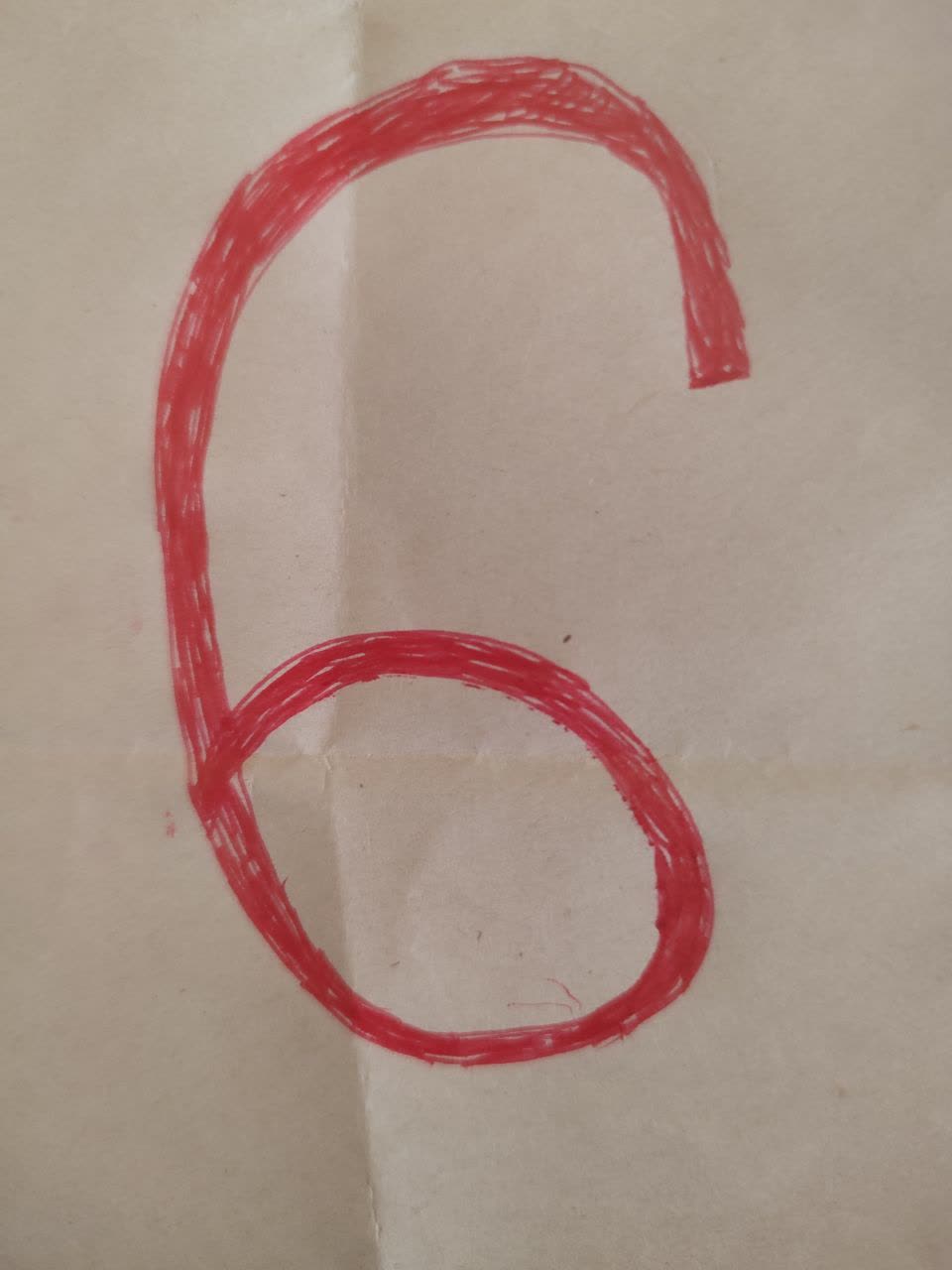
به نام خدا

برای پیش بینی یک عدد دست نویس ابتدا یک عدد روی برگه نوشته و از آن عکس گرفته و در repository در صفحه Github آن را ذخیره می‌کنیم.

عکس عدد ذخیره شده در Github



سپس با استفاده از دیتاست MNIST کدهای دستوری برای پیش بینی این عدد را در کولب اجرا می‌گیریم:

# Import کتابخانه‌های مورد نیاز

import cv2

import numpy as np

import matplotlib.pyplot as plt

from keras.models import load\_model

from keras.datasets import mnist

from keras.utils import to\_categorical

import os

# 1. بارگیری یا آموزش مدل

def load\_or\_train\_model():

    # بررسی وجود فایل مدل

    if os.path.exists('model\_mnist.h5'):

        print("بارگیری مدل از فایل ذخیره شده...")

        model = load\_model('model\_mnist.h5')

    else:

        print("آموزش مدل جدید...")

        # بارگیری داده‌های MNIST

        (X\_train, y\_train), (X\_test, y\_test) = mnist.load\_data()

        # پیش‌پردازش داده‌ها

        X\_train = X\_train.reshape(-1, 28, 28, 1).astype('float32') / 255.0

        X\_test = X\_test.reshape(-1, 28, 28, 1).astype('float32') / 255.0

        y\_train = to\_categorical(y\_train, 10)

        y\_test = to\_categorical(y\_test, 10)

        # ساخت مدل CNN

        from keras.models import Sequential

        from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

        model = Sequential([

            Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

            MaxPooling2D((2, 2)),

            Conv2D(64, (3, 3), activation='relu'),

            MaxPooling2D((2, 2)),

            Flatten(),

            Dense(128, activation='relu'),

            Dropout(0.5),

            Dense(10, activation='softmax')

        ])

        # کامپایل و آموزش مدل

        model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

        model.fit(X\_train, y\_train, epochs=5, batch\_size=64, validation\_data=(X\_test, y\_test))

        # ذخیره مدل

        model.save('model\_mnist.h5')

        print("مدل آموزش داده و ذخیره شد.")

    return model

# 2. تابع پیش‌پردازش تصویر ورودی

def preprocess\_image(image\_path):

    # خواندن تصویر به صورت grayscale

    img = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

    if img is None:

        raise ValueError(f"نمی‌توان تصویر را از مسیر {image\_path} بارگیری کرد")

    # تغییر سایز به 28x28 پیکسل

    img = cv2.resize(img, (28, 28))

    # معکوس کردن رنگ (اگر زمینه تصویر تیره است)

    img = cv2.bitwise\_not(img)

    # نرمال سازی به محدوده [0, 1]

    img = img.astype('float32') / 255.0

    # تغییر شکل به (1, 28, 28, 1)

    img = np.expand\_dims(img, axis=0)  # بعد batch

    img = np.expand\_dims(img, axis=-1)  # بعد channel

    return img

# 3. تابع اصلی برای تشخیص عدد

def recognize\_digit(image\_path):

    # بارگیری یا آموزش مدل

    model = load\_or\_train\_model()

    try:

        # پیش‌پردازش تصویر

        processed\_img = preprocess\_image(image\_path)

        # بررسی شکل تصویر پردازش شده

        print("شکل تصویر پردازش شده:", processed\_img.shape)  # باید (1, 28, 28, 1) باشد

        # نمایش تصویر ورودی

        plt.imshow(processed\_img[0, :, :, 0], cmap='gray')

        plt.title("تصویر ورودی پس از پیش‌پردازش")

        plt.show()

        # پیش‌بینی مدل

        prediction = model.predict(processed\_img)

        predicted\_number = np.argmax(prediction)

        confidence = np.max(prediction) \* 100

        print(f"\nنتیجه تشخیص: عدد {predicted\_number} با اطمینان {confidence:.2f}%")

        return predicted\_number, confidence

    except Exception as e:

        print("خطا در پردازش تصویر:", str(e))

        return None, None

# 4. استفاده از تابع

if \_\_name\_\_ == "\_\_main\_\_":

    # مسیر تصویر مورد نظر را اینجا قرار دهید

    image\_path = "/content/CNN-MNIST/number.jpg"  # یا مسیر کامل مانند '/content/number.jpg'

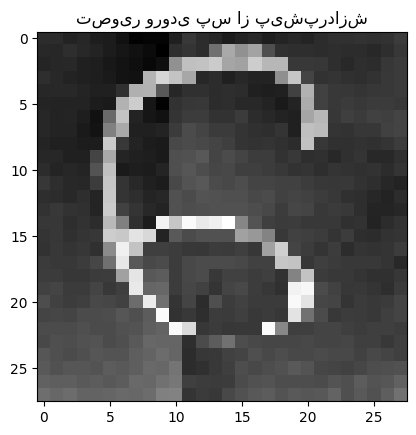
    # تشخیص عدد

    digit, confidence = recognize\_digit(image\_path)

    if digit is not None:

        print(f"\nعدد تشخیص داده شده: {digit} با اطمینان {confidence:.2f}%")

خروجی دستورات بالا :



نتیجه تشخیص: عدد 6 با اطمینان 31.98%