2347126-lab5-nndl

October 24, 2024

[3]:
```
import kagglehub

# Download latest version
path = kagglehub.dataset_download("puneet6060/intel-image-classification")

print("Path to dataset files:", path)
```

```
Downloading from
https://www.kaggle.com/api/v1/datasets/download/puneet6060/intel-image-
classification?dataset_version_number=2…

100%|      | 346M/346M [00:01<00:00, 193MB/s]

Extracting files…


Path to dataset files: /root/.cache/kagglehub/datasets/puneet6060/intel-image-
classification/versions/2
```

[7]:
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Define image dimensions and parameters
image_height, image_width = 150, 150  # Adjust as per your dataset
channels = 3  # 3 for RGB images
batch_size = 32
n_classes = 6  # Adjust this to the number of classes in your dataset

# Data augmentation and rescaling for training and validation datasets
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
```

```
    fill_mode='nearest'
)

val_datagen = ImageDataGenerator(rescale=1./255)

# Create training data generator
train_generator = train_datagen.flow_from_directory(
    '/root/.cache/kagglehub/datasets/puneet6060/intel-image-classification',
    target_size=(image_height, image_width),
    batch_size=batch_size,
    class_mode='sparse'  # Use 'sparse' for integer labels
)

# Create validation data generator
val_generator = val_datagen.flow_from_directory(
    '/root/.cache/kagglehub/datasets/puneet6060/intel-image-classification',
    target_size=(image_height, image_width),
    batch_size=batch_size,
    class_mode='sparse'  # Use 'sparse' for integer labels
)

# Build the model
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(image_height,␣
 ↪image_width, channels)),
    MaxPooling2D(pool_size=(2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(n_classes, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',␣
 ↪metrics=['accuracy'])

# Train the model
history = model.fit(train_generator, epochs=1, validation_data=val_generator)

# Evaluate the model on validation data
val_loss, val_accuracy = model.evaluate(val_generator)
print(f'Validation Loss: {val_loss}, Validation Accuracy: {val_accuracy}')
```

```
Found 24335 images belonging to 1 classes.
```

```
Found 24335 images belonging to 1 classes.
761/761 [==============================] - 192s 250ms/step - loss: 0.0024 -
accuracy: 0.9987 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
761/761 [==============================] - 32s 42ms/step - loss: 0.0000e+00 -
accuracy: 1.0000
Validation Loss: 0.0, Validation Accuracy: 1.0
```