CarStyle

October 1, 2024

[1]:
```
import tensorflow as tf
import os
import cv2
import math
import json
import numpy as np
from matplotlib import pyplot as plt
from keras.applications import MobileNet
from keras.models import Model
from keras.layers import Dense, GlobalAveragePooling2D
from keras.metrics import Precision, Recall, SparseCategoricalAccuracy
```

[2]:
```
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
gpus = tf.config.experimental.list_physical_devices('GPU')
if gpus:
    try:
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)
        logical_gpus = tf.config.experimental.list_logical_devices('GPU')
        print(len(gpus), "Physical GPUs,", len(logical_gpus), "Logical GPUs")
    except RuntimeError as e:
        print(e)
```

```
Num GPUs Available:  1
1 Physical GPUs, 1 Logical GPUs
```

[3]:
```
base_dir = 'Styles'
train_dir = os.path.join(base_dir, 'train')
val_dir = os.path.join(base_dir, 'valid')
test_dir = os.path.join(base_dir, 'test')

img_size = (224, 224)
batch_size = 32

train_data = tf.keras.utils.image_dataset_from_directory(
    train_dir,
    image_size=img_size,
    batch_size=batch_size,
```

```
    label_mode='int',
    interpolation='bilinear'
)

val_data = tf.keras.utils.image_dataset_from_directory(
    val_dir,
    image_size=img_size,
    batch_size=batch_size,
    label_mode='int',
    interpolation='bilinear'
)

test_data = tf.keras.utils.image_dataset_from_directory(
    test_dir,
    image_size=img_size,
    batch_size=batch_size,
    label_mode='int',
    interpolation='bilinear'
)
```

```
Found 5350 files belonging to 7 classes.
Found 1397 files belonging to 7 classes.
Found 802 files belonging to 7 classes.
```

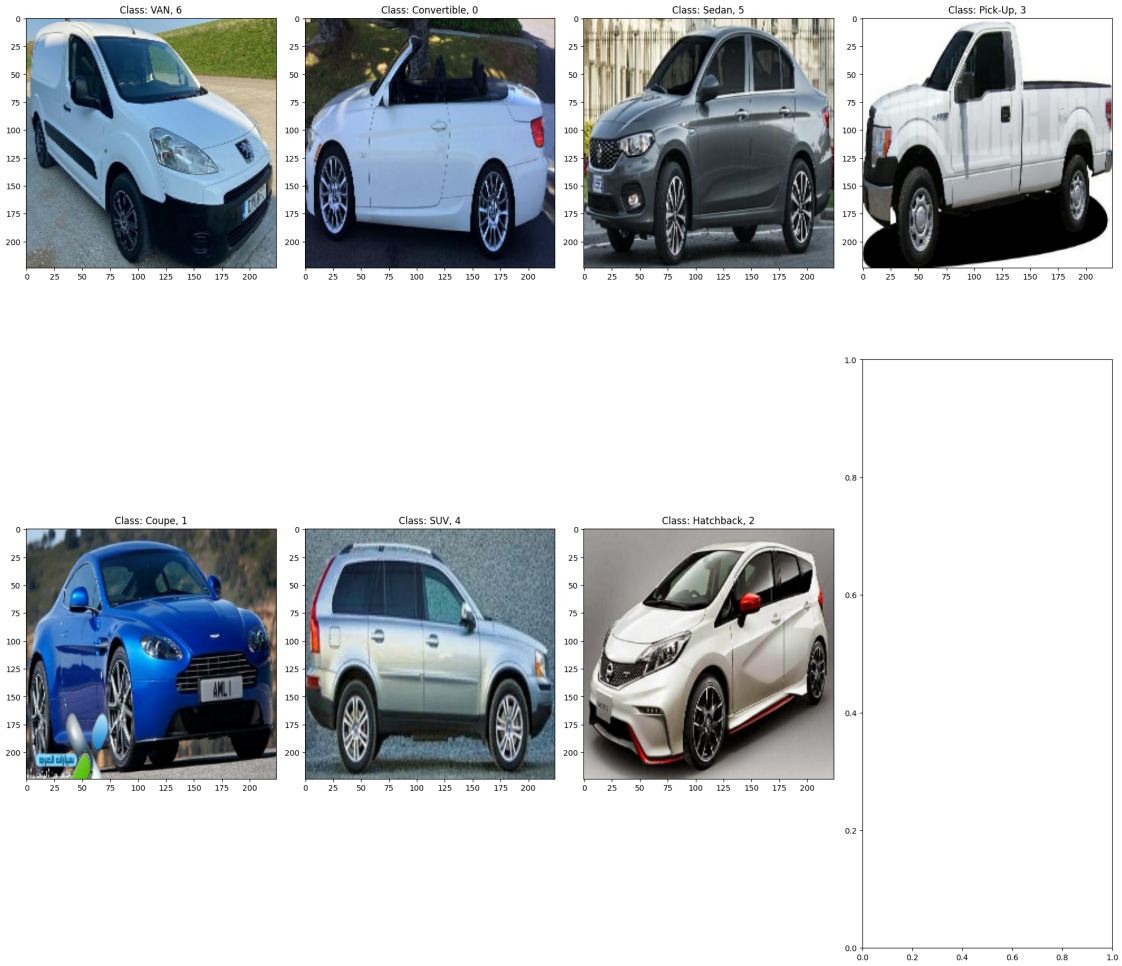
[4]:
```
class_names = train_data.class_names
print("Class names test:", class_names)

with open('CarStyle map.json', 'w') as f:
    json.dump(class_names, f)

data_iterator = train_data.as_numpy_iterator()
```

```
Class names test: ['Convertible', 'Coupe', 'Hatchback', 'Pick-Up', 'SUV',
'Sedan', 'VAN']
```

[5]:
```
batch = data_iterator.next()
num_classes = len(class_names)
```

[6]:
```
ncols = 4
nrows = math.ceil(num_classes / ncols)
fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=(20, 20))

if nrows == 1:
    ax = ax.flatten()
elif ncols == 1:
    ax = ax.flatten()

plotted = set()
```

```
count = 0
while count < num_classes:
    batch = next(data_iterator)

    for idx, img in enumerate(batch[0]):
        label = batch[1][idx]
        if label not in plotted:
            ax_idx = count if nrows == 1 or ncols == 1 else (count // ncols,␣
 ↪count % ncols)
            ax[ax_idx].imshow(img.astype(int))
            ax[ax_idx].title.set_text(f"Class: {class_names[label]}, {label}")
            plotted.add(label)
            count += 1

        if count == num_classes:
            break

plt.tight_layout()
plt.show()
```



[7]:
```
base_model = MobileNet(
    weights='imagenet',
    include_top=False,
    input_shape=(224, 224, 3)
)
base_model.summary()
```

```
Model: "mobilenet_1.00_224"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1 (Conv2D)              (None, 112, 112, 32)      864

 conv1_bn (BatchNormalizatio  (None, 112, 112, 32)     128
 n)

 conv1_relu (ReLU)           (None, 112, 112, 32)      0

 conv_dw_1 (DepthwiseConv2D)  (None, 112, 112, 32)     288

 conv_dw_1_bn (BatchNormaliz  (None, 112, 112, 32)     128
 ation)

 conv_dw_1_relu (ReLU)       (None, 112, 112, 32)      0

 conv_pw_1 (Conv2D)          (None, 112, 112, 64)      2048

 conv_pw_1_bn (BatchNormaliz  (None, 112, 112, 64)     256
 ation)

 conv_pw_1_relu (ReLU)       (None, 112, 112, 64)      0

 conv_pad_2 (ZeroPadding2D)  (None, 113, 113, 64)      0

 conv_dw_2 (DepthwiseConv2D)  (None, 56, 56, 64)       576

 conv_dw_2_bn (BatchNormaliz  (None, 56, 56, 64)       256
 ation)

 conv_dw_2_relu (ReLU)       (None, 56, 56, 64)        0

 conv_pw_2 (Conv2D)          (None, 56, 56, 128)       8192
```

```
conv_pw_2_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_pw_2_relu (ReLU)       (None, 56, 56, 128)       0

conv_dw_3 (DepthwiseConv2D)  (None, 56, 56, 128)      1152

conv_dw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_dw_3_relu (ReLU)       (None, 56, 56, 128)       0

conv_pw_3 (Conv2D)          (None, 56, 56, 128)       16384

conv_pw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_pw_3_relu (ReLU)       (None, 56, 56, 128)       0

conv_pad_4 (ZeroPadding2D)  (None, 57, 57, 128)       0

conv_dw_4 (DepthwiseConv2D)  (None, 28, 28, 128)      1152

conv_dw_4_bn (BatchNormaliz  (None, 28, 28, 128)      512
ation)

conv_dw_4_relu (ReLU)       (None, 28, 28, 128)       0

conv_pw_4 (Conv2D)          (None, 28, 28, 256)       32768

conv_pw_4_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_pw_4_relu (ReLU)       (None, 28, 28, 256)       0

conv_dw_5 (DepthwiseConv2D)  (None, 28, 28, 256)      2304

conv_dw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_dw_5_relu (ReLU)       (None, 28, 28, 256)       0

conv_pw_5 (Conv2D)          (None, 28, 28, 256)       65536

conv_pw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_pw_5_relu (ReLU)       (None, 28, 28, 256)       0
```

```
conv_pad_6 (ZeroPadding2D)  (None, 29, 29, 256)       0

conv_dw_6 (DepthwiseConv2D)  (None, 14, 14, 256)      2304

conv_dw_6_bn (BatchNormaliz  (None, 14, 14, 256)      1024
ation)

conv_dw_6_relu (ReLU)       (None, 14, 14, 256)       0

conv_pw_6 (Conv2D)          (None, 14, 14, 512)       131072

conv_pw_6_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_6_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_7 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_7_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_dw_7_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_7 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_7_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_7_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_8 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_8_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_dw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_8 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_8_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_9 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_9_bn (BatchNormaliz  (None, 14, 14, 512)      2048
```

```
ation)

conv_dw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_9 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_9_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_10 (DepthwiseConv2D  (None, 14, 14, 512)      4608
)

conv_dw_10_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_dw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_pw_10 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_10_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_pw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_dw_11 (DepthwiseConv2D  (None, 14, 14, 512)      4608
)

conv_dw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_dw_11_relu (ReLU)      (None, 14, 14, 512)       0

conv_pw_11 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_pw_11_relu (ReLU)      (None, 14, 14, 512)       0

conv_pad_12 (ZeroPadding2D)  (None, 15, 15, 512)      0

conv_dw_12 (DepthwiseConv2D  (None, 7, 7, 512)        4608
)

conv_dw_12_bn (BatchNormali  (None, 7, 7, 512)        2048
zation)
```

```
 conv_dw_12_relu (ReLU)      (None, 7, 7, 512)         0

 conv_pw_12 (Conv2D)         (None, 7, 7, 1024)        524288

 conv_pw_12_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_pw_12_relu (ReLU)      (None, 7, 7, 1024)        0

 conv_dw_13 (DepthwiseConv2D  (None, 7, 7, 1024)       9216
 )

 conv_dw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_dw_13_relu (ReLU)      (None, 7, 7, 1024)        0

 conv_pw_13 (Conv2D)         (None, 7, 7, 1024)        1048576

 conv_pw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_pw_13_relu (ReLU)      (None, 7, 7, 1024)        0

=================================================================
Total params: 3,228,864
Trainable params: 3,206,976
Non-trainable params: 21,888
_________________________________________________________________
```

[8]:
```
x = base_model.output
x = GlobalAveragePooling2D()(x)
output = Dense(num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=output)

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir='logs')

hist = model.fit(train_data, epochs=20, validation_data=val_data,␣
 ↪callbacks=[tensorboard_callback])
```

```
Epoch 1/20
168/168 [==============================] - 12s 54ms/step - loss: 0.4827 -
accuracy: 0.8329 - val_loss: 1.0766 - val_accuracy: 0.7230
```

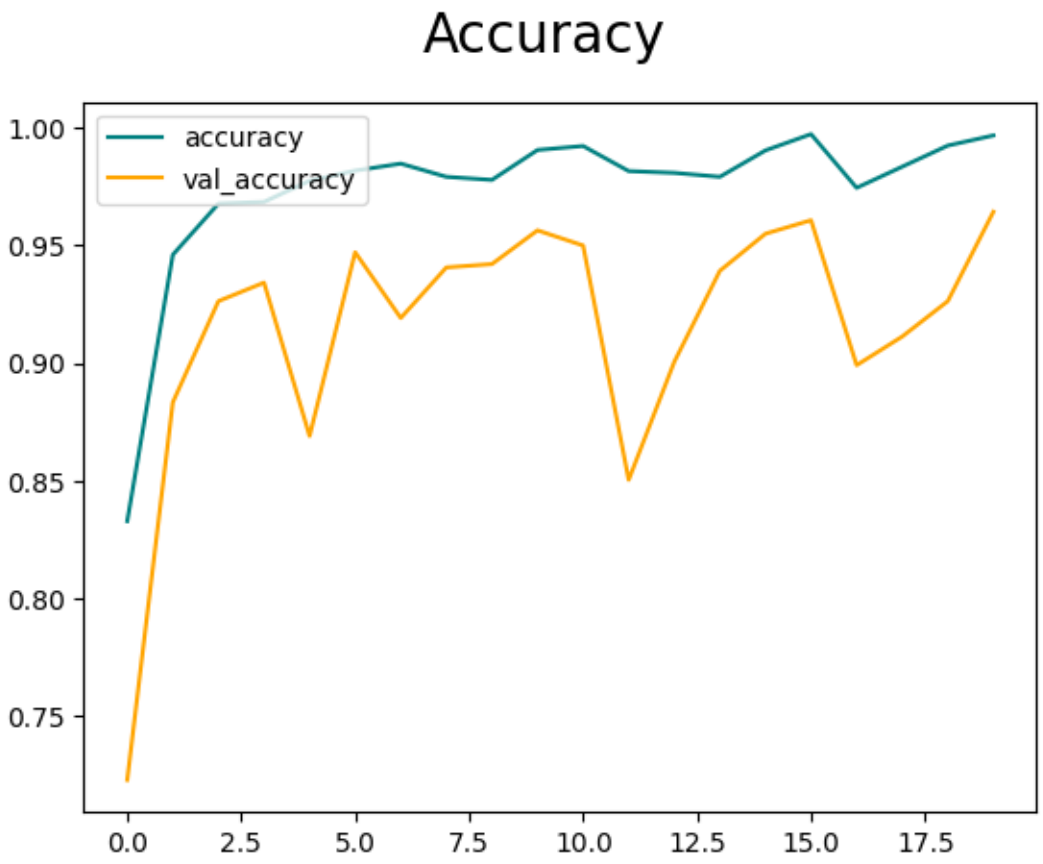
```
Epoch 2/20
168/168 [==============================] - 9s 53ms/step - loss: 0.1517 -
accuracy: 0.9460 - val_loss: 0.3466 - val_accuracy: 0.8833
Epoch 3/20
168/168 [==============================] - 9s 53ms/step - loss: 0.1027 -
accuracy: 0.9677 - val_loss: 0.2518 - val_accuracy: 0.9263
Epoch 4/20
168/168 [==============================] - 9s 54ms/step - loss: 0.0932 -
accuracy: 0.9684 - val_loss: 0.2440 - val_accuracy: 0.9341
Epoch 5/20
168/168 [==============================] - 10s 56ms/step - loss: 0.0593 -
accuracy: 0.9774 - val_loss: 0.5444 - val_accuracy: 0.8690
Epoch 6/20
168/168 [==============================] - 10s 57ms/step - loss: 0.0509 -
accuracy: 0.9817 - val_loss: 0.1731 - val_accuracy: 0.9470
Epoch 7/20
168/168 [==============================] - 10s 57ms/step - loss: 0.0426 -
accuracy: 0.9847 - val_loss: 0.3215 - val_accuracy: 0.9191
Epoch 8/20
168/168 [==============================] - 10s 57ms/step - loss: 0.0667 -
accuracy: 0.9791 - val_loss: 0.2126 - val_accuracy: 0.9406
Epoch 9/20
168/168 [==============================] - 9s 53ms/step - loss: 0.0633 -
accuracy: 0.9778 - val_loss: 0.2417 - val_accuracy: 0.9420
Epoch 10/20
168/168 [==============================] - 9s 53ms/step - loss: 0.0250 -
accuracy: 0.9905 - val_loss: 0.1601 - val_accuracy: 0.9563
Epoch 11/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0210 -
accuracy: 0.9921 - val_loss: 0.1807 - val_accuracy: 0.9499
Epoch 12/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0561 -
accuracy: 0.9815 - val_loss: 0.8208 - val_accuracy: 0.8504
Epoch 13/20
168/168 [==============================] - 10s 56ms/step - loss: 0.0606 -
accuracy: 0.9807 - val_loss: 0.3563 - val_accuracy: 0.9005
Epoch 14/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0622 -
accuracy: 0.9791 - val_loss: 0.2530 - val_accuracy: 0.9392
Epoch 15/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0295 -
accuracy: 0.9903 - val_loss: 0.1988 - val_accuracy: 0.9549
Epoch 16/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0076 -
accuracy: 0.9972 - val_loss: 0.1661 - val_accuracy: 0.9606
Epoch 17/20
168/168 [==============================] - 9s 55ms/step - loss: 0.0864 -
accuracy: 0.9744 - val_loss: 0.4750 - val_accuracy: 0.8991
```

```
Epoch 18/20
168/168 [==============================] - 9s 54ms/step - loss: 0.0473 -
accuracy: 0.9834 - val_loss: 0.3667 - val_accuracy: 0.9112
Epoch 19/20
168/168 [==============================] - 9s 53ms/step - loss: 0.0206 -
accuracy: 0.9923 - val_loss: 0.2806 - val_accuracy: 0.9263
Epoch 20/20
168/168 [==============================] - 9s 54ms/step - loss: 0.0099 -
accuracy: 0.9966 - val_loss: 0.1412 - val_accuracy: 0.9642
```

[9]:
```
fig = plt.figure()
plt .plot(hist.history['loss'], color='teal', label='loss')
plt.plot(hist.history['val_loss'], color='orange', label='val_loss')
fig.suptitle('Loss', fontsize=20)
plt.legend(loc="upper left")
plt.show()
```

[10]:
```
fig = plt.figure()
plt.plot(hist.history['accuracy'], color='teal', label='accuracy')
plt.plot(hist.history['val_accuracy'], color='orange', label='val_accuracy')
```

```
fig.suptitle('Accuracy', fontsize=20)
plt.legend(loc="upper left")
plt.show()
```



[11]:
```
pre = Precision()
re = Recall()
acc = SparseCategoricalAccuracy()
```

[12]:
```
for batch in test_data.as_numpy_iterator():
    X, y = batch
    yhat = model.predict(X)

    yhat_classes = tf.argmax(yhat, axis=1)

    pre.update_state(y, yhat_classes)
    re.update_state(y, yhat_classes)
    acc.update_state(y, yhat)
```

```
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 18ms/step
```

```
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 191ms/step
```

[13]:
```
print(f"Precision: {pre.result().numpy() * 100 : .2f}%")
print(f"Recall: {re.result().numpy() * 100 : .2f}%")
print(f"Accuracy: {acc.result().numpy() * 100 : .2f}%")
```

```
Precision:  99.52%
Recall:  98.25%
Accuracy:  96.38%
```

[14]:
```
img = cv2.imread('Styles/test/Hatchback/8_jpg.rf.
  ↪c314c1d6777942876503fa1482c82240.jpg')

img_resized = cv2.resize(img, img_size)
img_expanded = np.expand_dims(img_resized, axis=0)

yhat = model.predict(img_expanded)
predicted_class = tf.argmax(yhat, axis=1).numpy()[0]

plt.imshow(img)
plt.title(f'Predicted class: {predicted_class}')
plt.axis('off')
plt.show()
```

```
1/1 [==============================] - 0s 192ms/step
```

[15]:
```
print(f'Predicted class is: {class_names[predicted_class]}')
for idx, prob in enumerate(yhat[0]):
    print(f"Model probability for {class_names[idx]} is {prob * 100:.2f}%")
```

```
Predicted class is: Hatchback
Model probability for Convertible is 0.00%
Model probability for Coupe is 0.00%
Model probability for Hatchback is 99.96%
Model probability for Pick-Up is 0.00%
Model probability for SUV is 0.02%
Model probability for Sedan is 0.00%
Model probability for VAN is 0.02%
```

[16]:
```
model_file_name = f"CarStyle{acc.result().numpy() * 100 : .2f}% MobileNet.h5"
model.save(os.path.join('models', model_file_name))
```

[ ]:
```

```