b{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "markdown",

"source": [

"Import the necessary packages"

],

"metadata": {

"id": "--cnmsWjvNSv"

}

},

{

"cell\_type": "code",

"source": [

"import numpy as np \n",

"import pandas as pd\n",

"import matplotlib.pyplot as plt\n",

"from keras.utils import np\_utils\n",

"from tensorflow.keras.datasets import mnist\n",

"from tensorflow.keras.models import Sequential\n",

"from tensorflow.keras.layers import Conv2D, Dense, Flatten\n",

"from tensorflow.keras.optimizers import Adam\n",

"from tensorflow.keras.models import load\_model\n",

"from PIL import Image, ImageOps"

],

"metadata": {

"id": "jx\_f8BBbvQbm"

},

"execution\_count": 1,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"Load data"

],

"metadata": {

"id": "HJsIIQYUvsxW"

}

},

{

"cell\_type": "code",

"source": [

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

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "RnvdpFZjvvV1",

"outputId": "22c49394-8aae-4a7d-9842-5b4499980392"

},

"execution\_count": 2,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz\n",

"11490434/11490434 [==============================] - 0s 0us/step\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"\n",

"Data Analysis"

],

"metadata": {

"id": "I8o2Doc1wA8P"

}

},

{

"cell\_type": "code",

"source": [

"print(X\_train.shape)\n",

"print(X\_test.shape)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "xVD8hWDxwGG0",

"outputId": "c94570b5-a9af-4182-a042-e3130ab18f14"

},

"execution\_count": 3,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"(60000, 28, 28)\n",

"(10000, 28, 28)\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"X\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "ANE6GfyCwWRM",

"outputId": "150fd290-64ef-4b9f-faed-060527e3801a"

},

"execution\_count": 4,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3,\n",

" 18, 18, 18, 126, 136, 175, 26, 166, 255, 247, 127, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 30, 36, 94, 154, 170,\n",

" 253, 253, 253, 253, 253, 225, 172, 253, 242, 195, 64, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 49, 238, 253, 253, 253, 253,\n",

" 253, 253, 253, 253, 251, 93, 82, 82, 56, 39, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 18, 219, 253, 253, 253, 253,\n",

" 253, 198, 182, 247, 241, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 80, 156, 107, 253, 253,\n",

" 205, 11, 0, 43, 154, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1, 154, 253,\n",

" 90, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 139, 253,\n",

" 190, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 190,\n",

" 253, 70, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 35,\n",

" 241, 225, 160, 108, 1, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 81, 240, 253, 253, 119, 25, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 45, 186, 253, 253, 150, 27, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 16, 93, 252, 253, 187, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 249, 253, 249, 64, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 46, 130, 183, 253, 253, 207, 2, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 39,\n",

" 148, 229, 253, 253, 253, 250, 182, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 114, 221,\n",

" 253, 253, 253, 253, 201, 78, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 23, 66, 213, 253, 253,\n",

" 253, 253, 198, 81, 2, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 18, 171, 219, 253, 253, 253, 253,\n",

" 195, 80, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 55, 172, 226, 253, 253, 253, 253, 244, 133,\n",

" 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 136, 253, 253, 253, 212, 135, 132, 16, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0]], dtype=uint8)"

]

},

"metadata": {},

"execution\_count": 4

}

]

},

{

"cell\_type": "code",

"source": [

"y\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "-kBxvEB9wrvd",

"outputId": "c0726051-2c34-4e3c-c09e-b269b5a4bb01"

},

"execution\_count": 5,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"5"

]

},

"metadata": {},

"execution\_count": 5

}

]

},

{

"cell\_type": "code",

"source": [

"plt.imshow(X\_train[0])"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 282

},

"id": "wKrhHrkjw4UQ",

"outputId": "2b2d51fb-784b-4229-ad53-0a75924a275b"

},

"execution\_count": 6,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<matplotlib.image.AxesImage at 0x7f44967b3110>"

]

},

"metadata": {},

"execution\_count": 6

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

},

{

"cell\_type": "markdown",

"source": [

"Data Pre-Processing"

],

"metadata": {

"id": "ZxQukQi4xBE\_"

}

},

{

"cell\_type": "code",

"source": [

"X\_train = X\_train.reshape(60000, 28, 28, 1).astype('float32')\n",

"X\_test = X\_test.reshape(10000, 28, 28, 1).astype('float32')"

],

"metadata": {

"id": "xWUYBhmExDME"

},

"execution\_count": 7,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"number\_of\_classes = 10\n",

"Y\_train = np\_utils.to\_categorical(y\_train, number\_of\_classes)\n",

"Y\_test = np\_utils.to\_categorical(y\_test, number\_of\_classes)\n"

],

"metadata": {

"id": "sm69QfsdxOpE"

},

"execution\_count": 8,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"Y\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "YqYuXq8NxeCW",

"outputId": "114ee57a-94ce-40ee-fe75-ae9adeec2ec6"

},

"execution\_count": 9,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)"

]

},

"metadata": {},

"execution\_count": 9

}

]

},

{

"cell\_type": "markdown",

"source": [

"Create model"

],

"metadata": {

"id": "HIQlrWZtxkHx"

}

},

{

"cell\_type": "code",

"source": [

"model = Sequential()\n",

"model.add(Conv2D(64, (3, 3), input\_shape=(28, 28, 1), activation=\"relu\"))\n",

"model.add(Conv2D(32, (3, 3), activation=\"relu\"))\n",

"model.add(Flatten())\n",

"model.add(Dense(number\_of\_classes, activation=\"softmax\"))"

],

"metadata": {

"id": "GY8TgJa6xocT"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"\n",

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

],

"metadata": {

"id": "yilp2D3wyEZH"

},

"execution\_count": 16,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"Train the model"

],

"metadata": {

"id": "qc0jhTyTymEx"

}

},

{

"cell\_type": "code",

"source": [

"\n",

"model.fit(X\_train, Y\_train, batch\_size=32, epochs=5, validation\_data=(X\_test,Y\_test))"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "\_-PumDijynre",

"outputId": "a7ac6d4a-445d-4431-c6f8-1b02f8e522f6"

},

"execution\_count": 14,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Epoch 1/5\n",

"1875/1875 [==============================] - 198s 105ms/step - loss: 0.0222 - accuracy: 0.9930 - val\_loss: 0.1091 - val\_accuracy: 0.9782\n",

"Epoch 2/5\n",

"1875/1875 [==============================] - 186s 99ms/step - loss: 0.0216 - accuracy: 0.9935 - val\_loss: 0.1289 - val\_accuracy: 0.9760\n",

"Epoch 3/5\n",

"1875/1875 [==============================] - 189s 101ms/step - loss: 0.0173 - accuracy: 0.9949 - val\_loss: 0.1459 - val\_accuracy: 0.9763\n",

"Epoch 4/5\n",

"1875/1875 [==============================] - 187s 100ms/step - loss: 0.0162 - accuracy: 0.9952 - val\_loss: 0.1669 - val\_accuracy: 0.9775\n",

"Epoch 5/5\n",

"1875/1875 [==============================] - 186s 99ms/step - loss: 0.0181 - accuracy: 0.9949 - val\_loss: 0.1807 - val\_accuracy: 0.9758\n"

]

},

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<keras.callbacks.History at 0x7f4496723910>"

]

},

"metadata": {},

"execution\_count": 14

}

]

},

{

"cell\_type": "markdown",

"source": [

"Test the model"

],

"metadata": {

"id": "fu-aaoFzy6Pe"

}

},

{

"cell\_type": "code",

"source": [

"metrics = model.evaluate(X\_test, Y\_test, verbose=0)\n",

"print(\"Metrics (Test Loss & Test Accuracy): \")\n",

"print(metrics)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "FprPF7Xb1H1X",

"outputId": "a39bfa12-ba4e-4636-c039-59f9e9bff427"

},

"execution\_count": 18,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Metrics (Test Loss & Test Accuracy): \n",

"[24.948211669921875, 0.056699998676776886]\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"prediction = model.predict(X\_test[:4])\n",

"print(prediction)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "DHxRKlrp1MDZ",

"outputId": "adec4d85-e849-4a0c-9701-b1e03a93cdd8"

},

"execution\_count": 19,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"1/1 [==============================] - 0s 99ms/step\n",

"[[3.3031249e-01 5.2466442e-07 6.6763020e-01 7.4586644e-18 1.3067258e-03\n",

" 4.8238340e-05 2.6417222e-06 7.9962525e-08 1.8216917e-05 6.8085088e-04]\n",

" [4.1300183e-10 4.9158923e-12 8.4940614e-12 6.2312724e-25 9.9999964e-01\n",

" 8.6904546e-19 4.0910317e-07 4.1847433e-12 2.4420531e-26 1.5513685e-20]\n",

" [9.9933320e-01 2.5993482e-17 9.7027716e-12 1.9040267e-15 3.5872455e-07\n",

" 2.5597208e-11 4.5366080e-10 6.6644413e-04 3.2339611e-09 2.9506362e-14]\n",

" [1.5448860e-10 8.1449330e-01 1.8550673e-01 4.3593159e-25 2.7277794e-12\n",

" 8.2109436e-16 1.3275955e-10 3.9654284e-09 2.6381562e-24 7.2211286e-17]]\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"Save the model"

],

"metadata": {

"id": "DmnKyJHT0JI3"

}

},

{

"cell\_type": "code",

"source": [

"model.save(\"model.h5\")"

],

"metadata": {

"id": "PlzXkl5Y0O77"

},

"execution\_count": 21,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"Test the saved model"

],

"metadata": {

"id": "pfLJeyMY0dHL"

}

},

{

"cell\_type": "code",

"source": [

"model=load\_model(\"model.h5\")"

],

"metadata": {

"id": "VrKNrZHY0fAc"

},

"execution\_count": 22,

"outputs": []

}

]

}