```
import tensorflow as tf
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator

data_dir = "/content/drive/MyDrive/dataset"
image_size = (224, 224)
batch_size = 32
num_classes = 2  # Two classes: Suspicious activity and Non-suspicious activity

datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.2
)

train_generator = datagen.flow_from_directory(
    data_dir,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary',  # Binary classification
    subset='training'
)

validation_generator = datagen.flow_from_directory(
    data_dir,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary',
    subset='validation'
)


model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(1, activation='sigmoid')
])


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


epochs = 20
model.fit(train_generator, epochs=epochs, validation_data=validation_generator)

model.save("online_proctoring_model.h5")


     Found 257 images belonging to 2 classes.
     Found 63 images belonging to 2 classes.
     Epoch 1/20
     9/9 [==============================] - 39s 4s/step - loss: 0.8042 - accuracy: 0.7549 - val_loss: 0.4322 - val_accuracy: 0.8571
     Epoch 2/20
     9/9 [==============================] - 36s 4s/step - loss: 0.4337 - accuracy: 0.8482 - val_loss: 0.4338 - val_accuracy: 0.8571
     Epoch 3/20
     9/9 [==============================] - 35s 4s/step - loss: 0.4684 - accuracy: 0.8482 - val_loss: 0.4183 - val_accuracy: 0.8571
     Epoch 4/20
     9/9 [==============================] - 34s 4s/step - loss: 0.4549 - accuracy: 0.8482 - val_loss: 0.4768 - val_accuracy: 0.8571
     Epoch 5/20
     9/9 [==============================] - 35s 4s/step - loss: 0.4502 - accuracy: 0.8482 - val_loss: 0.6442 - val_accuracy: 0.8571
     Epoch 6/20
     9/9 [==============================] - 35s 4s/step - loss: 0.4731 - accuracy: 0.8482 - val_loss: 0.5360 - val_accuracy: 0.8571
     Epoch 7/20
     9/9 [==============================] - 34s 4s/step - loss: 0.5670 - accuracy: 0.8482 - val_loss: 0.4163 - val_accuracy: 0.8571
     Epoch 8/20
     9/9 [==============================] - 34s 4s/step - loss: 0.4394 - accuracy: 0.8482 - val_loss: 0.4263 - val_accuracy: 0.8571
     Epoch 9/20
     9/9 [==============================] - 37s 4s/step - loss: 0.4958 - accuracy: 0.8482 - val_loss: 0.4976 - val_accuracy: 0.8571
     Epoch 10/20
     9/9 [==============================] - 36s 4s/step - loss: 0.4763 - accuracy: 0.8482 - val_loss: 0.4125 - val_accuracy: 0.8571
     Epoch 11/20
     9/9 [==============================] - 34s 4s/step - loss: 0.4199 - accuracy: 0.8482 - val_loss: 0.4347 - val_accuracy: 0.8571
     Epoch 12/20
     9/9 [==============================] - 34s 4s/step - loss: 0.4204 - accuracy: 0.8482 - val_loss: 0.4105 - val_accuracy: 0.8571
     Epoch 13/20
     9/9 [==============================] - 36s 4s/step - loss: 0.4397 - accuracy: 0.8482 - val_loss: 0.4090 - val_accuracy: 0.8571
     Epoch 14/20
     9/9 [==============================] - 36s 4s/step - loss: 0.4370 - accuracy: 0.8482 - val_loss: 0.4051 - val_accuracy: 0.8571
     Epoch 15/20
     9/9 [==============================] - 34s 4s/step - loss: 0.4783 - accuracy: 0.8482 - val_loss: 0.4150 - val_accuracy: 0.8571
     Epoch 16/20
```

```
9/9 [==============================] - 34s 4s/step - loss: 0.4254 - accuracy: 0.8482 - val_loss: 0.4091 - val_accuracy: 0.8571
Epoch 17/20
9/9 [==============================] - 46s 5s/step - loss: 0.4237 - accuracy: 0.8482 - val_loss: 0.4114 - val_accuracy: 0.8571
Epoch 18/20
9/9 [==============================] - 37s 4s/step - loss: 0.4187 - accuracy: 0.8482 - val_loss: 0.4120 - val_accuracy: 0.8571
Epoch 19/20
9/9 [==============================] - 34s 4s/step - loss: 0.4252 - accuracy: 0.8482 - val_loss: 0.4157 - val_accuracy: 0.8571
Epoch 20/20
9/9 [==============================] - 34s 4s/step - loss: 0.4192 - accuracy: 0.8482 - val_loss: 0.4200 - val_accuracy: 0.8571
```

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report


loaded_model = tf.keras.models.load_model("/content/online_proctoring_model.h5")


test_data_dir = "/content/drive/MyDrive/dataset/validation"


test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary',
    shuffle=False
)


test_deep_features = loaded_model.predict(test_generator)


test_labels = test_generator.classes


# SVM classification
svm_classifier = SVC(kernel='linear')
svm_classifier.fit(test_deep_features, test_labels)
y_pred_svm = svm_classifier.predict(test_deep_features)
accuracy_svm = accuracy_score(test_labels, y_pred_svm)
print("Accuracy with SVM:", accuracy_svm)
print(classification_report(test_labels, y_pred_svm))

# KNN classification
knn_classifier = KNeighborsClassifier(n_neighbors=5)
knn_classifier.fit(test_deep_features, test_labels)
y_pred_knn = knn_classifier.predict(test_deep_features)
accuracy_knn = accuracy_score(test_labels, y_pred_knn)
print("Accuracy with KNN:", accuracy_knn)
print(classification_report(test_labels, y_pred_knn))
```

```
Found 48 images belonging to 2 classes.
2/2 [==============================] - 1s 415ms/step
Accuracy with SVM: 0.7291666666666666
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        13
           1       0.73      1.00      0.84        35

    accuracy                           0.73        48
   macro avg       0.36      0.50      0.42        48
weighted avg       0.53      0.73      0.61        48

Accuracy with KNN: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        35

    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and
  _warn_prf(average, modifier, msg_start, len(result))
```