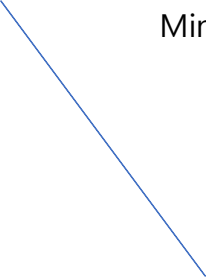




산업인공지능개론

Mini Project #3



2023254006

이선경

CONTENS

01. 분류 데이터

02. 코드

03. 결과

01. 분류데이터

- 5개 부류 사용
- 부류별 학습데이터 : 드론 - 66, 비행기 - 52, 열기구 - 56, 우주선 - 60, 헬리콥터 - 65
- 총 데이터 297ea

드론	비행기	열기구	우주선	헬리콥터
				

02. 코드

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

from google.colab import drive
drive.mount('/content/drive')

Drone_path = '/content/drive/MyDrive/CNN'
Plane_path = '/content/drive/MyDrive/CNN/Plane'

# 이미지 크기와 배치 크기 설정
image_size = (64, 64)
batch_size = 32

# 데이터 생성기 설정
data_generator = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    dtype='float64'
)

# 훈련 데이터 생성기 설정
train_generator = data_generator.flow_from_directory(
    Drone_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='categorical'
)

# 검증 데이터 생성기 설정
validation_generator = data_generator.flow_from_directory(
    Drone_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='categorical'
)
```

```
print(train_generator.class_indices)

from PIL import Image

image = Image.open('/content/drive/MyDrive/CNN/Drone/drone1.jpg')
image.show() # 이미지가 정상적으로 불러와지는지 확인합니다.

# CNN 모델 정의
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(image_size[0], image_size[1], 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(5, activation='softmax'))

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

# 모델 훈련
model.fit(train_generator,
          steps_per_epoch=train_generator.samples // batch_size,
          epochs=100,
          validation_data=validation_generator,
          validation_steps=validation_generator.samples // batch_size)
```

03. 결과

이미지 수, 클래스 수, 이미지 데이터 확인



03. 결과

전체 훈련 데이터 : 100

```
Epoch 88/100
9/9 [=====] - 11s 1s/step - loss: 0.0070 - accuracy: 0.9925 - val_loss: 0.0067 - val_accuracy: 0.9931
Epoch 89/100
9/9 [=====] - 12s 1s/step - loss: 0.0103 - accuracy: 0.9849 - val_loss: 0.0066 - val_accuracy: 0.9931
Epoch 90/100
9/9 [=====] - 10s 1s/step - loss: 0.0091 - accuracy: 0.9925 - val_loss: 0.0070 - val_accuracy: 0.9931
Epoch 91/100
9/9 [=====] - 12s 1s/step - loss: 0.0077 - accuracy: 0.9925 - val_loss: 0.0080 - val_accuracy: 0.9931
Epoch 92/100
9/9 [=====] - 12s 1s/step - loss: 0.0106 - accuracy: 0.9887 - val_loss: 0.0110 - val_accuracy: 0.9896
Epoch 93/100
9/9 [=====] - 11s 1s/step - loss: 0.0151 - accuracy: 0.9896 - val_loss: 0.0080 - val_accuracy: 0.9931
Epoch 94/100
9/9 [=====] - 10s 1s/step - loss: 0.0108 - accuracy: 0.9925 - val_loss: 0.0082 - val_accuracy: 0.9931
Epoch 95/100
9/9 [=====] - 11s 1s/step - loss: 0.0062 - accuracy: 0.9962 - val_loss: 0.0070 - val_accuracy: 0.9931
Epoch 96/100
9/9 [=====] - 15s 2s/step - loss: 0.0071 - accuracy: 0.9925 - val_loss: 0.0086 - val_accuracy: 0.9931
Epoch 97/100
9/9 [=====] - 11s 1s/step - loss: 0.0087 - accuracy: 0.9887 - val_loss: 0.0096 - val_accuracy: 0.9931
Epoch 98/100
9/9 [=====] - 11s 1s/step - loss: 0.0063 - accuracy: 0.9887 - val_loss: 0.0057 - val_accuracy: 0.9931
Epoch 99/100
9/9 [=====] - 16s 2s/step - loss: 0.0070 - accuracy: 0.9887 - val_loss: 0.0082 - val_accuracy: 0.9931
Epoch 100/100
9/9 [=====] - 11s 1s/step - loss: 0.0072 - accuracy: 0.9925 - val_loss: 0.0057 - val_accuracy: 0.9931
<keras.callbacks.History at 0x7fed1d9ccd0>
```