# Face Recognition using ResNet50, VGG19, GoogleNet,

**Tim Yann Le Cunn - CV3**
10 November 2023

## Meet the Team

1. Moch. Ardhi Kurniawan
2. Muhammad Bramansyah
3. Kio Sato
4. Mahdia Aliyya Nuha Kiswanto
5. Muhammad Fatih Zuhdy

# Timeline

| No | Task | PIC | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|
| a | VGG | | | | | | | |
| a1 | Studi literatur[1] | A | 🟨 | 🟨 | 🟨 | | | |
| a2 | Implementasi arsitektur[2] | A | | 🟨 | 🟨 | | | |
| a3 | Eksplorasi arsitektur[3] | A | | | 🟨 | 🟨 | | |
| b | GoogLeNet | | | | | | | |
| b1 | Studi literatur[1] | F | 🟨 | 🟨 | 🟨 | | | |
| b2 | Implementasi arsitektur[2] | F | | 🟨 | 🟨 | | | |
| b3 | Eksplorasi arsitektur[3] | F | | | 🟨 | 🟨 | | |
| c | Resnet | | | | | | | |
| c1 | Studi literatur[1] | F | 🟨 | 🟨 | 🟨 | | | |
| c2 | Implementasi arsitektur[2] | F | | 🟨 | 🟨 | | | |
| c3 | Eksplorasi arsitektur[3] | F | | | 🟨 | 🟨 | | |
| d | Pemilihan arsitektur final | | | | | | | |
| d1 | Studi literatur[4] | All | | | 🟨 | 🟨 | | |
| d2 | Pemilihan arsitektur final & pembahasan[5] | All | | | | 🟨 | 🟨 | |
| e | Presentasi | | | | | | | |
| e1 | Penyusun materi presentasi[6] | All | | | | 🟨 | 🟨 | 🟨 |
| e2 | Presentasi[7] | All | | | | | | 🟨 |
| | | | | | | | | |

# Background & Problem Statement

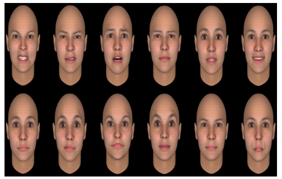
## Background

With technological advances in the fields of image processing and AI, facial recognition or facial recognition systems are becoming an increasingly important and popular application. Facial recognition is technology that allows computers to recognize and identify a persons face from images or videos. This technology is widely used in various fields such as security, attendance management, and user experience on electronic device.

## Problem Statement

Although facial recognition has great potential in a variety of applications, there are still several challenges that need to be overcome to improve its performance and ensure its successful implementations. Some common problems encountered in the development of facial recognition system include:

1. Accuracy
2.Privacy
3.adaptation to facial variability
4. Security
5 . processing efficiency



## Objective

1. Build Face Recognition using VGG.
2. Build Face Recognition using CNN.
3. Build Face Recognition using ResNet.
4. Build Face Recognition using GoogleNet.
5. Optimize the result from 4 different algorithm using Hyperparameter Tuning.
6. Comparing the results from 4 algorithm and choose the perfect algorithm.

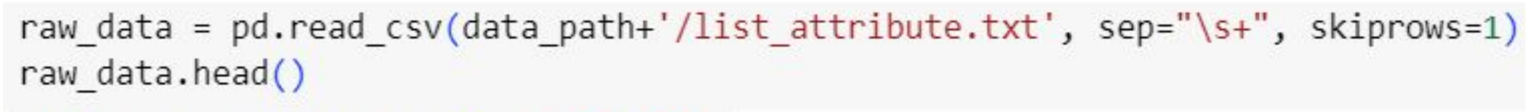
## Data Preprocessing

### Highlight

1. Face image data:
    - Consist of **5000 unique images**, with additional 17 duplicate images.
    - All data is in **one folder**, not separated into sub folders as required by the pytorch Dataset class. To avoid having to move data to each class folder, it is necessary to create an **inheritance class from the Dataset class**.
2. Target Attributes data:
    - The first row contains **invalid data** so it should be ignored.
    - Consist of **202599 rows data**. The data is then **filtered** based on the availability of image data, which produces 5000 valid data.
    - For gender prediction, the required attributes is only in the **'male' column**
    - Values in the 'male' column are **-1 and 1**. For next processing, -1 value was **changed to 0**.
3. The data is divided into **Test/Validation/Test** datasets with a ratio of **60/20/20**.
4. In the initial experiments, only a **simple transformation process** was performed on the image data and it was found that the prediction accuracy was already quite high. Based on these results and due to time and hardware limitation, it was decided not to perform further data augmentation process.

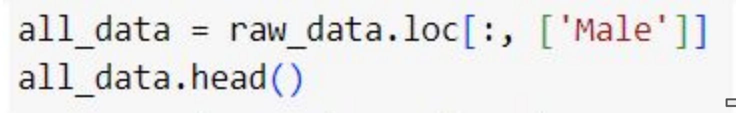
# Data Collection & Preparation

Based on file “list_attribute” that contain the attributes, select only ‘male’ column, change -1 into 0, and filter it based on the available images that have been provided, and split it into train, valid, and test, with 60/20/20



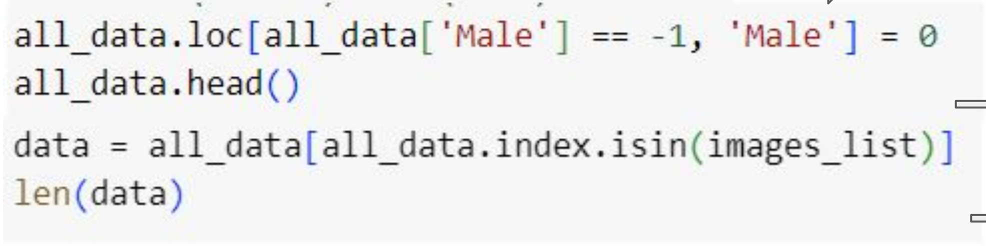
```
raw_data = pd.read_csv(data_path+'/list_attribute.txt', sep="\s+", skiprows=1)
raw_data.head()
```

skip unwanted first row



```
all_data = raw_data.loc[:, ['Male']]
all_data.head()
```

Only select ‘Male’ column

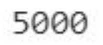


```
all_data.loc[all_data['Male'] == -1, 'Male'] = 0
all_data.head()
```

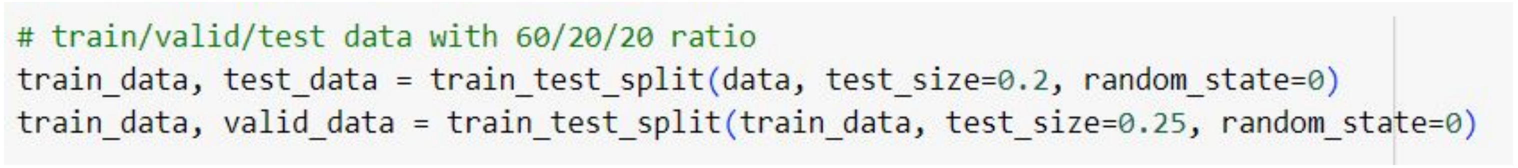
Convert -1 value to 0

```
data = all_data[all_data.index.isin(images_list)]
len(data)
```

Only use attribute data that has image data



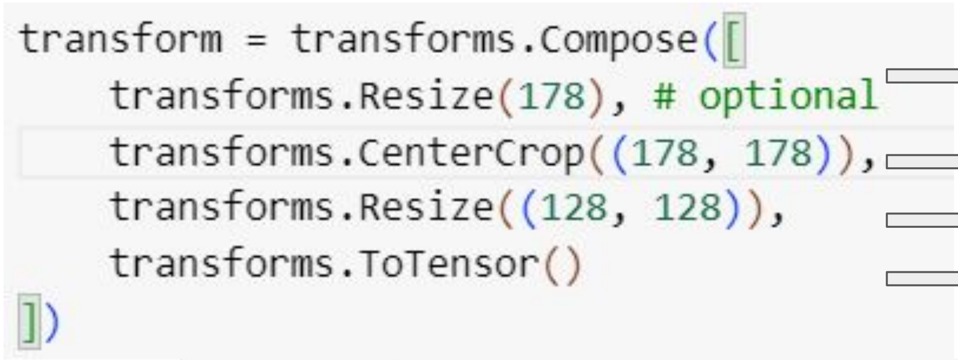
```
5000
```



```
# train/valid/test data with 60/20/20 ratio
train_data, test_data = train_test_split(data, test_size=0.2, random_state=0)
train_data, valid_data = train_test_split(train_data, test_size=0.25, random_state=0)
```

# Data Collection & Preparation

After the divided phase in text file, make folder train, valid and test and in each of it contain subfolder ‘0’ and ‘1’ based on the text file that have been divided before, and transform/augmented it then apply it



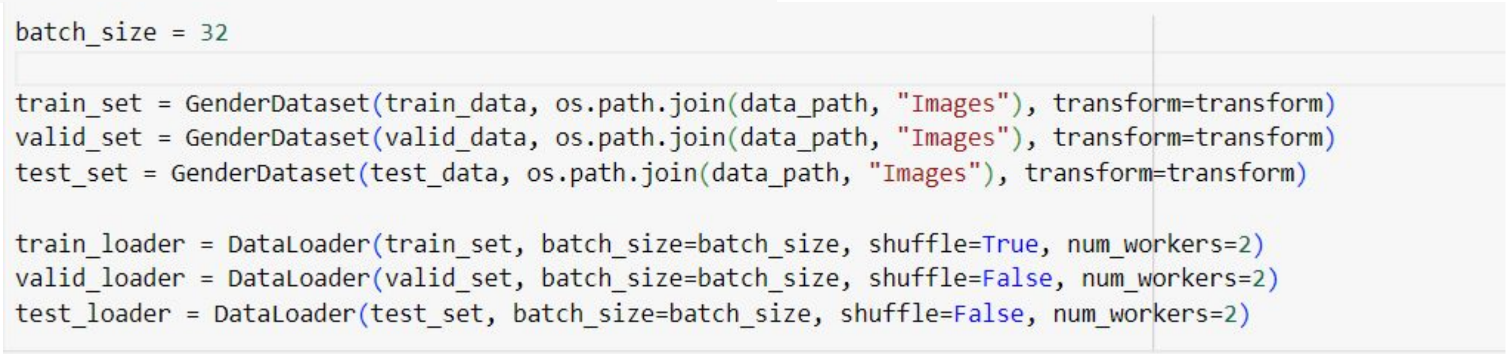
```python
transform = transforms.Compose([
    transforms.Resize(178), # optional
    transforms.CenterCrop((178, 178)),
    transforms.Resize((128, 128)),
    transforms.ToTensor()
])
```

Resize input image into 178

Center crop on the image, extract square region of size (178, 178) from the center

Resize again into 128

Convert into PyTorch tensor.



```python
batch_size = 32

train_set = GenderDataset(train_data, os.path.join(data_path, "Images"), transform=transform)
valid_set = GenderDataset(valid_data, os.path.join(data_path, "Images"), transform=transform)
test_set = GenderDataset(test_data, os.path.join(data_path, "Images"), transform=transform)

train_loader = DataLoader(train_set, batch_size=batch_size, shuffle=True, num_workers=2)
valid_loader = DataLoader(valid_set, batch_size=batch_size, shuffle=False, num_workers=2)
test_loader = DataLoader(test_set, batch_size=batch_size, shuffle=False, num_workers=2)
```

# Modeling

## Highlight

1. The **training and testing** procedures for all models are performed in the same procedure. The model to be tested is injected as a parameter.
2. Initial experiment use model without pre-trained weights. Further experiment then use **pre-trained weights from pytorch database**. (https://pytorch.org/vision/0.16/models.html#classification)
3. For initial experiments, only a **simple transformation process** was performed on the image data.
4. For initial experiments, the **fine-tuning process was carried out minimally**, only on the output layer.
5. From the initial experiments it was found that the prediction accuracy was already quite high. Based on these results and due to time and hardware limitation, it was decided **not to perform further hyperparameter tuning and fine-tuning process**.

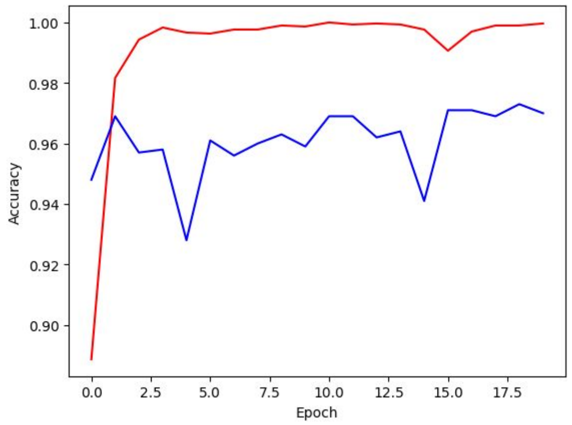
## Model Development

Summarise of all model that we’ve been created*

| Model | Training Time | Accuracy |
|---|---|---|
| CNN | ±70 minutes | 0.9420 |
| GoogleNet | ±4 minutes, 8 seconds | 0.960000 |
| Resnet50 | ±4 minutes, 56 seconds | 0.973000 |
| Vgg19 | ±65 minutes | 0.8220 |

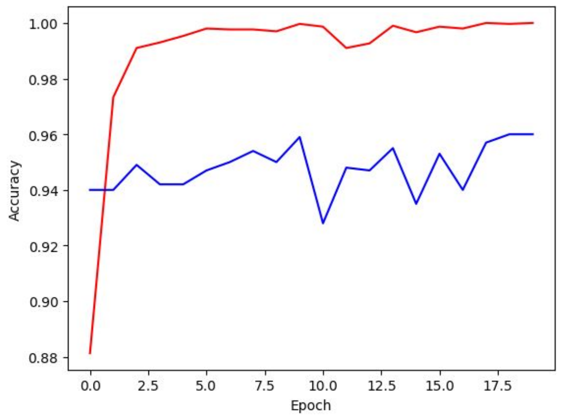
* not performed in a controlled test environment

# Training and validation accuracy each model



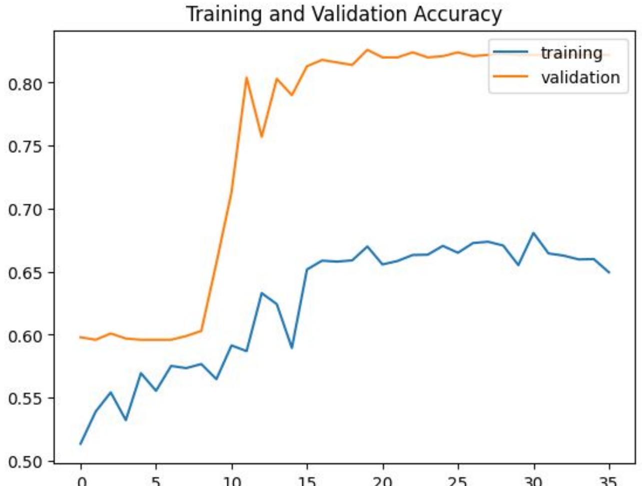
Resnet50

Red : train
Blue : valid

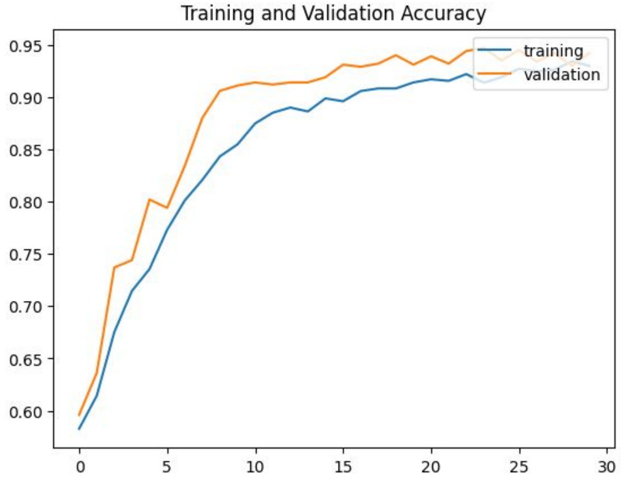


Googlenet

Red : train
Blue : valid



VGG19



CNN

## Evaluation

### Highlight

1. From existing dataset, 20% of the data is separated as Test dataset. This dataset is not visible during the training process. The model then tested using this Test dataset.
2. As an additional test, the prediction accuracy was also tested with image data obtained from the internet.

# Predict pictures with best model (Resnet50)

predicted: male

predicted: female

predicted: male

predicted: female

predicted: female

predicted: male

predicted: female

predicted: female

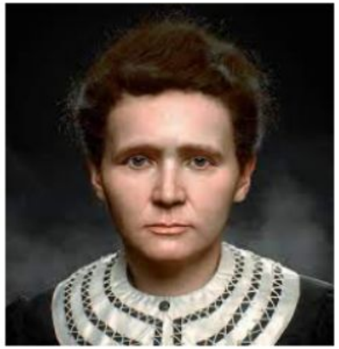
```
do_predict("hinton.jpg")
```

```
'male'
```

```
do_predict("ffl.jpg")
```

```
'female'
```

```
do_predict("curie.jpg")
```

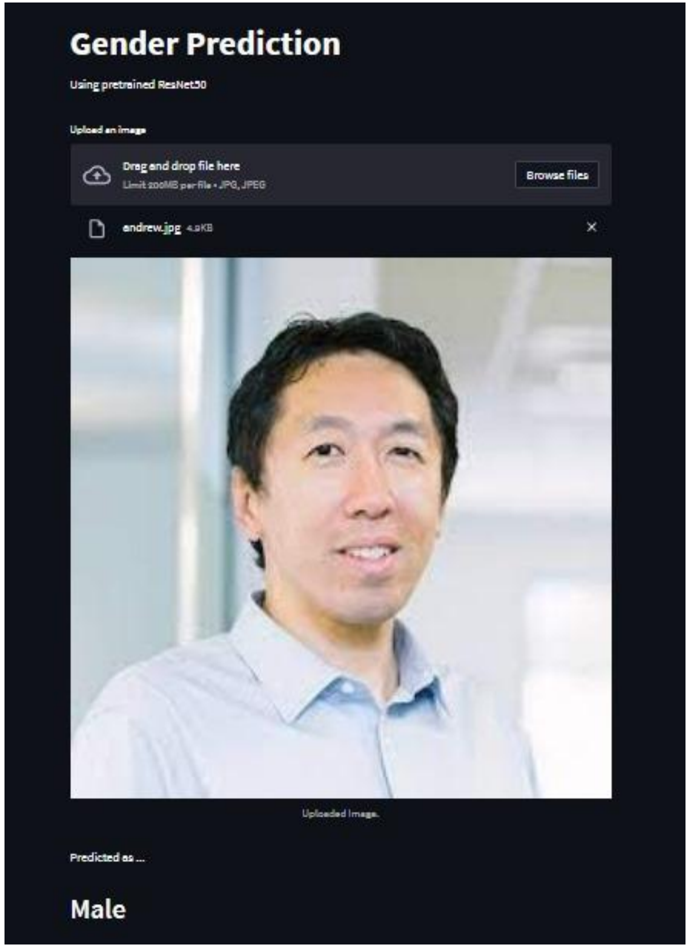


```
'male'
```

```
do_predict("curie2.jpg")
```

```
'female'
```

# Deploy model in local streamlit



Gender Prediction

Using pretrained ResNet50

Upload an image

Uploaded Image.

Predicted as ...

Male

## Future Improvement

- Since all of data train and valid are mostly focus around the face, maybe we need to variate the data that not only focus around the face
- Because, the train data only focus around the face, what we afraid is, when the input data is photos of full body, our model can’t predict it properly
- Add some more data, so it can be more variate

## Conclusion

- From all of the models that we’ve built, the best one is using Resnet50
- Besides the high accuracy, we need to consider the training time of the model, and Resnet50 achieve all of it, with accuracy in validation data around 97%
- We recommend that, when using our models, use photos that focus around the face