250729

- 4주차 - 1
    - opencv 활용하여 GUI 만들기(그림판, 저장, 초기화 버튼, 저장한 모델 사용해서 정답 출력하기)

```
# 모델 학습 및 평가 완료 후
# Google Drive 연결
from google.colab import drive
drive.mount('/content/drive')

# 필요한 라이브러리 불러오기
import tensorflow as tf
import glob
import numpy as np
import os
import matplotlib.pyplot as plt

# O/X 데이터 경로
O_DIR = '/content/DATA/O'
X_DIR = '/content/DATA/X'
IMAGE_SIZE = (28, 28)
CLASS_NAMES = ['O', 'X']

# 이미지 전처리 함수 재정의
def load_and_preprocess_image(path):
    image = tf.io.read_file(path)
    image = tf.image.decode_png(image, channels=1)
    image = tf.image.resize(image, IMAGE_SIZE)
    image = tf.cast(image, tf.float32) / 255.0
    return image


# 원래 저장했던 모델 불러오기 (Drive 경로 확인 필요)
```

```
drive_model_path = "/content/drive/MyDrive/my_ox_model.h5"
model = tf.keras.models.load_model(drive_model_path)
print(" 기존 모델 불러오기 성공")

# 필요시 다시 컴파일 (metrics 세팅)
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=T
rue),
              metrics=['accuracy'])

# (경고 없애기 위해 metrics 초기화)
_ = model.evaluate(test_ds, verbose=0)

# Colab 로컬에 다시 저장 (선택사항)
model_path = "/content/my_ox_model.h5"
model.save(model_path)
print(f" Colab 로컬에도 모델 저장 완료 : {model_path}")

# 불러온 모델로 예측 테스트 (O 이미지)
test_image_path = glob.glob('/content/DATA/O/*.png')[0]
test_image = load_and_preprocess_image(test_image_path)
test_batch = tf.expand_dims(test_image, 0)

pred_logits = model.predict(test_batch)
pred_probs = tf.nn.softmax(pred_logits)
pred_class = CLASS_NAMES[np.argmax(pred_probs)]
print(f" 불러온 모델 예측 결과 (O 이미지) : {pred_class}")

# 불러온 모델로 예측 테스트 (X 이미지)
test_image_path = glob.glob('/content/DATA/X/*.png')[0]
test_image = load_and_preprocess_image(test_image_path)
test_batch = tf.expand_dims(test_image, 0)

pred_logits = model.predict(test_batch)
pred_probs = tf.nn.softmax(pred_logits)
pred_class = CLASS_NAMES[np.argmax(pred_probs)]
print(f" 불러온 모델 예측 결과 (X 이미지) : {pred_class}")
```

```
WARNING:absl:Compiled the loaded model, but the compiled met
Drive already mounted at /content/drive; to attempt to forci
 기존 모델 불러오기 성공!
WARNING:absl:You are saving your model as an HDF5 file via `
 Colab 로컬에도 모델 저장 완료: /content/my_ox_model.h5
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 116ms/step
 불러온 모델 예측 결과 (O 이미지): O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step
 불러온 모델 예측 결과 (X 이미지): X
```

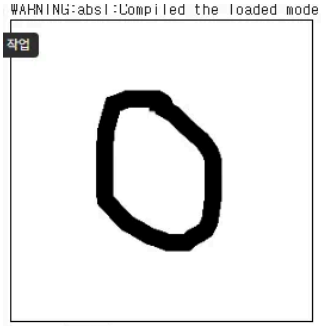
```
# IPython과 Colab 환경에서 UI 및 JavaScript 연동을 위해 필요한 라이브러리 불러오기
from IPython.display import display, Javascript, HTML
import ipywidgets as widgets

# TensorFlow 및 이미지 처리 관련 라이브러리 불러오기
import tensorflow as tf
import numpy as np
from PIL import Image
import io
import base64

# 학습된 모델 경로 지정
MODEL_PATH = "/content/my_ox_model.h5"

# 모델 불러오기 (O/X 분류 모델)
model = tf.keras.models.load_model(MODEL_PATH)

# 예측 결과 클래스 이름 정의
CLASS_NAMES = ['O', 'X']

# HTML + JavaScript로 그리기 캔버스 UI 생성
canvas_html = """
<canvas id="canvas" width=280 height=280 style="border:2px solid blac
k; background-color: white;"></canvas><br>
<button onclick="clearCanvas()">초기</button>
<button onclick="saveCanvas()">예측</button>
<script>
var canvas = document.getElementById('canvas'); // 캔버스 엘리먼트 가져오
기
```

```
var ctx = canvas.getContext('2d'); // 2D 드로잉 컨텍스트
ctx.lineWidth = 15; // 선 굵기 설정
ctx.lineCap = 'round'; // 선 끝 모양을 둥글게 설정
ctx.strokeStyle = 'black'; // 선 색상 설정

var drawing = false; // 마우스 드래그 상태 여부

// 마우스 클릭 시 그리기 시작
canvas.addEventListener('mousedown', function(e) {
    drawing = true;
    ctx.beginPath();
    ctx.moveTo(e.offsetX, e.offsetY);
});

// 마우스 이동 시 선 그리기
canvas.addEventListener('mousemove', function(e) {
    if (drawing) {
        ctx.lineTo(e.offsetX, e.offsetY);
        ctx.stroke();
    }
});

// 마우스 버튼 뗄 때 그리기 종료
canvas.addEventListener('mouseup', function(e) {
    drawing = false;
});

// 캔버스 초기화 함수
function clearCanvas() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
    ctx.fillStyle = "white"; // 배경색을 흰색으로 채우기
    ctx.fillRect(0, 0, canvas.width, canvas.height);
}

// 캔버스 이미지 저장 후 Python 함수 호출 (예측 요청)
function saveCanvas() {
    var dataURL = canvas.toDataURL('image/png'); // PNG 데이터 URL 추출
    google.colab.kernel.invokeFunction('notebook.predict_image', [dataUR
```

```
L], {}); // Python 콜백 호출
}
</script>
"""

# Colab에서 HTML 캔버스 표시
display(HTML(canvas_html))

# Colab에서 JS → Python 함수 연결
def predict_image(data_url):
    # data_url에서 base64 이미지 데이터 분리
    header, encoded = data_url.split(",", 1)
    data = base64.b64decode(encoded)

    # 이미지를 PIL로 로드 후 흑백 변환 및 28x28 크기로 리사이즈
    img = Image.open(io.BytesIO(data)).convert("L").resize((28, 28))

    # NumPy 배열로 변환 및 0~1 범위로 정규화
    img_array = np.array(img) / 255.0
    img_array = img_array.reshape(1, 28, 28, 1)  # 모델 입력 형식 맞추기 (배치, 높이, 너비, 채널)

    # 모델 예측 수행
    pred = model.predict(img_array)
    pred_class = CLASS_NAMES[np.argmax(pred)]  # 확률이 가장 높은 클래스 선택

    # 예측 결과 출력
    print(f"예측 결과 : {pred_class}")

# Colab의 JavaScript에서 Python 함수 호출을 등록
from google.colab import output
output.register_callback('notebook.predict_image', predict_image)
```

WARNING:absl:Compiled the loaded model, but the compiled me

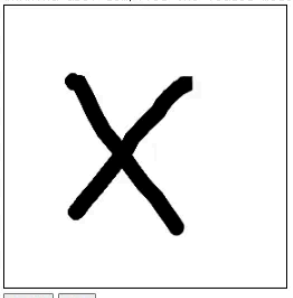
작업



초기화 예측

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 104ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step
예측 결과 : O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 56ms/step
예측 결과 : O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 60ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
예측 결과 : O

WARNING:absl:Compiled the loaded model, but the compiled me



초기화 예측

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 104ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step
예측 결과 : O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 56ms/step
예측 결과 : O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 60ms/step
예측 결과 : X
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
예측 결과 : O
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 54ms/step
예측 결과 : X