

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
IMG_SIZE = (256, 256)
```

```
BATCH_SIZE = 32
```

```
train_datagen = ImageDataGenerator(rescale=1./255,
validation_split=0.2)
```

```
train_generator=train_datagen.flow_from_directory(
    '/content/drive/MyDrive/Mask Detection/data',
    target_size = IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    subset='training'
```

```
)
```

Found 483 images belonging to 3 classes.

```
val_generator=train_datagen.flow_from_directory(
    '/content/drive/MyDrive/Mask Detection/data',
    target_size = IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    subset='validation'
```

```
)
```

Found 120 images belonging to 3 classes.

```
Class_indices=train_generator.class_indices
```

```
Class_names=list(Class_indices.keys())
```

```
print("Class indices:",Class_indices)
```

```
print("Class names:",Class_names)
```

```
Class indices: {'Mask weared incorrect': 0, 'With mask': 1, 'Without
mask': 2}
```

```
Class names: ['Mask weared incorrect', 'With mask', 'Without mask']
```

```
model = keras.Sequential([
    layers.Conv2D(32, (3,3), activation='relu',
input_shape=(IMG_SIZE[0],IMG_SIZE[1],3)),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(128, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(3, activation='softmax')
```

```
])
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
```

```
    super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
model.summary()
```

```
Model: "sequential"
```

Layer (type) Param #	Output Shape	
conv2d (Conv2D) 896	(None, 254, 254, 32)	
max_pooling2d (MaxPooling2D) 0	(None, 127, 127, 32)	
conv2d_1 (Conv2D) 18,496	(None, 125, 125, 64)	
max_pooling2d_1 (MaxPooling2D) 0	(None, 62, 62, 64)	
conv2d_2 (Conv2D) 73,856	(None, 60, 60, 128)	
max_pooling2d_2 (MaxPooling2D) 0	(None, 30, 30, 128)	
flatten (Flatten) 0	(None, 115200)	
dense (Dense) 14,745,728	(None, 128)	

dense_1 (Dense)	(None, 3)	
387		

Total params: 14,839,363 (56.61 MB)

Trainable params: 14,839,363 (56.61 MB)

Non-trainable params: 0 (0.00 B)

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

```
model.fit(train_generator, epochs=5, validation_data=val_generator,
batch_size=BATCH_SIZE)
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/
data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset`
class should call `super().__init__(**kwargs)` in its constructor.
`**kwargs` can include `workers`, `use_multiprocessing`,
`max_queue_size`. Do not pass these arguments to `fit()`, as they will
be ignored.
```

```
self._warn_if_super_not_called()
```

Epoch 1/5

```
16/16 _____ 0s 7s/step - accuracy: 0.4766 - loss:
1.6959
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/
data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset`
class should call `super().__init__(**kwargs)` in its constructor.
`**kwargs` can include `workers`, `use_multiprocessing`,
`max_queue_size`. Do not pass these arguments to `fit()`, as they will
be ignored.
```

```
self._warn_if_super_not_called()
```

```
16/16 _____ 154s 9s/step - accuracy: 0.4790 - loss:
1.6692 - val_accuracy: 0.6500 - val_loss: 0.7010
```

Epoch 2/5

```
16/16 _____ 80s 5s/step - accuracy: 0.7500 - loss:
0.5983 - val_accuracy: 0.8167 - val_loss: 0.4156
```

Epoch 3/5

```
16/16 _____ 84s 5s/step - accuracy: 0.8108 - loss:
0.4582 - val_accuracy: 0.8667 - val_loss: 0.3302
```

Epoch 4/5

```
16/16 _____ 79s 5s/step - accuracy: 0.9377 - loss:
0.2164 - val_accuracy: 0.9083 - val_loss: 0.3050
```

Epoch 5/5

```
16/16 _____ 79s 5s/step - accuracy: 0.9425 - loss:
0.1778 - val_accuracy: 0.8833 - val_loss: 0.2553
```

```
<keras.src.callbacks.history.History at 0x7c234eec5c50>
```

```
model.save('/MyDrive/Mask Detection/model.h5')
```

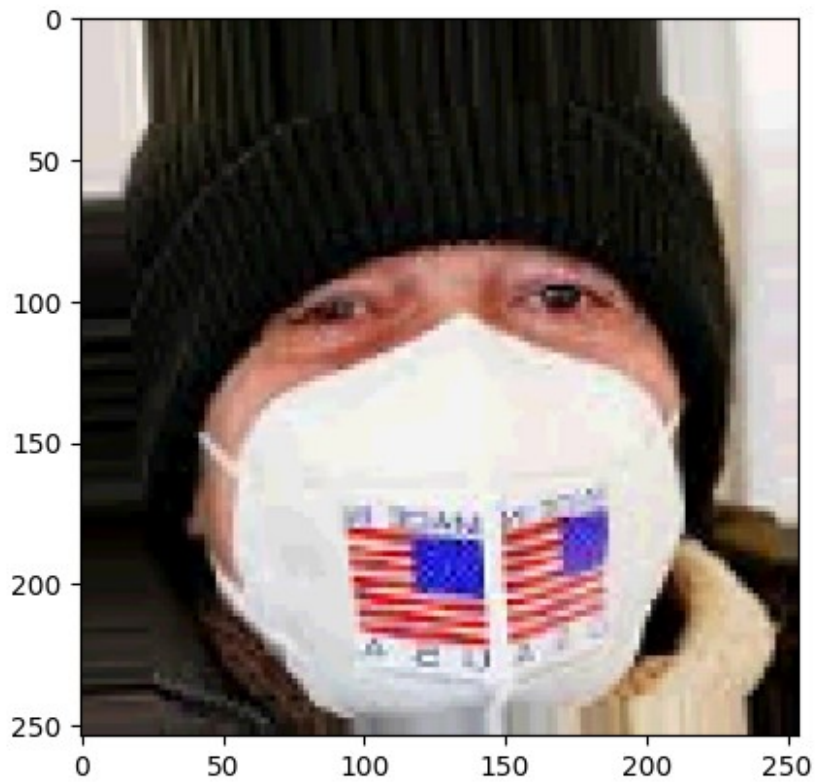
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
model = load_model('/MyDrive/Mask Detection/model.h5')
print("Model Loaded")
```

WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train or evaluate the model.

Model Loaded

```
test_image_path="/content/drive/MyDrive/Mask Detection/data/With
mask/101.png"
img = image.load_img(test_image_path, target_size=(254, 254))
plt.imshow(img)
plt.axis()
plt.show()
```



```
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array, axis=0)
img_array /=255.
```

```
prediction=model.predict(img_array)
ind=np.argmax(prediction)
print(Class_names[ind])
```

```
1/1 _____ 0s 236ms/step
With mask
```