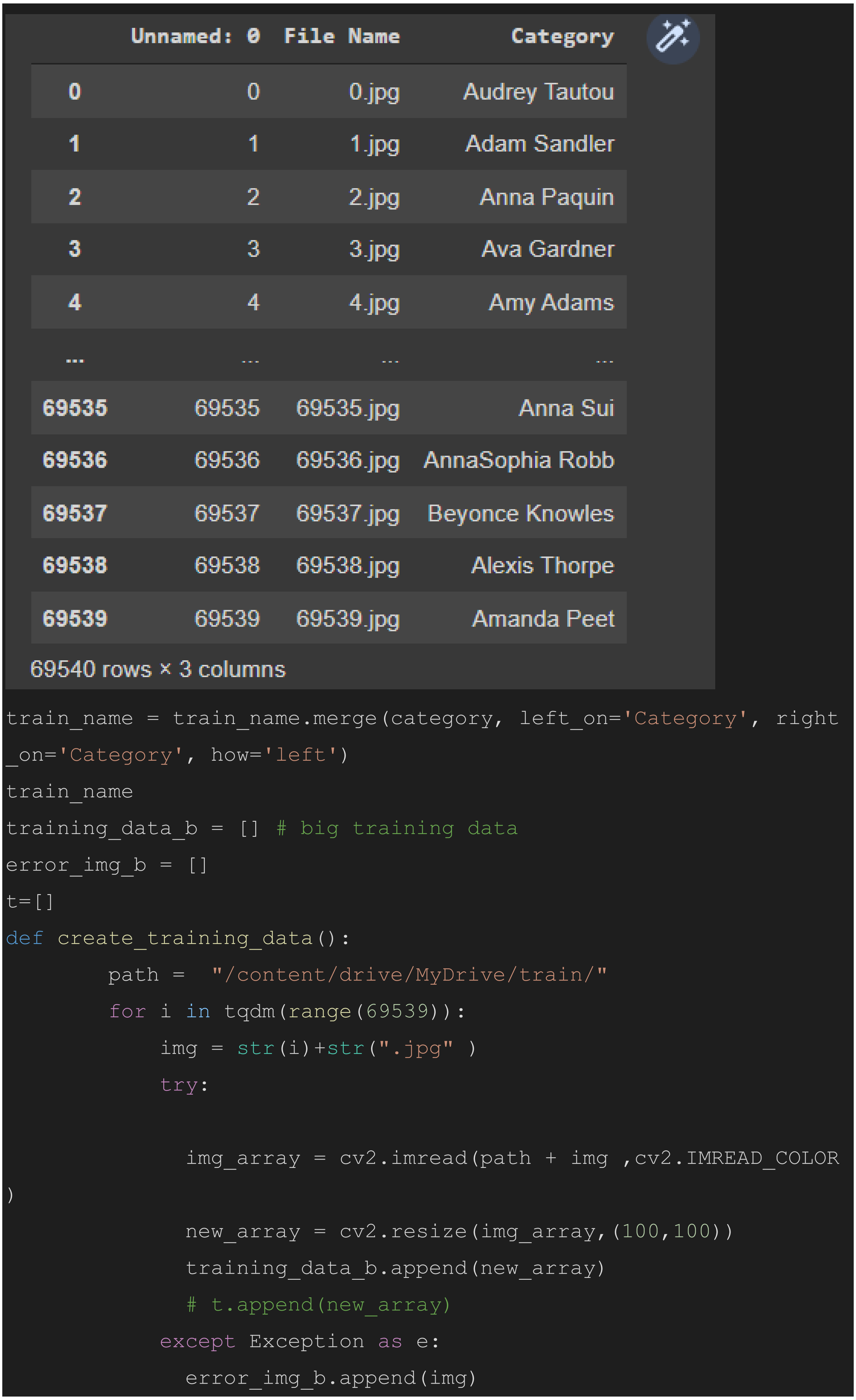
ECE 595

Na Yun Lai

1. The name on the kaggle : Yun Lai
2. Code

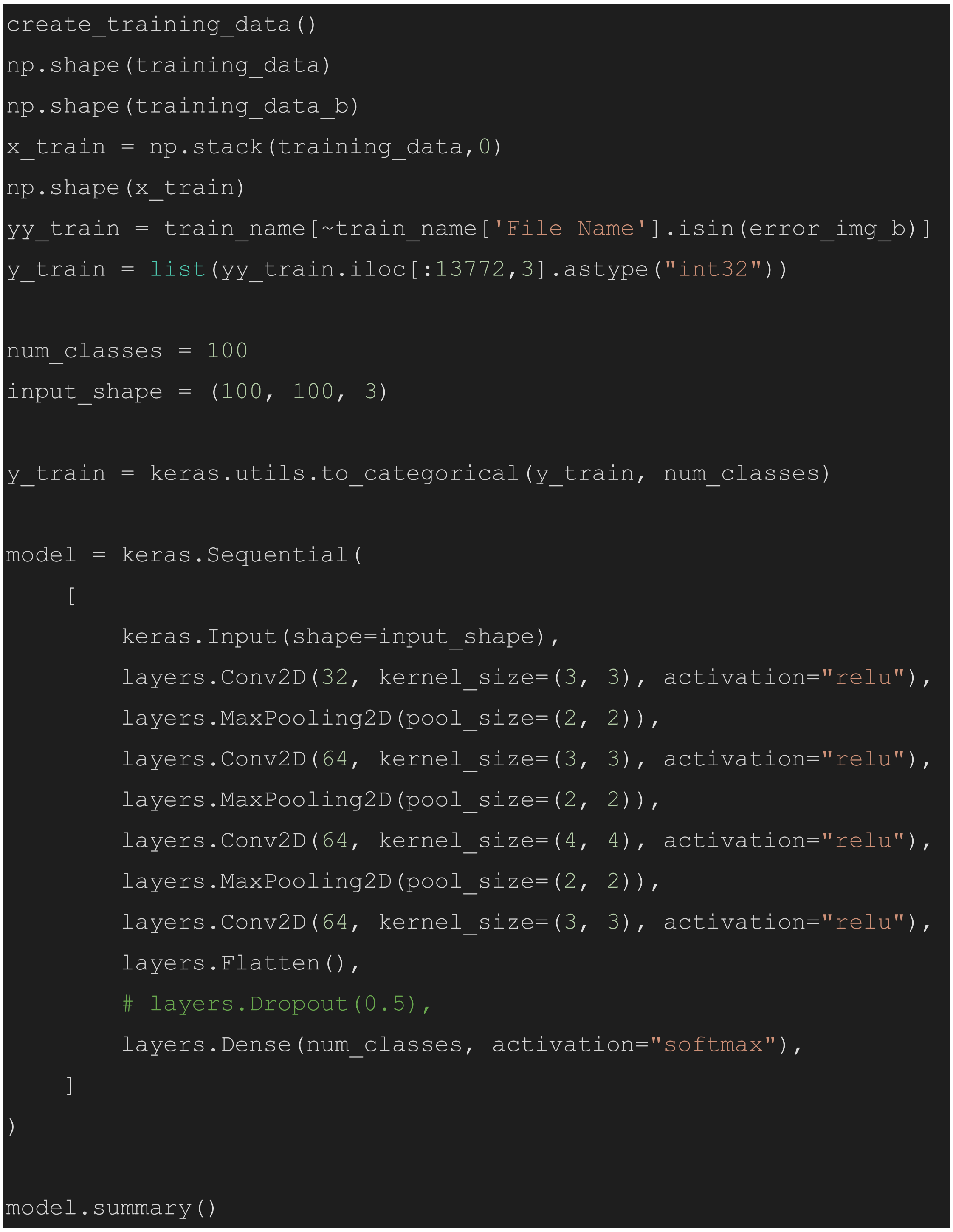
```
 !pip install opencv-python
from google.colab import drive
drive.mount('/content/drive')

import pandas as pd
import numpy as np
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm
from google.colab.patches import cv2_imshow
import matplotlib.pyplot as plt
import cv2
import numpy as np
from tensorflow import keras
from tensorflow.keras import layers
from matplotlib import pyplot as plt
import pdb

train_name = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/
ece595ml-
s2022/train.csv")  # local drive how to direct get the data
category = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/ec
e595ml-s2022/category.csv")
train_name
```



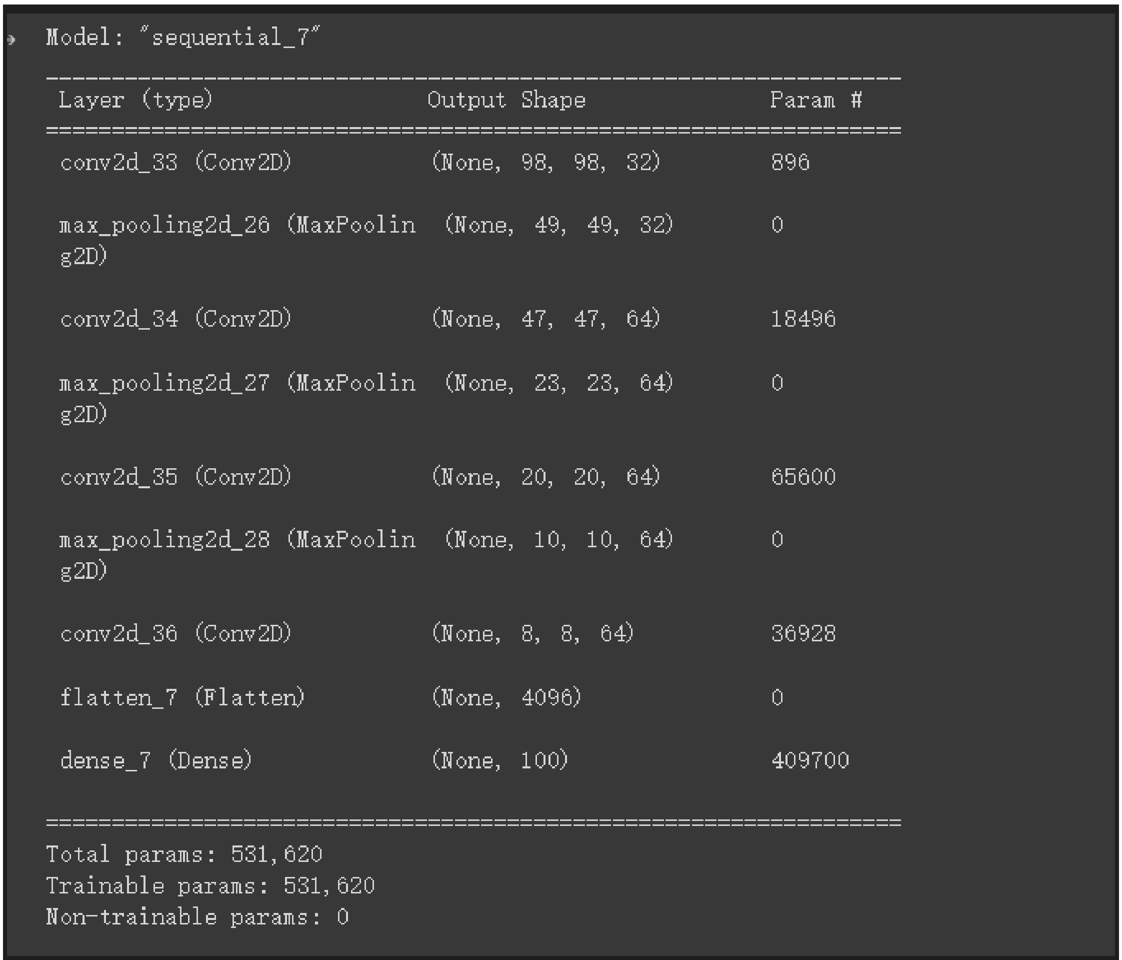
	Unnamed: 0	File Name	Category
0	0	0.jpg	Audrey Tautou
1	1	1.jpg	Adam Sandler
2	2	2.jpg	Anna Paquin
3	3	3.jpg	Ava Gardner
4	4	4.jpg	Amy Adams
...	...	...	...
69535	69535	69535.jpg	Anna Sui
69536	69536	69536.jpg	AnnaSophia Robb
69537	69537	69537.jpg	Beyonce Knowles
69538	69538	69538.jpg	Alexis Thorpe
69539	69539	69539.jpg	Amanda Peet

69540 rows × 3 columns

```
train_name = train_name.merge(category, left_on='Category', right_on='Category', how='left')
train_name
training_data_b = [] # big training data
error_img_b = []
t=[]
def create_training_data():
        path =  "/content/drive/MyDrive/train/"
        for i in tqdm(range(69539)):
            img = str(i)+str(".jpg" )
            try:

              img_array = cv2.imread(path + img ,cv2.IMREAD_COLOR)

              new_array = cv2.resize(img_array,(100,100))
              training_data_b.append(new_array)
              # t.append(new_array)
            except Exception as e:
              error_img_b.append(img)
```

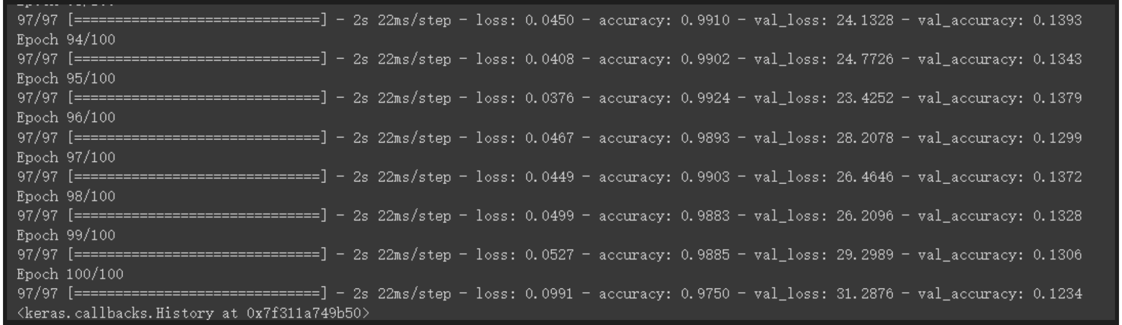


```
create_training_data()
np.shape(training_data)
np.shape(training_data_b)
x_train = np.stack(training_data,0)
np.shape(x_train)
yy_train = train_name[~train_name['File Name'].isin(error_img_b)]
y_train = list(yy_train.iloc[:13772,3].astype("int32"))

num_classes = 100
input_shape = (100, 100, 3)

y_train = keras.utils.to_categorical(y_train, num_classes)

model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(4, 4), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.Flatten(),
        # layers.Dropout(0.5),
        layers.Dense(num_classes, activation="softmax"),
    ]
)

model.summary()
```



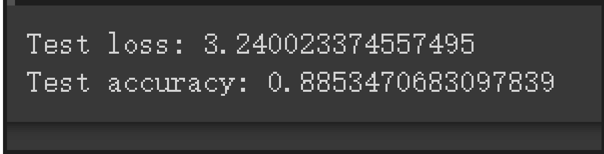
```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_33 (Conv2D)          (None, 98, 98, 32)        896

 max_pooling2d_26 (MaxPoolin  (None, 49, 49, 32)       0
 g2D)

 conv2d_34 (Conv2D)          (None, 47, 47, 64)        18496

 max_pooling2d_27 (MaxPoolin  (None, 23, 23, 64)       0
 g2D)

 conv2d_35 (Conv2D)          (None, 20, 20, 64)        65600

 max_pooling2d_28 (MaxPoolin  (None, 10, 10, 64)       0
 g2D)

 conv2d_36 (Conv2D)          (None, 8, 8, 64)          36928

 flatten_7 (Flatten)         (None, 4096)              0

 dense_7 (Dense)             (None, 100)               409700

=================================================================
Total params: 531,620
Trainable params: 531,620
Non-trainable params: 0
```

```
#run model
batch_size = 128
epochs = 100
model.compile(loss="categorical_crossentropy", optimizer="adam",
metrics=["accuracy"])

model.fit(x_train, y_train, batch_size = batch_size, epochs=epoch
s, validation_split=0.1 )
```



```
97/97 [==============================] - 2s 22ms/step - loss: 0.0450 - accuracy: 0.9910 - val_loss: 24.1328 - val_accuracy: 0.1393
Epoch 94/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0408 - accuracy: 0.9902 - val_loss: 24.7726 - val_accuracy: 0.1343
Epoch 95/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0376 - accuracy: 0.9924 - val_loss: 23.4252 - val_accuracy: 0.1379
Epoch 96/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0467 - accuracy: 0.9893 - val_loss: 28.2078 - val_accuracy: 0.1299
Epoch 97/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0449 - accuracy: 0.9903 - val_loss: 26.4646 - val_accuracy: 0.1372
Epoch 98/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0499 - accuracy: 0.9883 - val_loss: 26.2096 - val_accuracy: 0.1328
Epoch 99/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0527 - accuracy: 0.9885 - val_loss: 29.2989 - val_accuracy: 0.1306
Epoch 100/100
97/97 [==============================] - 2s 22ms/step - loss: 0.0991 - accuracy: 0.9750 - val_loss: 31.2876 - val_accuracy: 0.1234
<keras.callbacks.History at 0x7f311a749b50>
```

```
score = model.evaluate(x_train, y_train, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```



```
Test loss: 3.240023374557495
Test accuracy: 0.8853470683097839
```

```
x_test=[]
np.shape(org_tr_data)
x_test.append(org_tr_data[1])
np.shape(x_test)
x_test = []  #no crop

def testing_data():
        path =  "/content/drive/MyDrive/test/test/"
        for i in tqdm(range(4977)):
            img = str(i)+str(".jpg" )
            try:
              img_array = cv2.imread(path+img,cv2.IMREAD_COLOR)
              new_array = cv2.resize(img_array,(100,100))
              x_test.append(new_array)
            except Exception as e:
              #just add a name attribute
              x_test.append(org_tr_data[65])
testing_data()
x_test = np.stack(x_test,0)
x_test.shape
category = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ec
e595ml-
s2022/category.csv',skiprows=[0], names = ['Num','Category'])
y_test =[]
id = []
for i in tqdm(range(4977)):
  yhat = model.predict(x_test[i:i+1])
  index = np.where(yhat ==yhat.max())[1][0]
  y_test.append(category.loc[category['Num'] ==index, 'Category']
.tolist()[0])


Output = {'Category': y_test}
```

```
Output_df = pd.DataFrame(Output)

Output_df.to_csv('y_test_big.csv', index=True)
```