Traing Different Model for classification of Cat and Dog

Name : Prathamesh Nale

Roll No : DS13

prn : 2122000107

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import EarlyStopping


# Path to the dataset
train_dir = "/content/drive/MyDrive/Colab Notebooks/DL/cat_dog_dataset/train"
validation_dir = "/content/drive/MyDrive/Colab Notebooks/DL/cat_dog_dataset/test"


IMG_SIZE = (224, 224)
BATCH_SIZE = 32
EPOCHS = 5


train_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode="nearest",
)


validation_datagen = ImageDataGenerator(rescale=1.0 / 255)


train_generator = train_datagen.flow_from_directory(
    train_dir, target_size=IMG_SIZE, batch_size=BATCH_SIZE, class_mode="binary"
)
```

```
Found 568 images belonging to 2 classes.
```

```
validation_generator = validation_datagen.flow_from_directory(
    validation_dir, target_size=IMG_SIZE, batch_size=BATCH_SIZE, class_mode="binary"
)
```

```
Found 140 images belonging to 2 classes.
```

Traning Densnet121

```
from tensorflow.keras.applications import DenseNet121


from tensorflow.keras.applications import DenseNet121

def train_densenet():
    base_model = DenseNet121(include_top=False, weights="imagenet", input_shape=(224, 224, 3))
    base_model.trainable = False

    # Custom layers
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dropout(0.5)(x)
    x = Dense(256, activation="relu")(x)
    x = Dropout(0.5)(x)
    predictions = Dense(1, activation="sigmoid")(x)
```

```
model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])

early_stopping = EarlyStopping(monitor="val_loss", patience=5, restore_best_weights=True)

print("Training DenseNet121...")
model.fit(
    train_generator,
    validation_data=validation_generator,
    epochs=EPOCHS,
    callbacks=[early_stopping],
    verbose=1,
)
model.save("DenseNet121_cats_dogs.h5")
print("DenseNet121 model saved!")
```

```
train_densenet()
```

```
Training DenseNet121...
Epoch 1/5
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:122: UserWarning: Your `PyDataset` class
  self._warn_if_super_not_called()
18/18 ━━━━━━━━━━━━━━━━━━━━ 236s 10s/step - accuracy: 0.6349 - loss: 0.7879 - val_accuracy: 0.9071 - val_loss: 0.1961
Epoch 2/5
18/18 ━━━━━━━━━━━━━━━━━━━━ 187s 8s/step - accuracy: 0.8761 - loss: 0.2864 - val_accuracy: 0.9214 - val_loss: 0.1666
Epoch 3/5
18/18 ━━━━━━━━━━━━━━━━━━━━ 194s 7s/step - accuracy: 0.9357 - loss: 0.1519 - val_accuracy: 0.9571 - val_loss: 0.1221
Epoch 4/5
18/18 ━━━━━━━━━━━━━━━━━━━━ 158s 9s/step - accuracy: 0.9124 - loss: 0.2081 - val_accuracy: 0.9500 - val_loss: 0.1443
Epoch 5/5
18/18 ━━━━━━━━━━━━━━━━━━━━ 147s 8s/step - accuracy: 0.9509 - loss: 0.1206 - val_accuracy: 0.9500 - val_loss: 0.1329
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is consi
DenseNet121 model saved!
```

Start coding or generate with AI.

Traning VGG16

```
from tensorflow.keras.applications import VGG16
```

```
def train_vgg16():
    base_model = VGG16(include_top=False, weights="imagenet", input_shape=(224, 224, 3))
    base_model.trainable = False

    # Custom layers
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dropout(0.5)(x)
    x = Dense(256, activation="relu")(x)
    x = Dropout(0.5)(x)
    predictions = Dense(1, activation="sigmoid")(x)

    model = Model(inputs=base_model.input, outputs=predictions)
    model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])

    early_stopping = EarlyStopping(monitor="val_loss", patience=5, restore_best_weights=True)

    print("Training VGG16...")
    model.fit(
        train_generator,
        validation_data=validation_generator,
        epochs=3,
        callbacks=[early_stopping],
        verbose=1,
    )
    model.save("VGG16_cats_dogs.h5")
    print("VGG16 model saved!")
```

```
train_vgg16()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.
58889256/58889256 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Training VGG16...
Epoch 1/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 518s 29s/step - accuracy: 0.5301 - loss: 0.8260 - val_accuracy: 0.7429 - val_loss: 0.6188
Epoch 2/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 511s 26s/step - accuracy: 0.6132 - loss: 0.6835 - val_accuracy: 0.7429 - val_loss: 0.5813
Epoch 3/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 504s 26s/step - accuracy: 0.6127 - loss: 0.6547 - val_accuracy: 0.6643 - val_loss: 0.5932
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is consi
VGG16 model saved!
```

Start coding or generate with AI.

Traning VGG19

```
from tensorflow.keras.applications import VGG19

def train_vgg19():
    base_model = VGG19(include_top=False, weights="imagenet", input_shape=(224, 224, 3))
    base_model.trainable = False

    # Custom layers
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dropout(0.5)(x)
    x = Dense(256, activation="relu")(x)
    x = Dropout(0.5)(x)
    predictions = Dense(1, activation="sigmoid")(x)

    model = Model(inputs=base_model.input, outputs=predictions)
    model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])

    early_stopping = EarlyStopping(monitor="val_loss", patience=5, restore_best_weights=True)

    print("Training VGG19...")
    model.fit(
        train_generator,
        validation_data=validation_generator,
```

```
        epochs=3,
        callbacks=[early_stopping],
        verbose=1,
    )
    model.save("VGG19_cats_dogs.h5")
    print("VGG19 model saved!")
```

```
train_vgg19()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weights_tf_dim_ordering_tf_kernels_notop.
80134624/80134624 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
Training VGG19...
Epoch 1/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 588s 33s/step - accuracy: 0.5125 - loss: 0.7781 - val_accuracy: 0.7571 - val_loss: 0.6149
Epoch 2/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 617s 32s/step - accuracy: 0.5856 - loss: 0.6893 - val_accuracy: 0.6214 - val_loss: 0.6021
Epoch 3/3
18/18 ━━━━━━━━━━━━━━━━━━━━ 582s 32s/step - accuracy: 0.6749 - loss: 0.6383 - val_accuracy: 0.7429 - val_loss: 0.5423
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is consi
VGG19 model saved!
```

Start coding or generate with AI.

Traning InceptionResNetV2

```
from tensorflow.keras.applications import InceptionResNetV2
```

```
def train_inceptionresnetv2():
    base_model = InceptionResNetV2(include_top=False, weights="imagenet", input_shape=(224, 224, 3))
    base_model.trainable = False

    # Custom layers
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dropout(0.5)(x)
    x = Dense(256, activation="relu")(x)
    x = Dropout(0.5)(x)
    predictions = Dense(1, activation="sigmoid")(x)

    model = Model(inputs=base_model.input, outputs=predictions)
    model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])

    early_stopping = EarlyStopping(monitor="val_loss", patience=5, restore_best_weights=True)

    print("Training InceptionResNetV2...")
    model.fit(
        train_generator,
        validation_data=validation_generator,
        epochs=2,
        callbacks=[early_stopping],
        verbose=1,
    )
    model.save("InceptionResNetV2_cats_dogs.h5")
    print("InceptionResNetV2 model saved!")
```

```
train_inceptionresnetv2()
```