**به نام خدا**

**موضوع****: پیش‌بینی عدد دست‌نویس با استفاده از شبکه‌های عصبی پیچشی (Convolutional Neural Networks)**

کدهای پایتون این پروژه در صفحه gitab به آدرس زیر است:

<https://github.com/Esmaeili403/CNN-MNIST1.git>

برای پیش‌بینی عدد دست‌نویس با استفاده از شبکه‌های عصبی پیچشی مراحل زیر را انجام می‌دهیم:

**فراخوانی کتابخانه های مورد نیاز:**

* **:NumPy** پیش‌پردازش داده‌ها و محاسبات
* **Matplotlib** : نمایش تصاویر و نمودارها
* **TensorFlow/Keras:**
  + بارگذاری داده‌های MNIST
  + ساخت مدل CNN
  + آموزش و ارزیابی مدل
  + ذخیره/بارگذاری مدل

**بارگذاری داده‌ها:**

* از **مجموعه‌داده MNIST** استفاده شده که شامل:
  + **60,000 تصویر آموزشی** و **10,000 تصویر تست**.
  + هر تصویر  **28x28 پیکسل** و **سیاه‌وسفید** (تک‌کاناله) است.
  + اعداد **0 تا 9** 10) کلاس مختلف(.

**پیش‌پردازش داده‌ها:**

* **تغییر شکل داده‌ها :(Reshaping)** تبدیل تصاویر 28 28 x پیکسلی به شکل (28, 28, 1) برای سازگاری با لایه‌های Conv2D
* **نرمال‌سازی(Normalization):** مقادیر پیکسل‌ها از بازه [0, 255] به [0, 1] تبدیل می‌شوند.

**ساخت مدل CNN**

1. **لایه Conv2D اول**:
   * 32 فیلتر 33 x
   * ورودی: تصاویر 28x28 پیکسل (سیاه‌وسفید)
   * فعال‌ساز ReLU :
   * وظیفه: تشخیص لبه‌ها و الگوهای ساده
2. **لایه MaxPooling2D اول**:
   * پنجره 2 x 2
   * وظیفه: کاهش ابعاد و حفظ مهم‌ترین ویژگی‌ها
3. **لایه Conv2D دوم**:
   * 64 فیلتر 33 x
   * فعال‌ساز: ReLU
   * وظیفه: تشخیص الگوهای پیچیده‌تر
4. **لایه MaxPooling2D دوم**:
   * پنجره 2 x 2
   * وظیفه: کاهش بیشتر ابعاد
5. **لایه Flatten** :
   * تبدیل ماتریس 3بعدی به بردار 1بعدی
   * آماده‌سازی برای لایه‌های متراکم
6. **لایه متراکم(Dense) :**
   * 128 نورون
   * فعال‌ساز ReLU :
   * وظیفه: یادگیری ارتباطات غیرخطی
7. **لایه خروجی**:
   * 10 نورون (مطابق ارقام 0-9)
   * فعال‌ساز Softmax :
   * وظیفه: محاسبه احتمال تعلق به هر کلاس

**کامپایل مدل:**

* **بهینه‌ساز** Adam **:(Optimizer)**
* **تابع هزینه**categorical crossentory **:(loss)**
* **متریک**accuracy **:(metric)**

**آموزش مدل:**

* **:epochs=5**مدل 5 بار روی کل داده‌های آموزشی یادگیری می‌کند.
* **:batch\_size=64**در هر مرحله 64 نمونه پردازش می‌شود.
* 20%  **:validation\_split=0.2**داده آموزشی برای اعتبارسنجی جدا می‌شود.

**پیش بینی دو مدل تصادفی**

import numpy as np

import matplotlib.pyplot as plt

import tensorflow as tf

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential

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

# 1. بارگذاری و پیش‌پردازش داده‌ها

(train\_images, train\_labels), (test\_images, test\_labels) = mnist.load\_data()

# نرمال‌سازی و تغییر شکل داده‌ها

train\_images = train\_images.reshape(-1, 28, 28, 1).astype('float32') / 255

test\_images = test\_images.reshape(-1, 28, 28, 1).astype('float32') / 255

# تبدیل برچسب‌ها به فرم one-hot

train\_labels = tf.keras.utils.to\_categorical(train\_labels, 10)

test\_labels = tf.keras.utils.to\_categorical(test\_labels, 10)

# 2. ساخت مدل CNN

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'),

    Dense(10, activation='softmax')

])

model.compile(optimizer='adam',

              loss='categorical\_crossentropy',

              metrics=['accuracy'])

# 3. آموزش مدل

print("در حال آموزش مدل...")

history = model.fit(train\_images, train\_labels,

                    epochs=5,

                    batch\_size=64,

                    validation\_split=0.2)

# 4. ذخیره مدل

model.save('mnist\_cnn\_model.h5')

print("مدل با موفقیت ذخیره شد.")

# 5. پیش‌بینی روی دو نمونه تصادفی

def show\_predictions(num\_samples=2):

    indices = np.random.choice(len(test\_images), num\_samples, replace=False)

    plt.figure(figsize=(10, 4))

    for i, idx in enumerate(indices):

        plt.subplot(1, num\_samples, i+1)

        img = test\_images[idx].reshape(28, 28)

        pred = model.predict(img[np.newaxis, ...])

        true\_label = np.argmax(test\_labels[idx])

        pred\_label = np.argmax(pred)

        plt.imshow(img, cmap='gray')

        plt.title(f'پیش‌بینی: {pred\_label}\nواقعی: {true\_label}')

        plt.axis('off')

    plt.tight\_layout()

    plt.show()

show\_predictions()

**خروجی:**

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz>

**11490434/11490434** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base\_conv.py:107: UserWarning: Do not pass an `input\_shape`/`input\_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.

super().\_\_init\_\_(activity\_regularizer=activity\_regularizer, \*\*kwargs)

در حال آموزش مدل...

Epoch 1/5

**750/750** ━━━━━━━━━━━━━━━━━━━━ **56s** 68ms/step - accuracy: 0.8823 - loss: 0.3949 - val\_accuracy: 0.9775 - val\_loss: 0.0713

Epoch 2/5

**750/750** ━━━━━━━━━━━━━━━━━━━━ **75s** 59ms/step - accuracy: 0.9826 - loss: 0.0571 - val\_accuracy: 0.9846 - val\_loss: 0.0514

Epoch 3/5

**750/750** ━━━━━━━━━━━━━━━━━━━━ **42s** 56ms/step - accuracy: 0.9884 - loss: 0.0373 - val\_accuracy: 0.9871 - val\_loss: 0.0396

Epoch 4/5

**750/750** ━━━━━━━━━━━━━━━━━━━━ **84s** 58ms/step - accuracy: 0.9917 - loss: 0.0255 - val\_accuracy: 0.9873 - val\_loss: 0.0434

Epoch 5/5

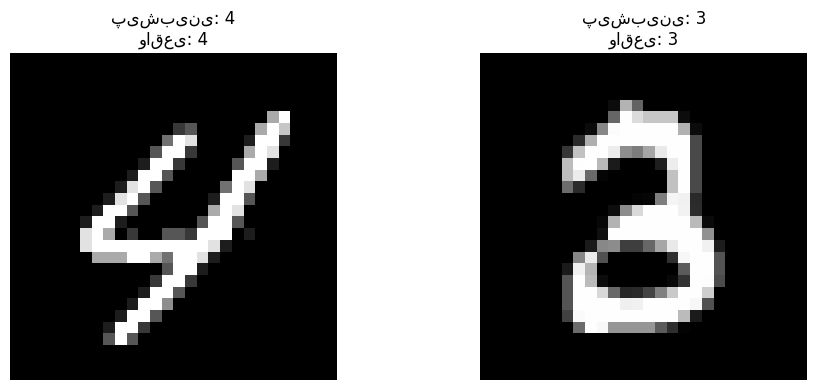
**750/750** ━━━━━━━━━━━━━━━━━━━━ **83s** 59ms/step - accuracy: 0.9942 - loss: 0.0183 - val\_accuracy: 0.9893 - val\_loss: 0.0374

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save\_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my\_model.keras')` or `keras.saving.save\_model(model, 'my\_model.keras')`.

مدل با موفقیت ذخیره شد.

**1/1** ━━━━━━━━━━━━━━━━━━━━ **0s** 192ms/step

**1/1** ━━━━━━━━━━━━━━━━━━━━ **0s** 71ms/step



**رسم ماتریس آشفتگی:**

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential, load\_model

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

from sklearn.metrics import confusion\_matrix

# 1. بارگذاری و پیش‌پردازش داده‌ها

(\_, \_), (test\_images, test\_labels) = mnist.load\_data()

test\_images = test\_images.reshape(-1, 28, 28, 1).astype('float32') / 255

test\_labels\_true = np.argmax(tf.keras.utils.to\_categorical(test\_labels, 10), axis=1)

# 2. بارگذاری مدل (یا آموزش جدید)

try:

    model = load\_model('mnist\_cnn\_model.h5')

    print("مدل از فایل بارگذاری شد.")

except:

    print("مدل یافت نشد! در حال آموزش مدل جدید...")

    (train\_images, train\_labels) = mnist.load\_data()[0]

    train\_images = train\_images.reshape(-1, 28, 28, 1).astype('float32') / 255

    train\_labels = tf.keras.utils.to\_categorical(train\_labels, 10)

    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'),

        Dense(10, activation='softmax')

    ])

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

    model.fit(train\_images, train\_labels, epochs=5, batch\_size=64)

    model.save('mnist\_cnn\_model.h5')

# 3. پیش‌بینی روی داده‌های تست

test\_pred = np.argmax(model.predict(test\_images), axis=1)

# 4. محاسبه ماتریس آشفتگی

cm = confusion\_matrix(test\_labels\_true, test\_pred)

# 5. رسم ماتریس آشفتگی

plt.figure(figsize=(10, 8))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',

            xticklabels=range(10),

            yticklabels=range(10))

plt.xlabel('پیش‌بینی مدل')

plt.ylabel('مقادیر واقعی')

plt.title('ماتریس آشفتگی (MNIST CNN Model)')

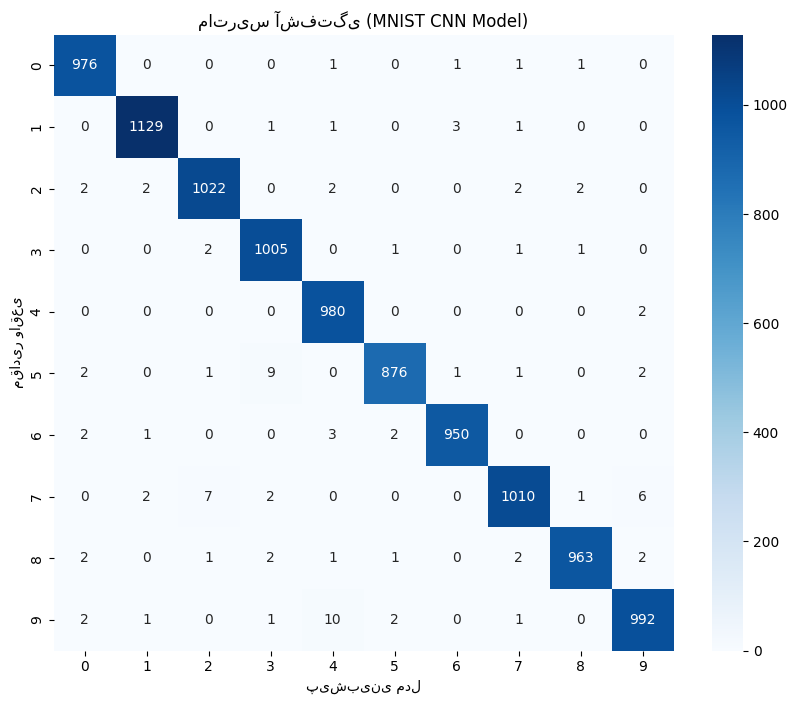
plt.show()

**خروجی:**

WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile\_metrics` will be empty until you train or evaluate the model.

مدل از فایل بارگذاری شد.

**313/313** ━━━━━━━━━━━━━━━━━━━━ **4s** 11ms/step



**تفسیر ماتریس آشفتگی (Confusion Matrix) برای مدل تشخیص اعدادMNIST :**

**1 . ساختار ماتریس:**

* **محور افقی (x)**: پیش‌بینی‌های مدل (۰ تا ۹)
* **محور عمودی (y)**: مقادیر واقعی (۰ تا ۹)
* **هر سلول**: تعداد/درصد نمونه‌هایی که مدل آنها را به یک کلاس خاص نسبت داده است.

**2. مقادیر مطلوب:**

* **مقادیر روی قطر اصلی**
  + نشان‌دهنده پیش‌بینی‌های **صحیح** هستند.
  + مثال: سلول (3,3) تعداد دفعاتی که عدد **۳** به درستی تشخیص داده شده را نشان می‌دهد.

**3 . مقادیر مشکل‌ساز:**

* **مقادیر خارج از قطر اصلی**
  + نشان‌دهنده **خطاهای مدل** هستند.
  + مثال: سلول (5,8) تعداد دفعاتی که عدد **۵** به اشتباه به عنوان **۸** پیش‌بینی شده را نشان می‌دهد.

**4 . نمونه‌های کاربردی از تفسیر:**

* اگر سلول (7,1) مقدار بالایی دارد:
  + مدل در تشخیص تفاوت بین **۷** و **۱** مشکل دارد.
  + احتمالاً به دلیل شباهت ظاهری (مثلاً عدد ۷ با خط مورب کوتاه).
* اگر سطر مربوط به عدد **۹** مقادیر پراکنده‌ای دارد:
  + مدل در تشخیص عدد ۹ ضعیف عمل می‌کند.
  + ممکن است به دلیل تنوع در نحوه نوشتن عدد ۹ باشد.

**پیش بینی یک عدد دستنویس:**

تصویر یک عدد دستنویس در آدرس ریپوی :

<https://github.com/Esmaeili403/CNN-MNIST1.git>

ذخیره شده را روی تصویر پیش پردازش میکنیم:

import cv2

import numpy as np

import matplotlib.pyplot as plt

from tensorflow import keras

from PIL import Image

# 1. دانلود ریپوی عمومی شما

!git clone https://github.com/Esmaeili403/CNN-MNIST1.git

# 2. بارگذاری مدل

try:

    model = keras.models.load\_model('/content/CNN-MNIST1/mnist\_cnn\_model.h5')

    print("✅ مدل با موفقیت بارگذاری شد")

except Exception as e:

    print("❌ خطا در بارگذاری مدل:", e)

    print("\nلیست فایل‌های موجود در ریپو:")

    !ls -la /content/CNN-MNIST1/

# 3. تابع پیش‌پردازش تصویر

def preprocess\_image(image\_path):

    try:

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

        if img is None:

            raise ValueError("تصویر بارگذاری نشد! مسیر را بررسی کنید")

        img = 255 - img  # تبدیل به فرمت MNIST (زمینه سیاه)

        img = cv2.resize(img, (28, 28))

        img = img.astype('float32') / 255

        return img.reshape(1, 28, 28, 1)

    except Exception as e:

        print("❌ خطا در پیش‌پردازش:", e)

        return None

# 4. مسیر تصویر

image\_path = '/content/CNN-MNIST1/5.jpg'

# 5. نمایش و پیش‌بینی

try:

    # نمایش تصویر

    print(f"\n🔍 نمایش تصویر از مسیر: {image\_path}")

    img = Image.open(image\_path)

    plt.imshow(img, cmap='gray')

    plt.axis('off')

    plt.title('تصویر ورودی')

    plt.show()

    # پیش‌بینی

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

    if processed\_img is not None:

        prediction = model.predict(processed\_img)

        predicted\_num = np.argmax(prediction)

        confidence = np.max(prediction) \* 100

        print(f"\n🔢 نتیجه پیش‌بینی:")

        print(f"- عدد تشخیص داده شده: {predicted\_num}")

        print(f"- میزان اطمینان: {confidence:.2f}%")

        # نمایش احتمال‌های تمام کلاس‌ها

        print("\n📊 احتمال‌های تمام اعداد 0-9:")

        for i, prob in enumerate(prediction[0]):

            print(f"عدد {i}: {prob\*100:.2f}%")

except Exception as e:

    print("\n❌ خطا:", e)

    print("\nبرای عیب‌یابی، محتوای ریپو را بررسی کنید:")

    !ls -la /content/CNN-MNIST1/

**خروجی مدل تصویر اصلی:**

Cloning into 'CNN-MNIST1'...

remote: Enumerating objects: 8, done.

remote: Counting objects: 100% (8/8), done.

remote: Compressing objects: 100% (7/7), done.

remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)

Receiving objects: 100% (8/8), 32.82 KiB | 4.69 MiB/s, done.

Resolving deltas: 100% (1/1), done.

❌ خطا در بارگذاری مدل: [Errno 2] Unable to synchronously open file (unable to open file: name = '/content/CNN-MNIST1/mnist\_cnn\_model.h5', errno = 2, error message = 'No such file or directory', flags = 0, o\_flags = 0)

لیست فایل‌های موجود در ریپو:

total 60

drwxr-xr-x 3 root root 4096 Jul 22 12:50 .

drwxr-xr-x 1 root root 4096 Jul 22 12:50 ..

-rw-r--r-- 1 root root 29662 Jul 22 12:50 5.jpg

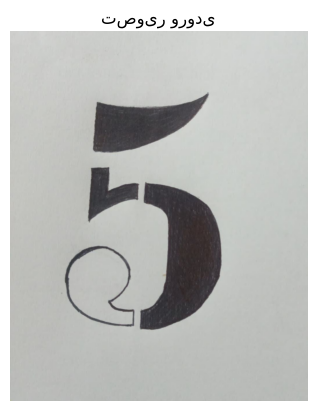
drwxr-xr-x 8 root root 4096 Jul 22 12:50 .git

-rw-r--r-- 1 root root 4688 Jul 22 12:50 .gitignore

-rw-r--r-- 1 root root 1065 Jul 22 12:50 LICENSE

-rw-r--r-- 1 root root 12 Jul 22 12:50 README.md

🔍 نمایش تصویر از مسیر: /content/CNN-MNIST1/5.jpg



**1/1** ━━━━━━━━━━━━━━━━━━━━ **0s** 66ms/step

🔢 نتیجه پیش‌بینی:

- عدد تشخیص داده شده: 5

- میزان اطمینان: 59.54%

📊 احتمال‌های تمام اعداد 0-9:

عدد 0: 0.01%

عدد 1: 0.43%

عدد 2: 0.19%

عدد 3: 35.96%

عدد 4: 0.04%

عدد 5: 59.54%

عدد 6: 0.91%

عدد 7: 1.62%

عدد 8: 0.27%

عدد 9: 1.04%

import cv2

import numpy as np

import matplotlib.pyplot as plt

from tensorflow import keras

# 1. تابع پیش‌پردازش تصویر

def preprocess\_image(image\_path):

    # خواندن تصویر به صورت خاکستری

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

    # معکوس کردن رنگ‌ها (تبدیل به زمینه سیاه)

    img = 255 - img

    # تغییر اندازه به 28x28 پیکسل

    img = cv2.resize(img, (28, 28))

    # نرمال‌سازی

    img = img.astype('float32') / 255

    # تغییر شکل برای ورودی مدل

    return img.reshape(1, 28, 28, 1)

# 2. مسیر تصویر شما - تغییر دهید

image\_path = '/content/CNN-MNIST1/5.jpg'  # مسیر تصویر خود را وارد کنید

# 3. پیش‌پردازش تصویر

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

# 4. نمایش تصویر پردازش شده

print("تصویر پس از پردازش (آماده برای مدل):")

plt.imshow(processed\_img.reshape(28, 28), cmap='gray')

plt.axis('off')

plt.show()

# 5. پیش‌بینی با مدل

prediction = model.predict(processed\_img)

predicted\_number = np.argmax(prediction)

confidence = np.max(prediction) \* 100

# 6. نمایش نتیجه

print(f"\nمدل پیش‌بینی می‌کند این عدد {predicted\_number} است با اطمینان {confidence:.2f}%")

# 7. نمایش احتمال‌های تمام کلاس‌ها

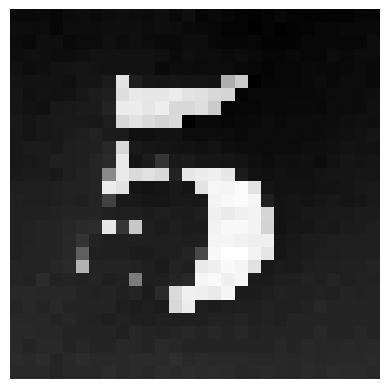
print("\nمیزان اطمینان برای تمام اعداد 0-9:")

for i, prob in enumerate(prediction[0]):

    print(f"عدد {i}: {prob\*100:.2f}%")

**خروجی:**

تصویر پس از پردازش (آماده برای مدل):

****

**1/1** ━━━━━━━━━━━━━━━━━━━━ **0s** 56ms/step

مدل پیش‌بینی می‌کند این عدد 5 است با اطمینان 59.54%

میزان اطمینان برای تمام اعداد 0-9:

عدد 0: 0.01%

عدد 1: 0.43%

عدد 2: 0.19%

عدد 3: 35.96%

عدد 4: 0.04%

عدد 5: 59.54%

عدد 6: 0.91%

عدد 7: 1.62%

عدد 8: 0.27%

عدد 9: 1.04%

**تفسیر نتایج پیش‌بینی مدل:**

**.1 پیش‌بینی اصلی:**

* **عدد تشخیص داده شده** : ۵
* **میزان اطمینان** : % 59.54
  + این نشان می‌دهد مدل با **اعتماد متوسطی** عدد ۵ را پیش‌بینی کرده است (حداقل ۸۰% معمولاً اطمینان بالایی محسوب می‌شود).

**2. توزیع احتمالات:**

* **رقم دوم محتمل**: عدد ۳ با 35.96%
  + این نشان می‌دهد تصویر ورودی ممکن است ویژگی‌های مشترکی با عدد ۳ داشته باشد (مثلاً شباهت در انحناها).
* **سایر ارقام**: احتمال ناچیز (محدوده ۱%) برای سایر اعداد.