Assignment 04

MNIST CNN

2021270660 이지원

CODE DOCUMENTATION

```
# import libraries and modules
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import cv2
import matplotlib.pyplot as plt
```

[6] ✓ 0.0s Python

CODE DOCUMENTATION

```python
# load mnist data
(train_images, train_labels), (test_images, test_labels) = datasets.mnist.load_data()
train_images = train_images.reshape((60000, 28, 28, 1)) # change shape
test_images = test_images.reshape((10000, 28, 28, 1)) # change shape

# normalize pixel to 0~1
train_images, test_images = train_images / 255.0, test_images / 255.0

# build model
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5)
```

[9] ✓ 2m 16.1s Python

```
Epoch 1/5
1875/1875 [==============================] - 28s 14ms/step - loss: 0.1470 - accuracy: 0.9545
Epoch 2/5
1875/1875 [==============================] - 39s 21ms/step - loss: 0.0464 - accuracy: 0.9848
Epoch 3/5
1875/1875 [==============================] - 27s 14ms/step - loss: 0.0331 - accuracy: 0.9897
Epoch 4/5
1875/1875 [==============================] - 23s 12ms/step - loss: 0.0261 - accuracy: 0.9919
Epoch 5/5
1875/1875 [==============================] - 19s 10ms/step - loss: 0.0209 - accuracy: 0.9930

<keras.callbacks.History at 0x7fa7c13dc9d0>
```

Week 11 교안과 동일한 구조로 CNN 생성

CODE DOCUMENTATION

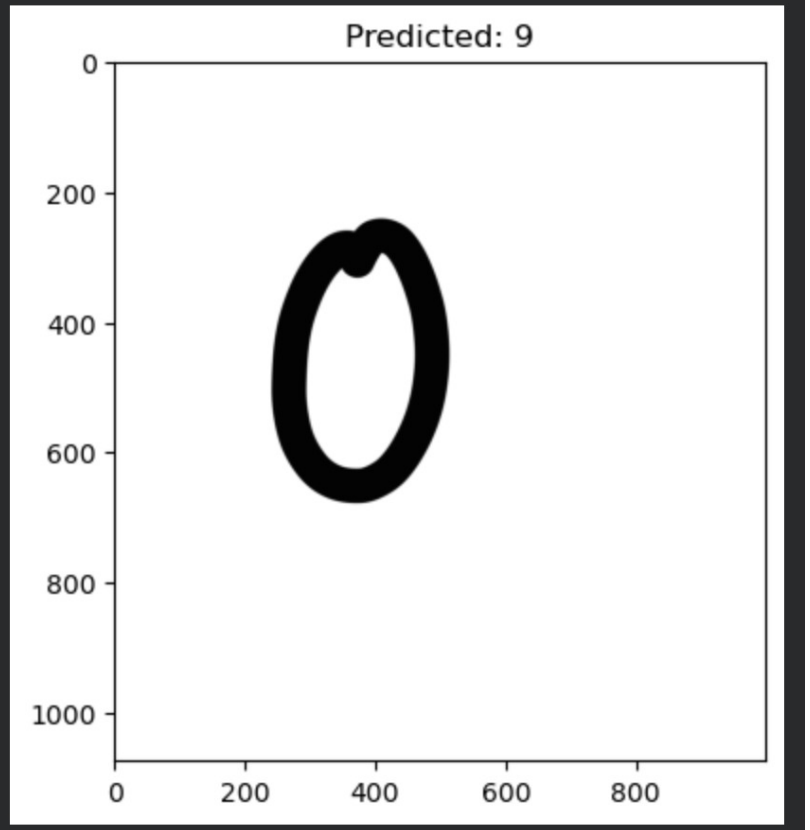
```
img_path = './6.jpeg'  # 직접 쓴 숫자 이미지 파일 경로

image = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
image = cv2.resize(image, (28, 28))
image = image.astype('float32')
image = image.reshape(1, 784)
image = 255-image
image /= 255.0

pred = model.predict(image.reshape(1, 28,28,1), batch_size=1)
print("predicted number=", pred.argmax())

img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
plt.imshow(img, cmap='gray')
plt.title(f'Predicted: {pred.argmax()}')
plt.show()
```

✓ 0.2s Python

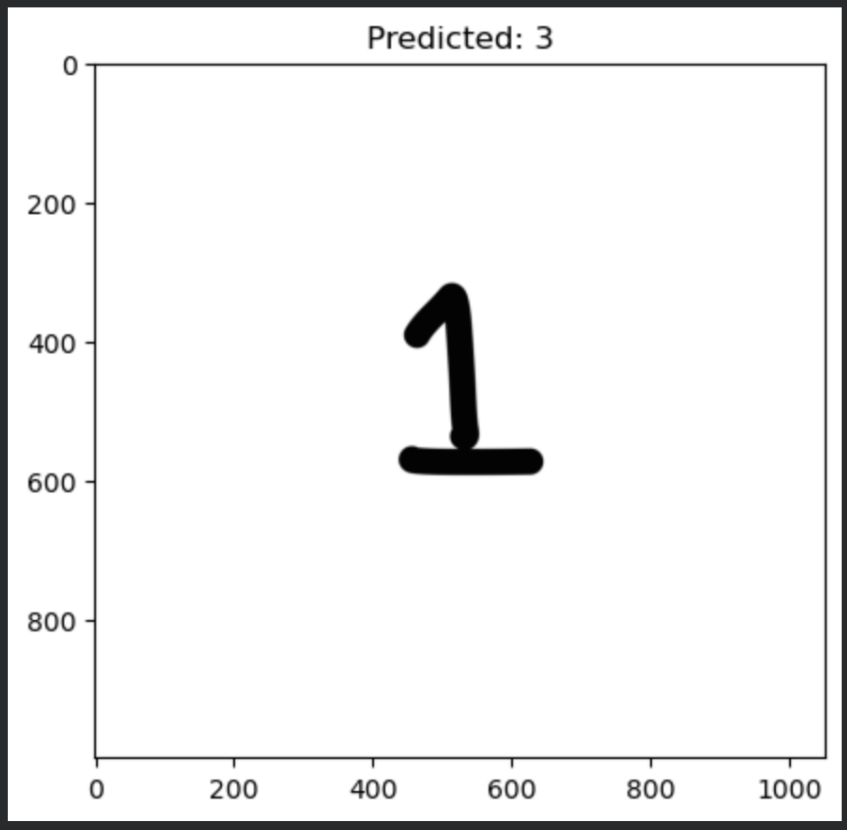
Img_path를 변경하며 결과 확인

RESULT

```
1/1 [==============================] - 0s 43ms/step
predicted number= 9
```

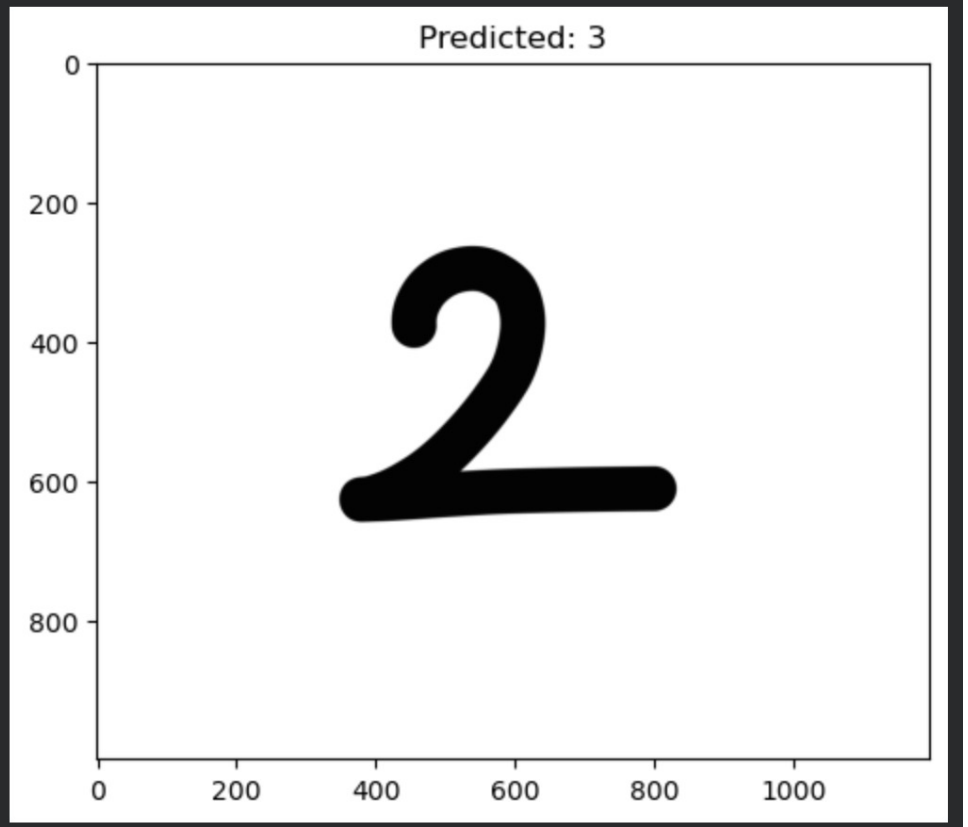



```
1/1 [==============================] - 0s 73ms/step
predicted number= 3
```

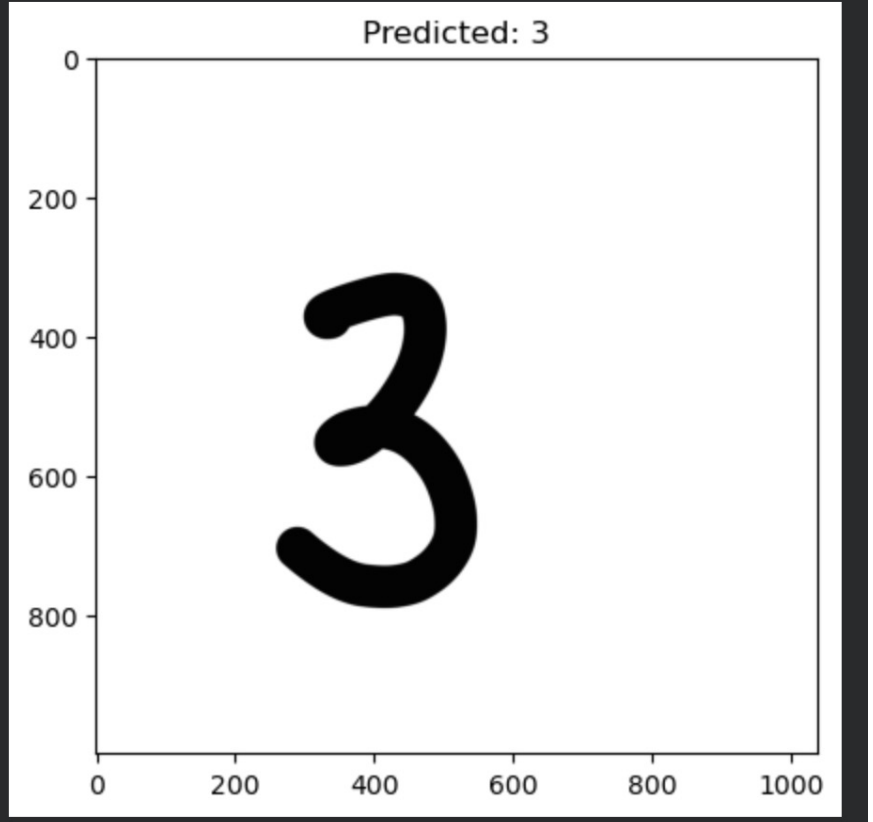


RESULT

```
1/1 [==============================] - 0s 46ms/step
predicted number= 3
```

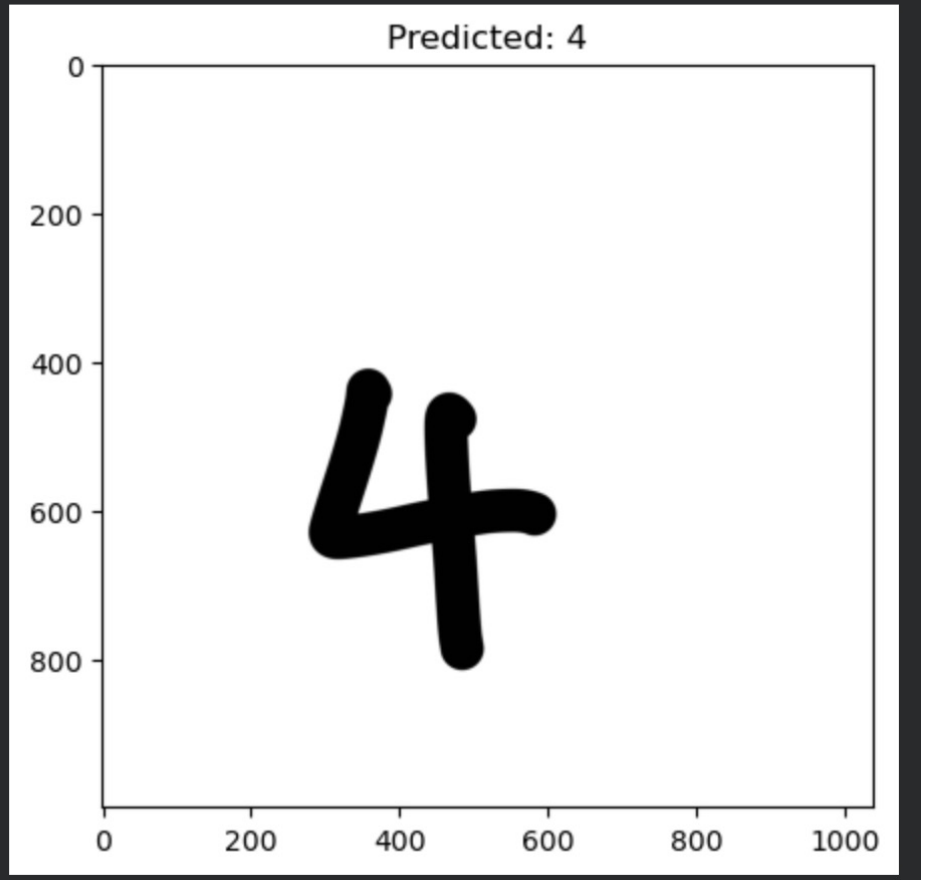



```
1/1 [==============================] - 0s 26ms/step
predicted number= 3
```

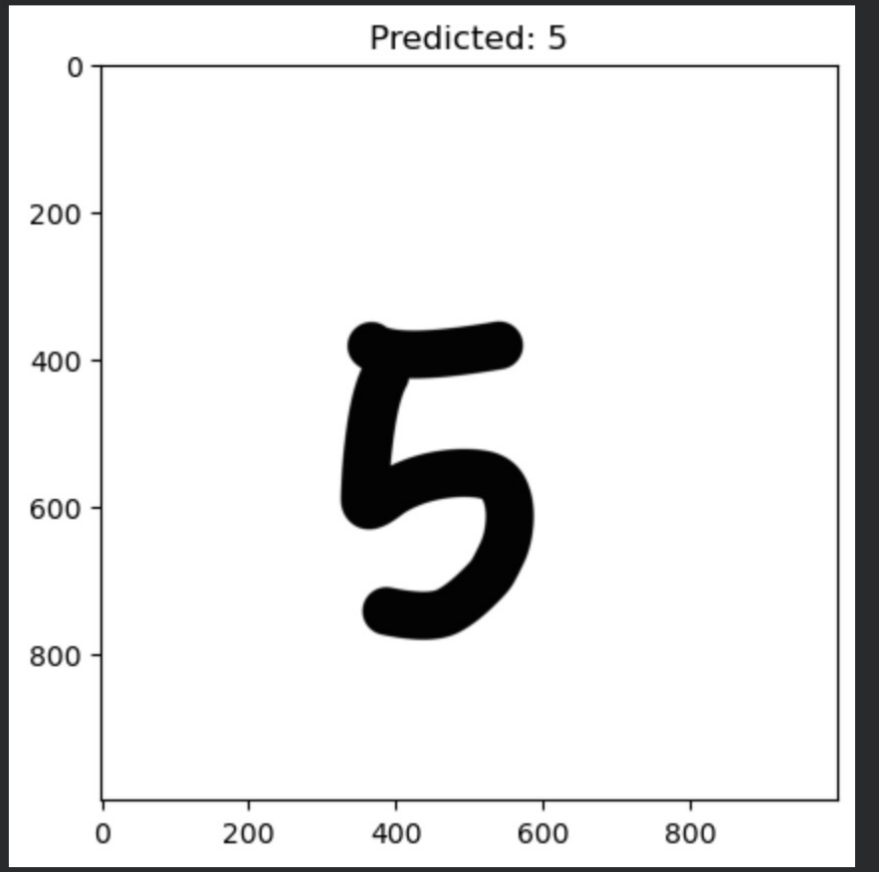


RESULT

```
1/1 [==============================] - 0s 29ms/step
predicted number= 4
```

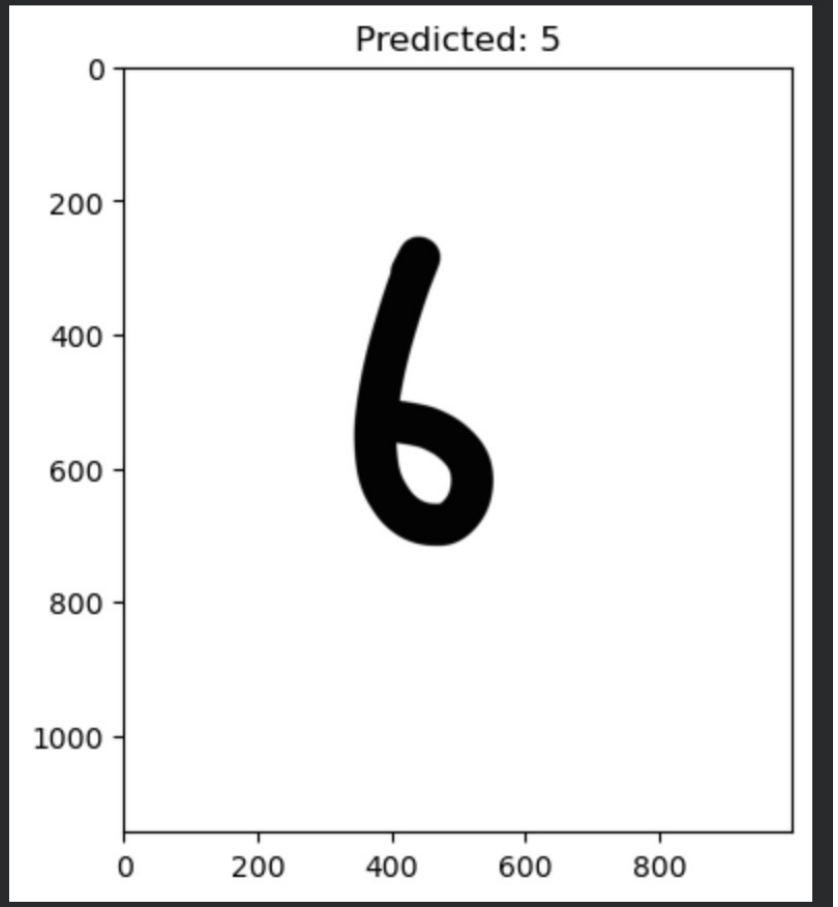



```
1/1 [==============================] - 0s 49ms/step
predicted number= 5
```

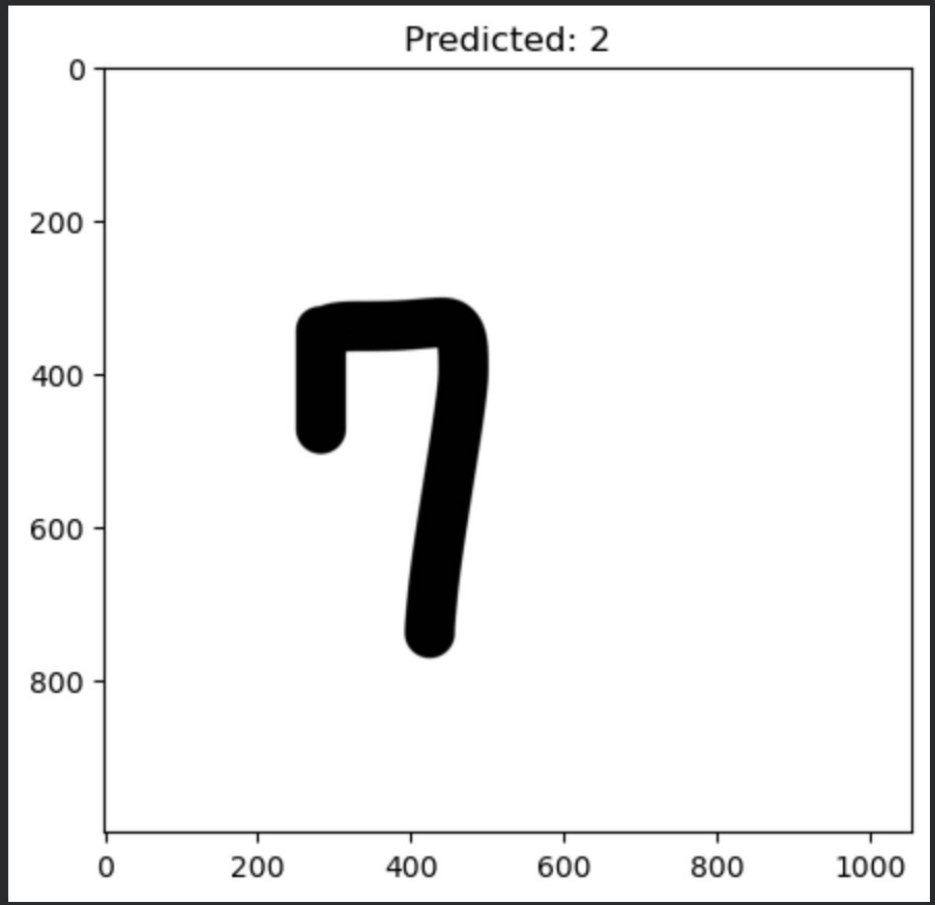


RESULT

```
1/1 [==============================] - 0s 48ms/step
predicted number= 5
```

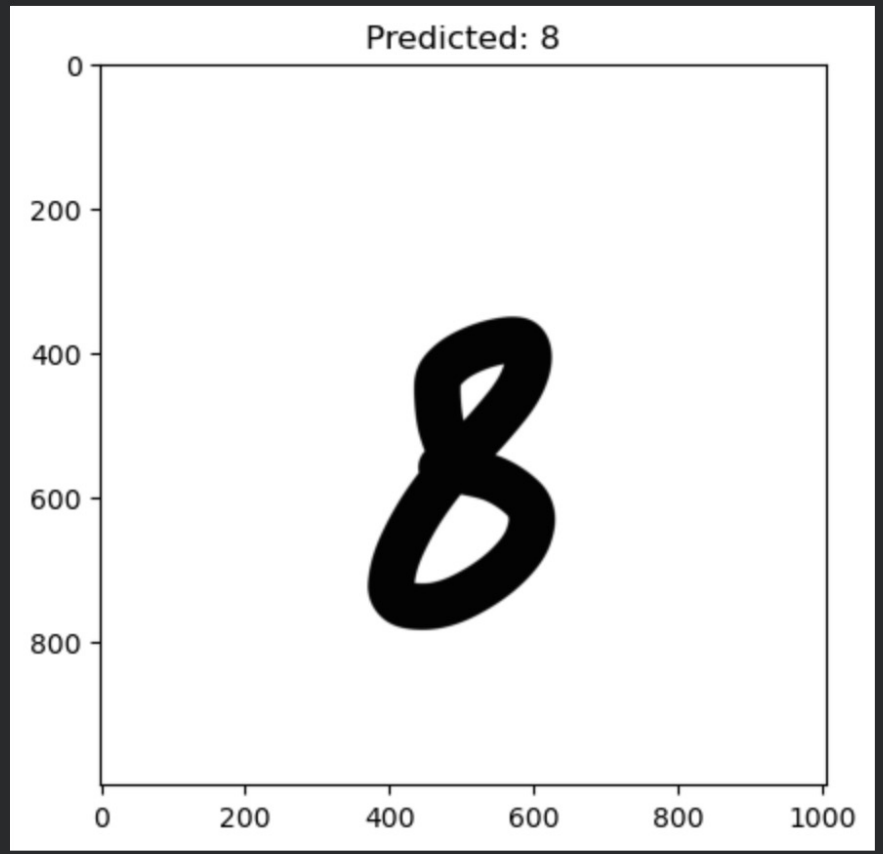



```
1/1 [==============================] - 0s 37ms/step
predicted number= 2
```



RESULT

```
1/1 [==============================] - 0s 40ms/step
predicted number= 8
```




```
1/1 [==============================] - 0s 38ms/step
predicted number= 9
```