MobileNet Model

Action Classes - 10

In [1]:
```
from keras import models
from keras.layers import Dense,Flatten
from keras import backend as K
import numpy as np
import matplotlib.pyplot as plt

from keras.applications import mobilenet
```

In [2]:
```
import tensorflow as tf
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
```

```
Num GPUs Available:  1
2022-08-25 17:12:41.388990: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:41.416692: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:41.416840: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
```

Dataset

In [3]:
```
from keras.preprocessing.image import ImageDataGenerator

dataset_path = "./frames/"
# will contain the categories in respective folders

# Data generators
train_datagen = ImageDataGenerator(rescale=1/255, validation_split=0.2)
```

In [4]:
```
image_size = (224,224)
batch_size = 10

train_batches = train_datagen.flow_from_directory(
    dataset_path,
    target_size = image_size,
    batch_size = batch_size,
    class_mode = "categorical",
    subset = "training"
)

validation_batches = train_datagen.flow_from_directory(
    dataset_path,
    target_size = image_size,
    batch_size = batch_size,
    class_mode = "categorical",
    subset = "validation"
)

test_batches = train_datagen.flow_from_directory(
    dataset_path,
    target_size = image_size,
    batch_size = batch_size,
    class_mode = "categorical",
    subset = "validation"
)
```

```
Found 2734 images belonging to 10 classes.
Found 679 images belonging to 10 classes.
Found 679 images belonging to 10 classes.
```

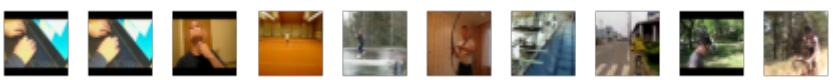
In [5]:
```
train_batches.class_indices
```

Out[5]:
```
{'ApplyLipstick': 0,
 'Archery': 1,
 'Biking': 2,
 'Diving': 3,
 'Kayaking': 4,
 'MilitaryParade': 5,
 'ShavingBeard': 6,
 'SkateBoarding': 7,
 'TennisSwing': 8,
 'Typing': 9}
```

In [6]:
```
from matplotlib import pyplot as plt

def plot_images(images_arr):
    fig, axes = plt.subplots(1,10)
    axes = axes.flatten()
    for img, ax in zip(images_arr, axes):
        ax.imshow(img)
        ax.axis('off')
    plt.tight_layout()
    plt.show()
```

In [7]:
```
imgs, labels = train_batches[0]
plot_images(imgs)
print(labels[:10])
```



```
[[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]
```

Initialize model

In [8]:
```
mobilenet_model_top = mobilenet.MobileNet(include_top=True,
                    input_shape=(224,224,3),
                    pooling='avg',
                    weights='imagenet')

for (i,layer) in enumerate(mobilenet_model_top.layers):
    print((i, layer.name, layer.output_shape))
```

```
2022-08-25 17:12:42.124998: I tensorflow/core/platform/cpu_feature_guard.
cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Netwo
rk Library (oneDNN) to use the following CPU instructions in performance-
critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropria
te compiler flags.
2022-08-25 17:12:42.125515: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.125690: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.125802: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.555082: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.555244: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.555365: I tensorflow/stream_executor/cuda/cuda_gpu_ex
ecutor.cc:975] successful NUMA node read from SysFS had negative value (-
1), but there must be at least one NUMA node, so returning NUMA node zero
2022-08-25 17:12:42.555467: I tensorflow/core/common_runtime/gpu/gpu_devi
ce.cc:1532] Created device /job:localhost/replica:0/task:0/device:GPU:0 w
ith 3368 MB memory:  -> device: 0, name: NVIDIA GeForce GTX 1050 Ti, pci
bus id: 0000:01:00.0, compute capability: 6.1
```

```
(0, 'input_1', [(None, 224, 224, 3)])
(1, 'conv1', (None, 112, 112, 32))
(2, 'conv1_bn', (None, 112, 112, 32))
(3, 'conv1_relu', (None, 112, 112, 32))
(4, 'conv_dw_1', (None, 112, 112, 32))
(5, 'conv_dw_1_bn', (None, 112, 112, 32))
(6, 'conv_dw_1_relu', (None, 112, 112, 32))
(7, 'conv_pw_1', (None, 112, 112, 64))
(8, 'conv_pw_1_bn', (None, 112, 112, 64))
(9, 'conv_pw_1_relu', (None, 112, 112, 64))
(10, 'conv_pad_2', (None, 113, 113, 64))
(11, 'conv_dw_2', (None, 56, 56, 64))
(12, 'conv_dw_2_bn', (None, 56, 56, 64))
(13, 'conv_dw_2_relu', (None, 56, 56, 64))
(14, 'conv_pw_2', (None, 56, 56, 128))
(15, 'conv_pw_2_bn', (None, 56, 56, 128))
(16, 'conv_pw_2_relu', (None, 56, 56, 128))
(17, 'conv_dw_3', (None, 56, 56, 128))
(18, 'conv_dw_3_bn', (None, 56, 56, 128))
(19, 'conv_dw_3_relu', (None, 56, 56, 128))
(20, 'conv_pw_3', (None, 56, 56, 128))
(21, 'conv_pw_3_bn', (None, 56, 56, 128))
(22, 'conv_pw_3_relu', (None, 56, 56, 128))
(23, 'conv_pad_4', (None, 57, 57, 128))
(24, 'conv_dw_4', (None, 28, 28, 128))
(25, 'conv_dw_4_bn', (None, 28, 28, 128))
(26, 'conv_dw_4_relu', (None, 28, 28, 128))
(27, 'conv_pw_4', (None, 28, 28, 256))
(28, 'conv_pw_4_bn', (None, 28, 28, 256))
(29, 'conv_pw_4_relu', (None, 28, 28, 256))
(30, 'conv_dw_5', (None, 28, 28, 256))
(31, 'conv_dw_5_bn', (None, 28, 28, 256))
(32, 'conv_dw_5_relu', (None, 28, 28, 256))
(33, 'conv_pw_5', (None, 28, 28, 256))
(34, 'conv_pw_5_bn', (None, 28, 28, 256))
(35, 'conv_pw_5_relu', (None, 28, 28, 256))
(36, 'conv_pad_6', (None, 29, 29, 256))
(37, 'conv_dw_6', (None, 14, 14, 256))
(38, 'conv_dw_6_bn', (None, 14, 14, 256))
(39, 'conv_dw_6_relu', (None, 14, 14, 256))
(40, 'conv_pw_6', (None, 14, 14, 512))
(41, 'conv_pw_6_bn', (None, 14, 14, 512))
(42, 'conv_pw_6_relu', (None, 14, 14, 512))
(43, 'conv_dw_7', (None, 14, 14, 512))
(44, 'conv_dw_7_bn', (None, 14, 14, 512))
(45, 'conv_dw_7_relu', (None, 14, 14, 512))
(46, 'conv_pw_7', (None, 14, 14, 512))
(47, 'conv_pw_7_bn', (None, 14, 14, 512))
(48, 'conv_pw_7_relu', (None, 14, 14, 512))
(49, 'conv_dw_8', (None, 14, 14, 512))
(50, 'conv_dw_8_bn', (None, 14, 14, 512))
(51, 'conv_dw_8_relu', (None, 14, 14, 512))
(52, 'conv_pw_8', (None, 14, 14, 512))
(53, 'conv_pw_8_bn', (None, 14, 14, 512))
(54, 'conv_pw_8_relu', (None, 14, 14, 512))
(55, 'conv_dw_9', (None, 14, 14, 512))
(56, 'conv_dw_9_bn', (None, 14, 14, 512))
(57, 'conv_dw_9_relu', (None, 14, 14, 512))
(58, 'conv_pw_9', (None, 14, 14, 512))
(59, 'conv_pw_9_bn', (None, 14, 14, 512))
(60, 'conv_pw_9_relu', (None, 14, 14, 512))
(61, 'conv_dw_10', (None, 14, 14, 512))
(62, 'conv_dw_10_bn', (None, 14, 14, 512))
```

```
(63, 'conv_dw_10_relu', (None, 14, 14, 512))
(64, 'conv_pw_10', (None, 14, 14, 512))
(65, 'conv_pw_10_bn', (None, 14, 14, 512))
(66, 'conv_pw_10_relu', (None, 14, 14, 512))
(67, 'conv_dw_11', (None, 14, 14, 512))
(68, 'conv_dw_11_bn', (None, 14, 14, 512))
(69, 'conv_dw_11_relu', (None, 14, 14, 512))
(70, 'conv_pw_11', (None, 14, 14, 512))
(71, 'conv_pw_11_bn', (None, 14, 14, 512))
(72, 'conv_pw_11_relu', (None, 14, 14, 512))
(73, 'conv_pad_12', (None, 15, 15, 512))
(74, 'conv_dw_12', (None, 7, 7, 512))
(75, 'conv_dw_12_bn', (None, 7, 7, 512))
(76, 'conv_dw_12_relu', (None, 7, 7, 512))
(77, 'conv_pw_12', (None, 7, 7, 1024))
(78, 'conv_pw_12_bn', (None, 7, 7, 1024))
(79, 'conv_pw_12_relu', (None, 7, 7, 1024))
(80, 'conv_dw_13', (None, 7, 7, 1024))
(81, 'conv_dw_13_bn', (None, 7, 7, 1024))
(82, 'conv_dw_13_relu', (None, 7, 7, 1024))
(83, 'conv_pw_13', (None, 7, 7, 1024))
(84, 'conv_pw_13_bn', (None, 7, 7, 1024))
(85, 'conv_pw_13_relu', (None, 7, 7, 1024))
(86, 'global_average_pooling2d', (None, 1, 1, 1024))
(87, 'dropout', (None, 1, 1, 1024))
(88, 'conv_preds', (None, 1, 1, 1000))
(89, 'reshape_2', (None, 1000))
(90, 'predictions', (None, 1000))
```

In [9]:

```
mobilenet_model = mobilenet.MobileNet(include_top=False,
                    input_shape=(224,224,3),
                    pooling='avg',classes=10,
                    weights='imagenet')

for (i,layer) in enumerate(mobilenet_model.layers):
    layer.trainable = False
    print((i, layer.name, layer.output_shape))
```

```
(0, 'input_2', [(None, 224, 224, 3)])
(1, 'conv1', (None, 112, 112, 32))
(2, 'conv1_bn', (None, 112, 112, 32))
(3, 'conv1_relu', (None, 112, 112, 32))
(4, 'conv_dw_1', (None, 112, 112, 32))
(5, 'conv_dw_1_bn', (None, 112, 112, 32))
(6, 'conv_dw_1_relu', (None, 112, 112, 32))
(7, 'conv_pw_1', (None, 112, 112, 64))
(8, 'conv_pw_1_bn', (None, 112, 112, 64))
(9, 'conv_pw_1_relu', (None, 112, 112, 64))
(10, 'conv_pad_2', (None, 113, 113, 64))
(11, 'conv_dw_2', (None, 56, 56, 64))
(12, 'conv_dw_2_bn', (None, 56, 56, 64))
(13, 'conv_dw_2_relu', (None, 56, 56, 64))
(14, 'conv_pw_2', (None, 56, 56, 128))
(15, 'conv_pw_2_bn', (None, 56, 56, 128))
(16, 'conv_pw_2_relu', (None, 56, 56, 128))
(17, 'conv_dw_3', (None, 56, 56, 128))
(18, 'conv_dw_3_bn', (None, 56, 56, 128))
(19, 'conv_dw_3_relu', (None, 56, 56, 128))
(20, 'conv_pw_3', (None, 56, 56, 128))
(21, 'conv_pw_3_bn', (None, 56, 56, 128))
(22, 'conv_pw_3_relu', (None, 56, 56, 128))
(23, 'conv_pad_4', (None, 57, 57, 128))
(24, 'conv_dw_4', (None, 28, 28, 128))
(25, 'conv_dw_4_bn', (None, 28, 28, 128))
(26, 'conv_dw_4_relu', (None, 28, 28, 128))
(27, 'conv_pw_4', (None, 28, 28, 256))
(28, 'conv_pw_4_bn', (None, 28, 28, 256))
(29, 'conv_pw_4_relu', (None, 28, 28, 256))
(30, 'conv_dw_5', (None, 28, 28, 256))
(31, 'conv_dw_5_bn', (None, 28, 28, 256))
(32, 'conv_dw_5_relu', (None, 28, 28, 256))
(33, 'conv_pw_5', (None, 28, 28, 256))
(34, 'conv_pw_5_bn', (None, 28, 28, 256))
(35, 'conv_pw_5_relu', (None, 28, 28, 256))
(36, 'conv_pad_6', (None, 29, 29, 256))
(37, 'conv_dw_6', (None, 14, 14, 256))
(38, 'conv_dw_6_bn', (None, 14, 14, 256))
(39, 'conv_dw_6_relu', (None, 14, 14, 256))
(40, 'conv_pw_6', (None, 14, 14, 512))
(41, 'conv_pw_6_bn', (None, 14, 14, 512))
(42, 'conv_pw_6_relu', (None, 14, 14, 512))
(43, 'conv_dw_7', (None, 14, 14, 512))
(44, 'conv_dw_7_bn', (None, 14, 14, 512))
(45, 'conv_dw_7_relu', (None, 14, 14, 512))
(46, 'conv_pw_7', (None, 14, 14, 512))
(47, 'conv_pw_7_bn', (None, 14, 14, 512))
(48, 'conv_pw_7_relu', (None, 14, 14, 512))
(49, 'conv_dw_8', (None, 14, 14, 512))
(50, 'conv_dw_8_bn', (None, 14, 14, 512))
(51, 'conv_dw_8_relu', (None, 14, 14, 512))
(52, 'conv_pw_8', (None, 14, 14, 512))
(53, 'conv_pw_8_bn', (None, 14, 14, 512))
(54, 'conv_pw_8_relu', (None, 14, 14, 512))
(55, 'conv_dw_9', (None, 14, 14, 512))
(56, 'conv_dw_9_bn', (None, 14, 14, 512))
(57, 'conv_dw_9_relu', (None, 14, 14, 512))
(58, 'conv_pw_9', (None, 14, 14, 512))
(59, 'conv_pw_9_bn', (None, 14, 14, 512))
(60, 'conv_pw_9_relu', (None, 14, 14, 512))
(61, 'conv_dw_10', (None, 14, 14, 512))
(62, 'conv_dw_10_bn', (None, 14, 14, 512))
```

```
(63, 'conv_dw_10_relu', (None, 14, 14, 512))
(64, 'conv_pw_10', (None, 14, 14, 512))
(65, 'conv_pw_10_bn', (None, 14, 14, 512))
(66, 'conv_pw_10_relu', (None, 14, 14, 512))
(67, 'conv_dw_11', (None, 14, 14, 512))
(68, 'conv_dw_11_bn', (None, 14, 14, 512))
(69, 'conv_dw_11_relu', (None, 14, 14, 512))
(70, 'conv_pw_11', (None, 14, 14, 512))
(71, 'conv_pw_11_bn', (None, 14, 14, 512))
(72, 'conv_pw_11_relu', (None, 14, 14, 512))
(73, 'conv_pad_12', (None, 15, 15, 512))
(74, 'conv_dw_12', (None, 7, 7, 512))
(75, 'conv_dw_12_bn', (None, 7, 7, 512))
(76, 'conv_dw_12_relu', (None, 7, 7, 512))
(77, 'conv_pw_12', (None, 7, 7, 1024))
(78, 'conv_pw_12_bn', (None, 7, 7, 1024))
(79, 'conv_pw_12_relu', (None, 7, 7, 1024))
(80, 'conv_dw_13', (None, 7, 7, 1024))
(81, 'conv_dw_13_bn', (None, 7, 7, 1024))
(82, 'conv_dw_13_relu', (None, 7, 7, 1024))
(83, 'conv_pw_13', (None, 7, 7, 1024))
(84, 'conv_pw_13_bn', (None, 7, 7, 1024))
(85, 'conv_pw_13_relu', (None, 7, 7, 1024))
(86, 'global_average_pooling2d_1', (None, 1024))
```

In [10]:

```
model = models.Sequential()

dense_layer_1 = Dense(32, activation='relu')
dense_layer_2 = Dense(32, activation='relu')
dense_layer_2 = Dense(32, activation='relu')
prediction_layer = Dense(10, activation='softmax')

model.add(mobilenet_model)
model.add(dense_layer_1)
model.add(dense_layer_2)
model.add(prediction_layer)

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 mobilenet_1.00_224 (Functio  (None, 1024)             3228864
 nal)

 dense (Dense)               (None, 32)                32800

 dense_2 (Dense)             (None, 32)                1056

 dense_3 (Dense)             (None, 10)                330

=================================================================
Total params: 3,263,050
Trainable params: 34,186
Non-trainable params: 3,228,864
_________________________________________________________________
```

In [11]:
```
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy'],
)
```

In [12]:
```
model.save("./models/action-class-10-model-mobilenet.h5")
```

In [13]:
```
fit = model.fit(train_batches, epochs=20, validation_data=validation_batc
```

```
Epoch 1/20
2022-08-25 17:12:46.073490: I tensorflow/stream_executor/cuda/cuda_dnn.c
c:384] Loaded cuDNN version 8401
2022-08-25 17:12:46.480355: I tensorflow/core/platform/default/subproces
s.cc:304] Start cannot spawn child process: No such file or directory
```

```
274/274 [==============================] - 13s 40ms/step - loss: 0.5004 -
accuracy: 0.8431 - val_loss: 0.2767 - val_accuracy: 0.8984
Epoch 2/20
274/274 [==============================] - 10s 36ms/step - loss: 0.0598 -
accuracy: 0.9850 - val_loss: 0.2467 - val_accuracy: 0.9116
Epoch 3/20
274/274 [==============================] - 10s 37ms/step - loss: 0.0182 -
accuracy: 0.9982 - val_loss: 0.2627 - val_accuracy: 0.8984
Epoch 4/20
274/274 [==============================] - 10s 36ms/step - loss: 0.0079 -
accuracy: 1.0000 - val_loss: 0.2292 - val_accuracy: 0.9175
Epoch 5/20
274/274 [==============================] - 10s 36ms/step - loss: 0.0036 -
accuracy: 1.0000 - val_loss: 0.2465 - val_accuracy: 0.9234
Epoch 6/20
274/274 [==============================] - 10s 36ms/step - loss: 0.0019 -
accuracy: 1.0000 - val_loss: 0.2329 - val_accuracy: 0.9219
Epoch 7/20
274/274 [==============================] - 10s 36ms/step - loss: 0.0014 -
accuracy: 1.0000 - val_loss: 0.2437 - val_accuracy: 0.9264
Epoch 8/20
274/274 [==============================] - 10s 36ms/step - loss: 8.6862e-
04 - accuracy: 1.0000 - val_loss: 0.2528 - val_accuracy: 0.9264
Epoch 9/20
274/274 [==============================] - 10s 36ms/step - loss: 6.8076e-
04 - accuracy: 1.0000 - val_loss: 0.2670 - val_accuracy: 0.9249
Epoch 10/20
274/274 [==============================] - 10s 37ms/step - loss: 5.2615e-
04 - accuracy: 1.0000 - val_loss: 0.2724 - val_accuracy: 0.9205
Epoch 11/20
274/274 [==============================] - 10s 37ms/step - loss: 4.1977e-
04 - accuracy: 1.0000 - val_loss: 0.2588 - val_accuracy: 0.9264
Epoch 12/20
274/274 [==============================] - 10s 37ms/step - loss: 3.3263e-
04 - accuracy: 1.0000 - val_loss: 0.2661 - val_accuracy: 0.9278
Epoch 13/20
274/274 [==============================] - 10s 38ms/step - loss: 2.6883e-
04 - accuracy: 1.0000 - val_loss: 0.2678 - val_accuracy: 0.9308
Epoch 14/20
274/274 [==============================] - 11s 39ms/step - loss: 2.1617e-
04 - accuracy: 1.0000 - val_loss: 0.2718 - val_accuracy: 0.9308
Epoch 15/20
274/274 [==============================] - 11s 39ms/step - loss: 1.7567e-
04 - accuracy: 1.0000 - val_loss: 0.2903 - val_accuracy: 0.9249
Epoch 16/20
274/274 [==============================] - 12s 43ms/step - loss: 1.4695e-
04 - accuracy: 1.0000 - val_loss: 0.2836 - val_accuracy: 0.9264
Epoch 17/20
274/274 [==============================] - 12s 45ms/step - loss: 1.2083e-
04 - accuracy: 1.0000 - val_loss: 0.2885 - val_accuracy: 0.9234
Epoch 18/20
274/274 [==============================] - 14s 49ms/step - loss: 9.9981e-
05 - accuracy: 1.0000 - val_loss: 0.2955 - val_accuracy: 0.9219
Epoch 19/20
274/274 [==============================] - 15s 56ms/step - loss: 8.2963e-
05 - accuracy: 1.0000 - val_loss: 0.2964 - val_accuracy: 0.9234
Epoch 20/20
274/274 [==============================] - 14s 50ms/step - loss: 6.8844e-
05 - accuracy: 1.0000 - val_loss: 0.2907 - val_accuracy: 0.9308
```

In [14]:
```
model.save("./models/action-class-10-trained-mobilenet.h5")
```

Evaluate and Predict

In [15]:
```
model = models.load_model("./models/action-class-10-trained-mobilenet.h5")
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 mobilenet_1.00_224 (Functio  (None, 1024)             3228864
 nal)

 dense (Dense)               (None, 32)                32800

 dense_2 (Dense)             (None, 32)                1056

 dense_3 (Dense)             (None, 10)                330

=================================================================
Total params: 3,263,050
Trainable params: 34,186
Non-trainable params: 3,228,864
_________________________________________________________________
```

In [16]:
```
model.evaluate(test_batches)
```

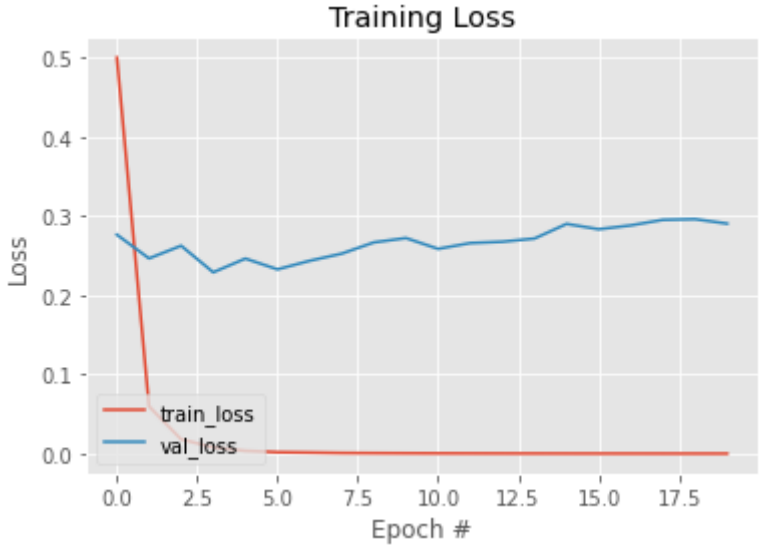
```
68/68 [==============================] - 5s 59ms/step - loss: 0.2907 - ac
curacy: 0.9308
```

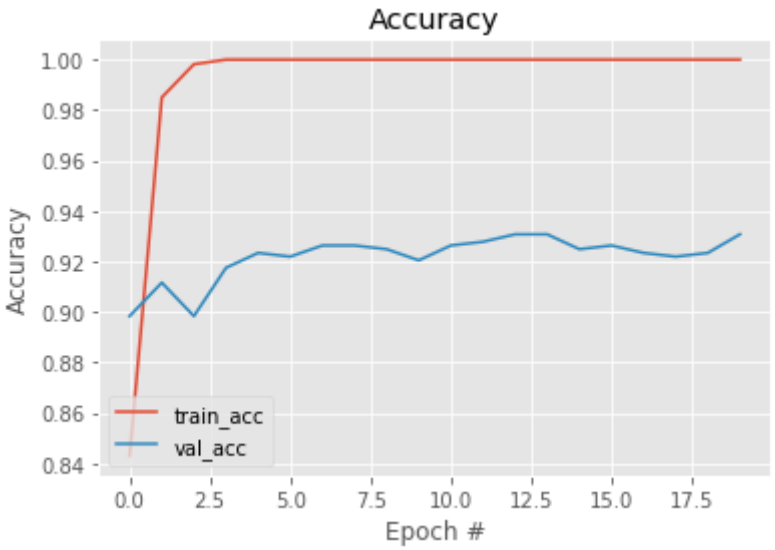
Out[16]:
```
[0.2907187044620514, 0.9307805299758911]
```

In [17]:
```
plt.style.use("ggplot")
plt.figure()

plt.plot(np.arange(0, 20), fit.history["loss"], label="train_loss")
plt.plot(np.arange(0, 20), fit.history["val_loss"], label="val_loss")
plt.title("Training Loss")
plt.xlabel("Epoch #")
plt.ylabel("Loss")
plt.legend(loc="lower left")
plt.show()

plt.plot(np.arange(0, 20), fit.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, 20), fit.history["val_accuracy"], label="val_acc")
plt.title("Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Accuracy")
plt.legend(loc="lower left")
plt.show()
```





In [18]:
```
print("Avg Val Acc: " + str(sum(fit.history["val_accuracy"])/20*100))
print("Avg Val Loss: " + str(sum(fit.history["val_loss"])/20*100))
```

```
Avg Val Acc: 92.17967540025711
Avg Val Loss: 26.701502278447155
```