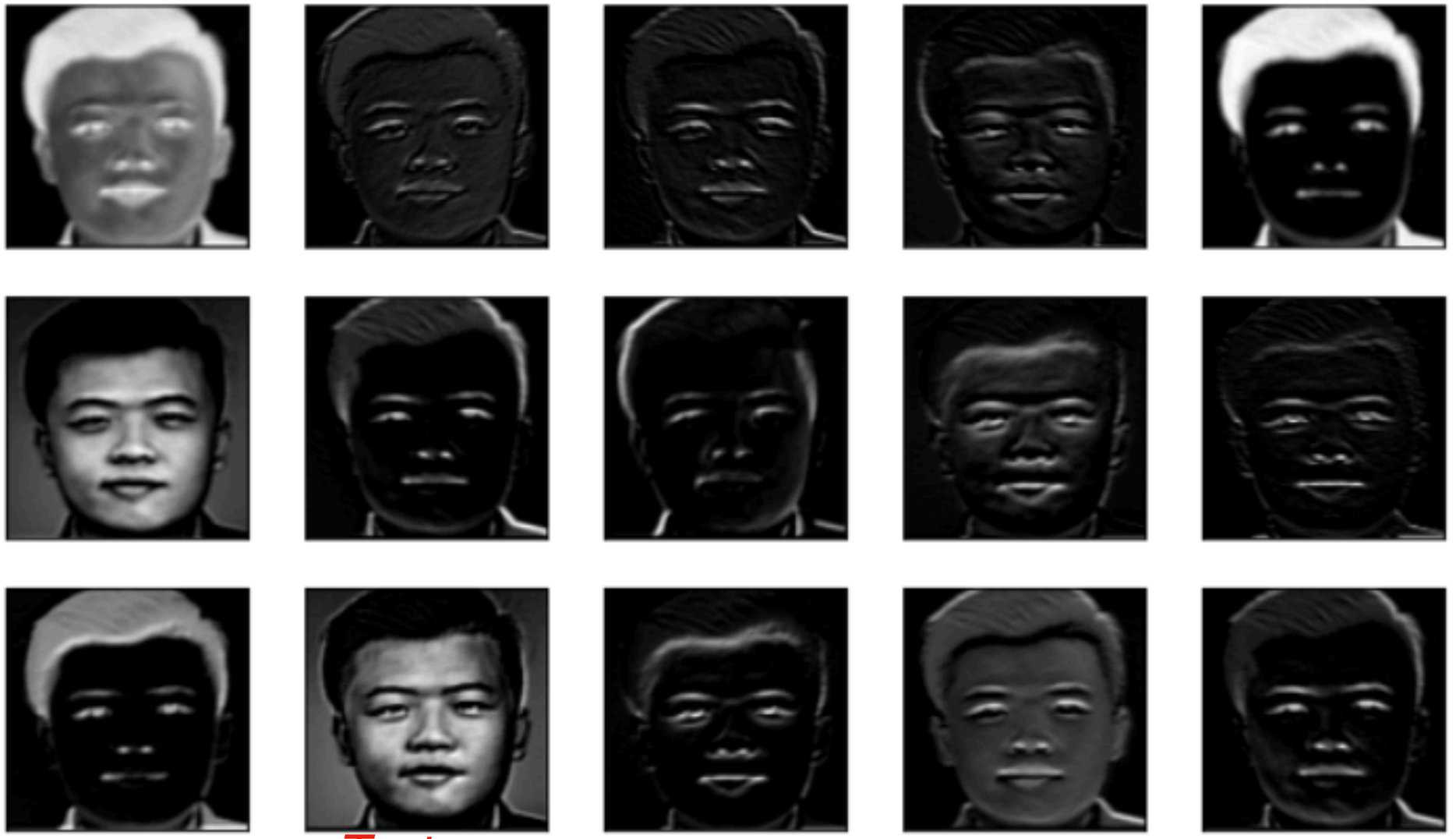
Outline

Contrast



Facial Landmarks

Second Convolutional Block

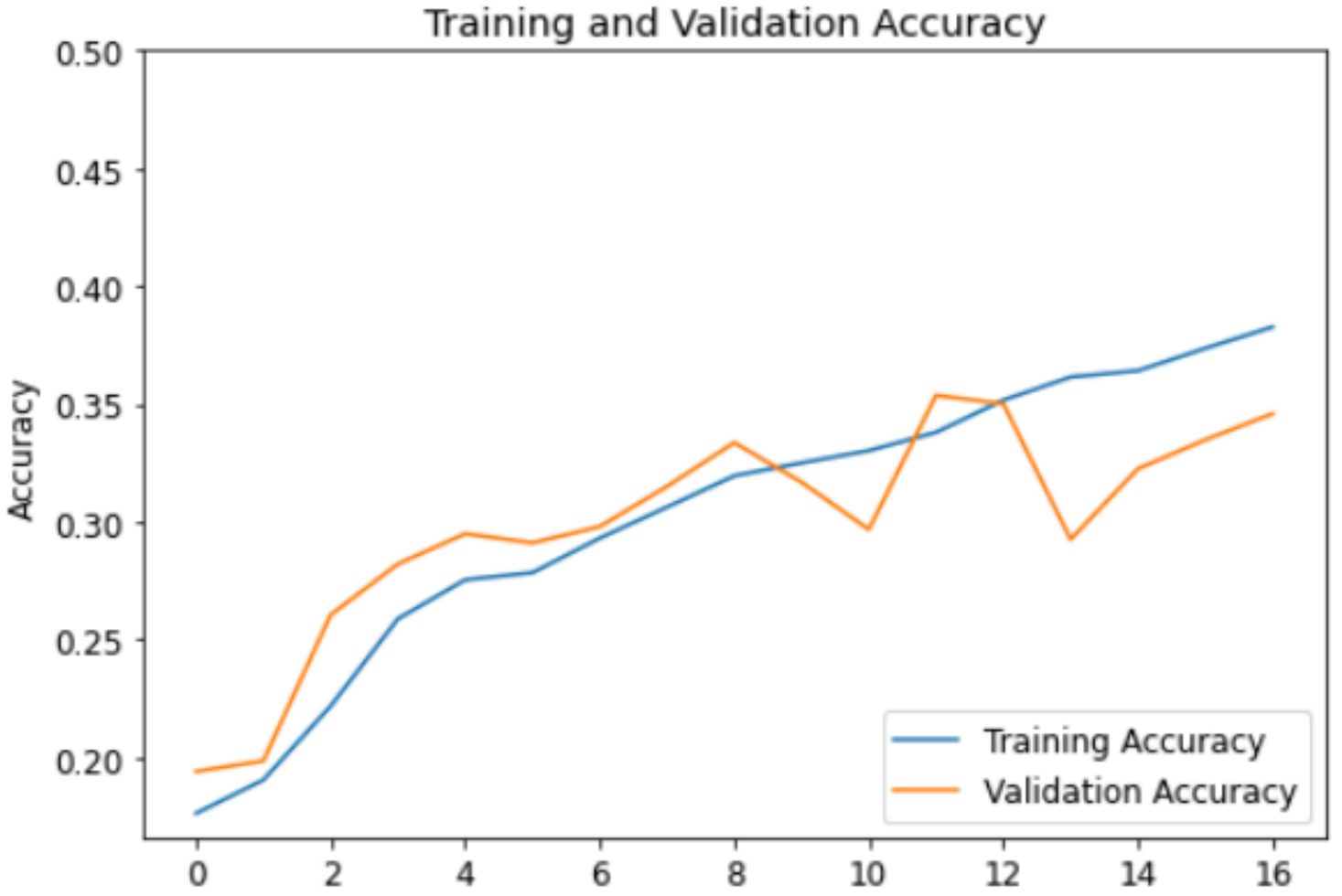


Third Convolutional Block

First Model:

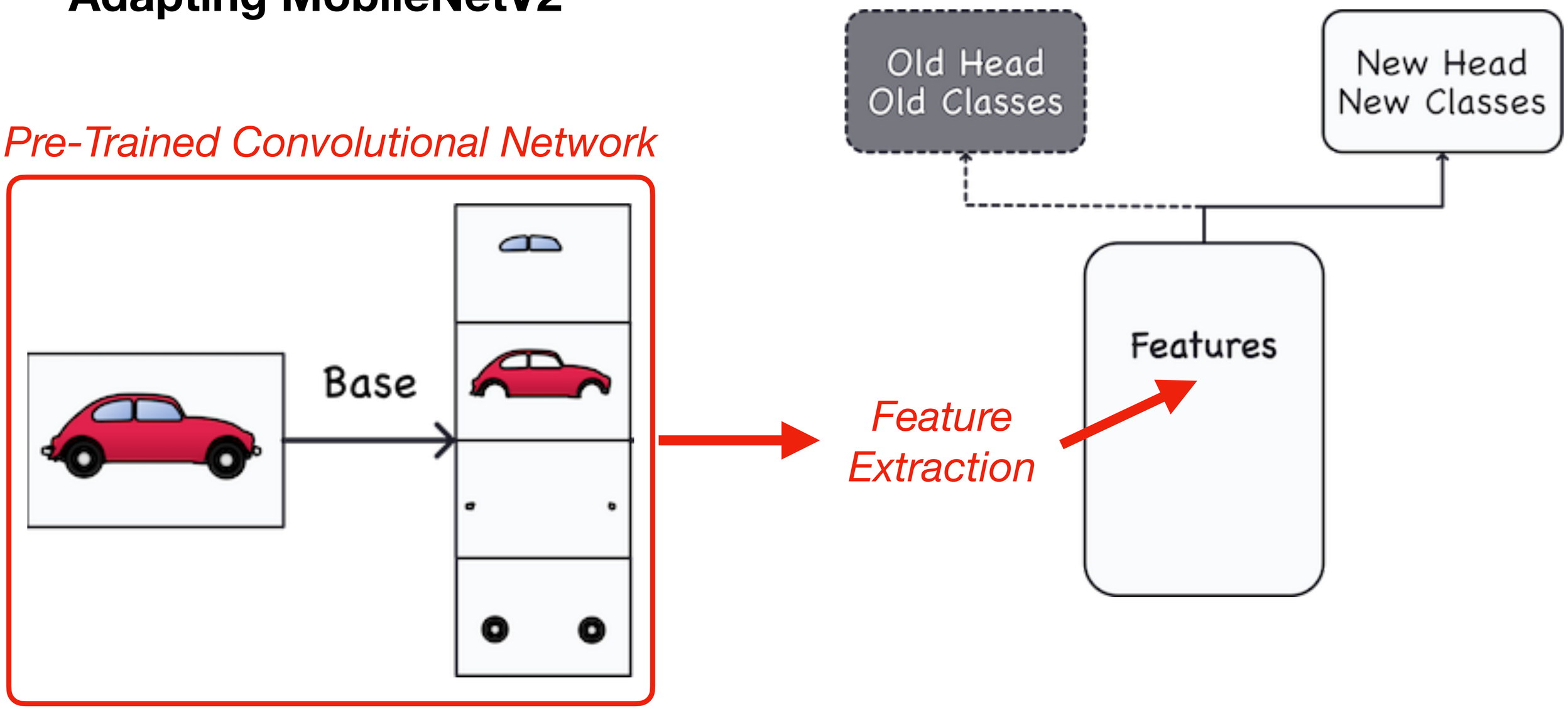
Self-Built CNN

- 15 layers, **1 million trainable parameters**
- 17 epochs before early stopping
- Accuracy score: 39%



Second Model: Transfer Learning

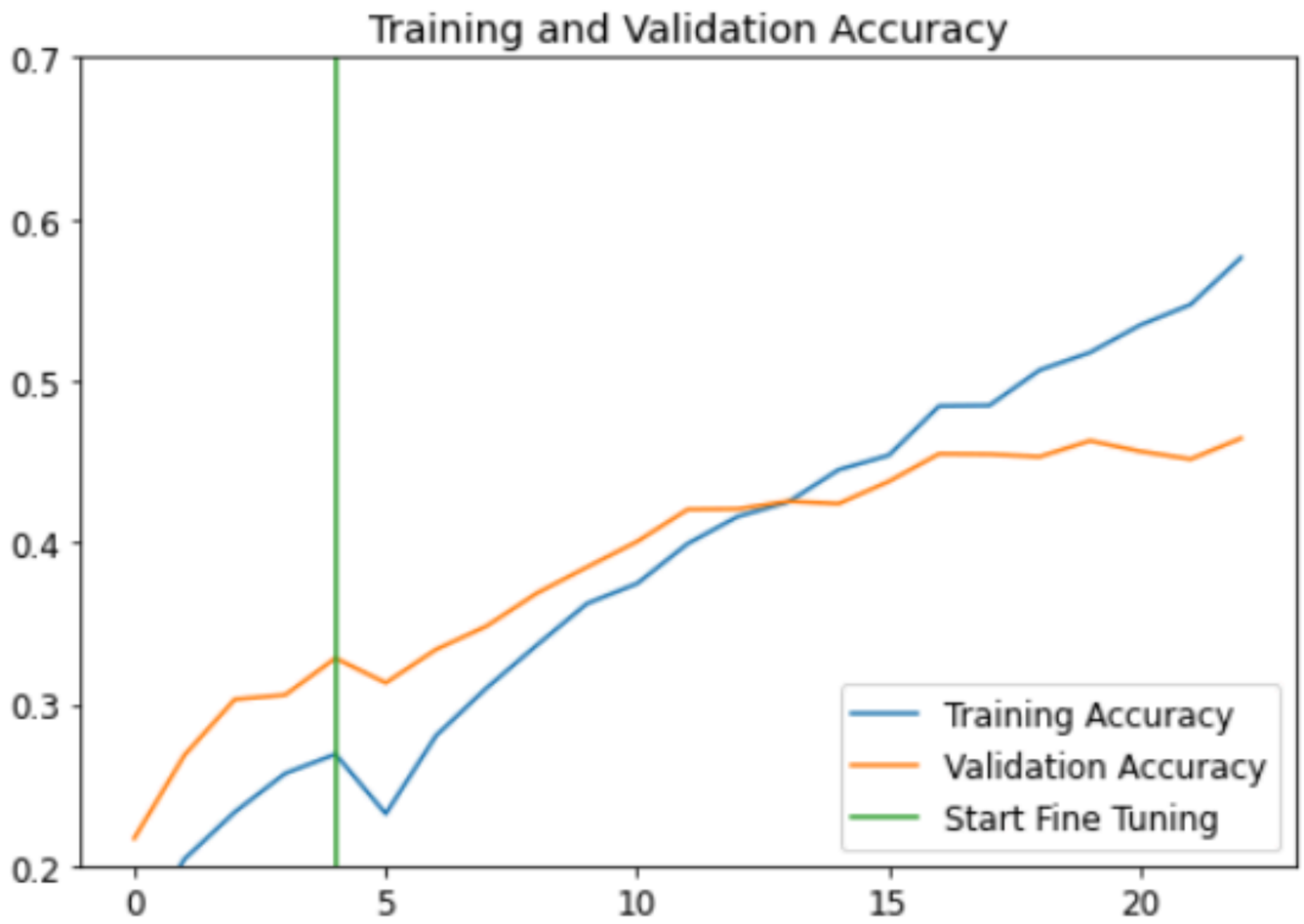
Adapting MobileNetV2



Second Model: Transfer Learning

Adapting MobileNetV2

- Initial Training
  (Convolutional Layers Frozen)
- Additional Training
  (Unfreeze 35 out of 155 layers)
  (**2 million trainable parameters**)
- Final Accuracy Score: 46%
- **Fast**: Designed to be deployed on Mobile devices



Third Model: Transfer Learning

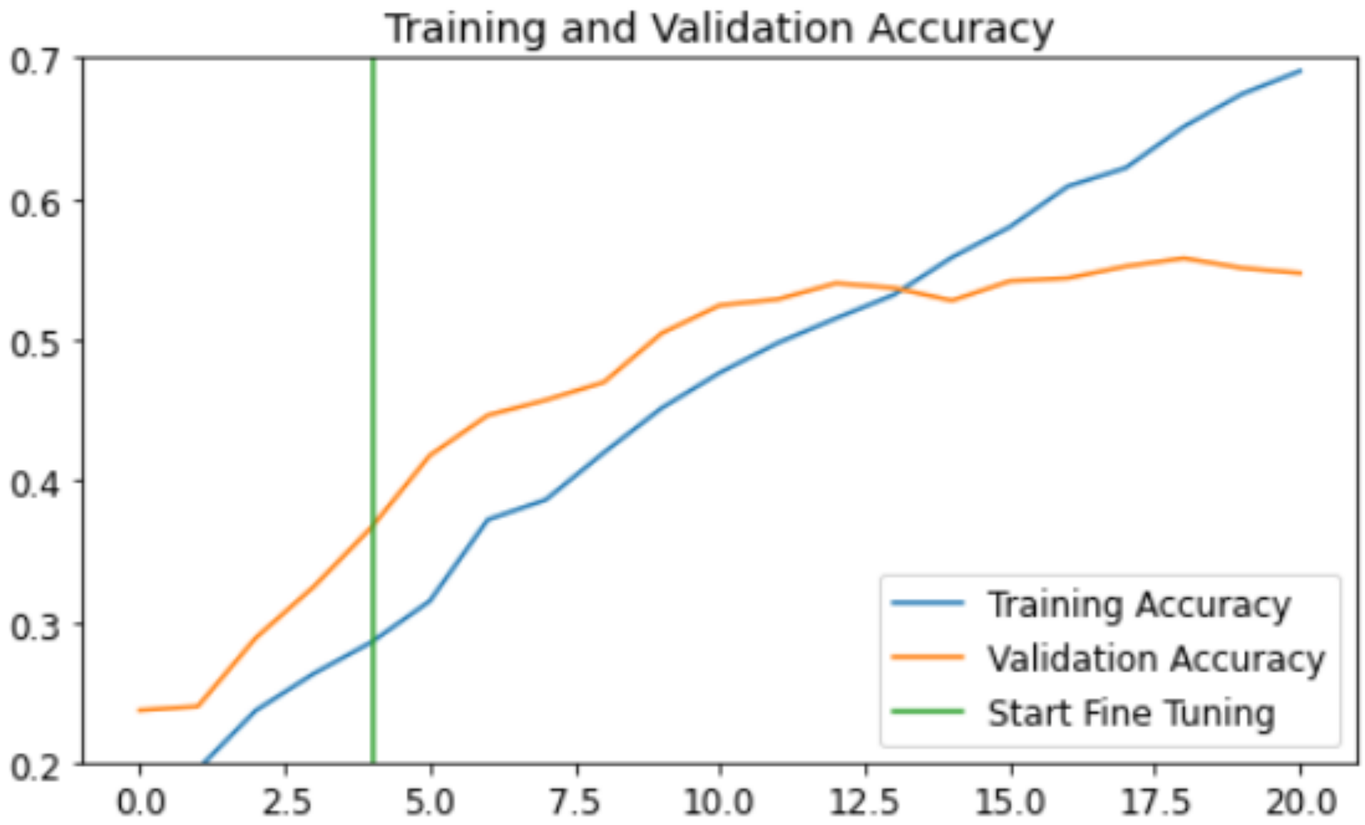
Adapting VggFace

- The VggFace model is trained specifically for Facial Recognition (92.8% to 99.1% accuracy)
- We can still use the Convolutional Base of this model as it has been trained to extract facial features

Third Model: Transfer Learning

Adapting VggFace

- Initial Training
  (Convolutional Layers Frozen)
- Additional Training
  (Unfreeze 5 out of 20 layers)
  (**7 million trainable parameters**)
- Final Accuracy Score: 55%
- **3X Slower** than MobileNetV2
- Saved model is 112 MB!



Evaluation & Results

Misclassification

By how much?

- Our predicted age-groups are given as ordinal, discrete values (from 0 to 9)
- This allows us to see, for each image; what is the difference between the predicted label and the actual label?

age label	Pred Label	Difference
7	6	1
3	2	1
5	5	0
0	1	1
5	5	0

Misclassification

By how much?

- 55.1% of images were classified accordingly
- 38.7% of images were misclassified by only 1 age-group
- 5.2% of images were misclassified by 2 age groups
- The rest: 1% (outlier)

Misclass Difference	Count
0	1840
1	1292
2	172
3	26
4	4
5	2
6	2
7	1

Misclassification

By how much?

- If we combine these two values,
- **The model is able to predict 93.8% of images accurately, up to 1-class difference**

Misclass Difference	Count
0	1840
1	1292
2	172
3	26
4	4
5	2
6	2
7	1

Misclassification

A closer look

- The ‘Baby’ age group is defined for age 1 only
- The ‘Toddler’ age group is defined for ages 2-3
- Due to small range, often overlap

P: Baby A: Toddler

P: Baby A: Toddler

Misclassification

A closer look

- Make up and pose does seem to help adult women appear younger
- And men too!

P: Young Adult A: Adult

P: Young Adult A: Adult

P: Young Adult A: Adult

P: Young Adult A: Adult

P: Adult A: Mature Adult

Misclassification

A closer look

- Makeup on children and teenagers seem to make them appear older

P: Young Adult A: Teenager

P: Child A: Toddler

Misclassification

A closer look

- Blurry images seem to be misclassified a lot as well

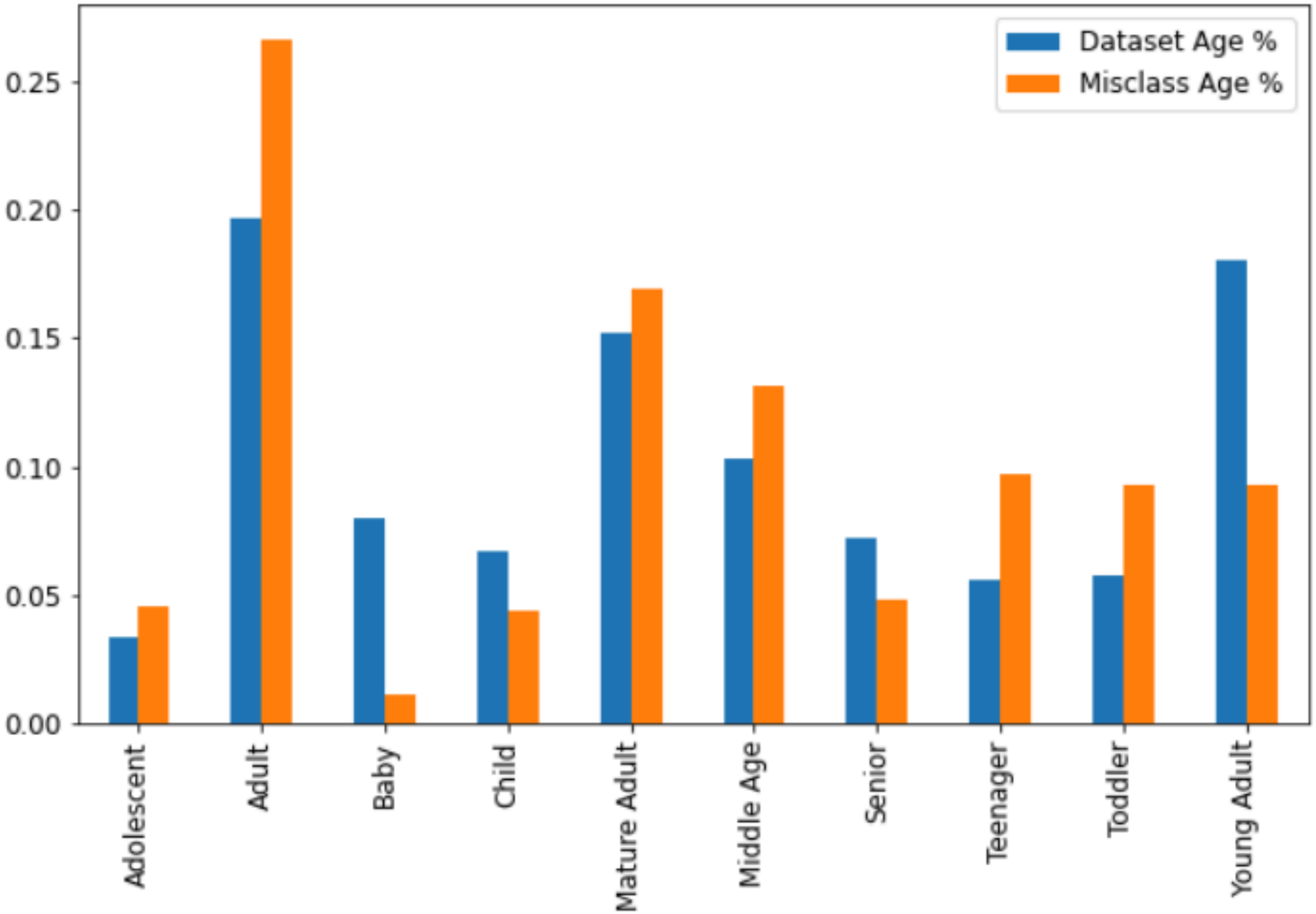
P: Mature Adult A: Adult

P: Senior A: Middle Age

Misclassification

A closer look

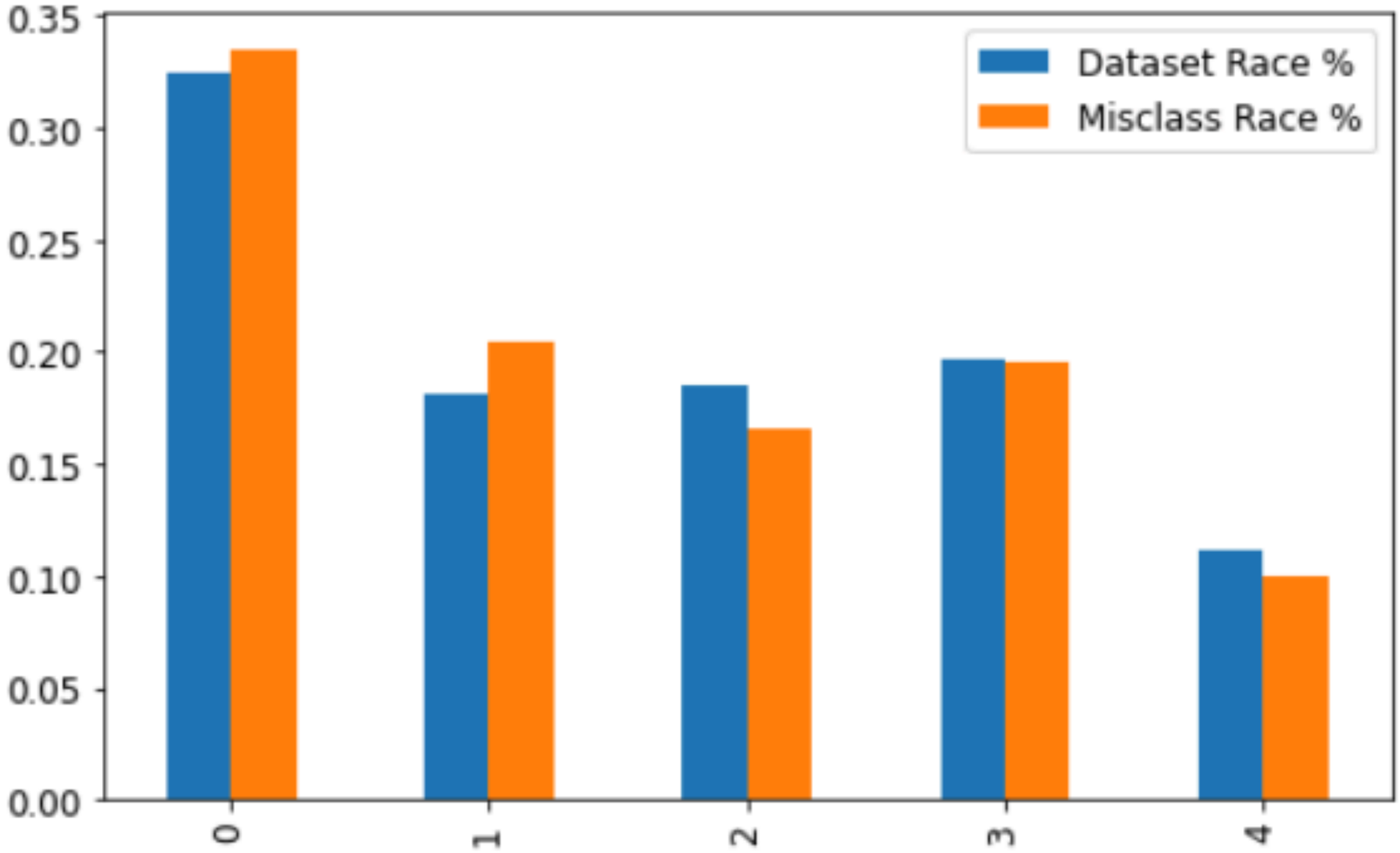
- The age-group imbalance does not seem to affect misclassification



Misclassification

A closer look

- Similarly for the ethnic imbalance as well



Prediction on new data

Misclassification

By how much?

- 40.3% of images were classified accordingly
- The model is able to predict 90.4% of images accurately, up to 1-class difference
- **Disclaimer: I was only able to collect 52 labelled images from family and friends, so this test is quite limited*

Misclass Difference	Count
0	21
1	26
2	2
3	2
4	1

Misclassification

A closer look

- The shiny spectacles seem to be throwing the model off

P: Mature Adult A: Young Adult

P: Adult A: Young Adult

P: Toddler A: Young Adult

Misclassification

A closer look

- Head coverings could be a weakness in the model as well

P: Teenager A: Adult

P: Young Adult A: Middle Age

Presenting….DSI18 Cohort!

P: Adult

P: Adult

P: Adult

P: Mature Adult

P: Adult

P: Adult

P: Mature Adult

P: Adult

P: Adult

P: Mature Adult

P: Young Adult

P: Mature Adult

P: Adult

P: Adult

P: Mature Adult

P: Adult

P: Mature Adult

P: Adult

P: Adult

P: Adult

P: Mature Adult

Guess who?

P: Child

P: Young Adult

P: Adult

P: Toddler

P: Child

P: Toddler

P: Toddler

P: Baby

P: Adult

P: Adult

P: Young Adult

P: Young Adult

Conclusions & Recommendations

Conclusions

- The model can be considered a moderate success as it is able to classify 94% of images up to 1-class accuracy.
- This is expected as the age groups are continuous integers, and hence images of people who are near the boundaries of the age-ranges will tend to be misclassified.
- However, if we look at the exact accuracy score of 54%, there is definitely room for improvement.

Future Improvements

- Fine-tune model further: **vary the number of layers unfrozen** in the CNN-base to better adapt the model to the dataset.
- There are also **other Pre-Trained Convolutional Networks** we could try and adapt to this model.
- Carry out a more thorough **Data Cleaning** process; sieve out poor quality images, etc
- **Look for more Data** to improve imbalance in age, ethnicity instead of removing images
- **Pair with Face segmentation** model for deployment