**Lab Exercise - Deploy a TensorFlow/Keras CNN in Docker**

**Objective**

Train and save a simple CNN model using Keras, build a FastAPI app to serve predictions, and run it inside a Docker container.

**Project Structure**

cnn-tf-docker/

├── app/

│ ├── main.py

│ ├── cnn\_model.h5

├── train\_cnn.py

├── requirements.txt

├── Dockerfile

**Step 1: Train a Simple CNN Model (train\_cnn.py)**

# train\_cnn.py

import tensorflow as tf

from tensorflow.keras.models import Sequential

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

from tensorflow.keras.datasets import mnist

from tensorflow.keras.utils import to\_categorical

(x\_train, y\_train), \_ = mnist.load\_data()

x\_train = x\_train.reshape(-1, 28, 28, 1).astype("float32") / 255

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

model = Sequential([

Conv2D(32, kernel\_size=(3, 3), activation="relu", input\_shape=(28, 28, 1)),

MaxPooling2D(pool\_size=(2, 2)),

Flatten(),

Dense(128, activation="relu"),

Dense(10, activation="softmax")

])

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

model.fit(x\_train, y\_train, epochs=1, batch\_size=128, verbose=1)

model.save("app/cnn\_model.h5")

print("Model saved to app/cnn\_model.h5")

Run this script to save the model:

python train\_cnn.py

**Step 2: Create FastAPI App (app/main.py)**

from fastapi import FastAPI

from pydantic import BaseModel

import tensorflow as tf

from tensorflow.keras.models import load\_model

import numpy as np

from PIL import Image

import base64

import io

app = FastAPI()

model = load\_model("app/cnn\_model.h5")

class ImageInput(BaseModel):

image\_base64: str

@app.post("/predict/")

def predict(data: ImageInput):

image = Image.open(io.BytesIO(base64.b64decode(data.image\_base64))).convert("L")

image = image.resize((28, 28))

image = np.array(image).reshape(1, 28, 28, 1).astype("float32") / 255

prediction = model.predict(image)

pred\_class = int(np.argmax(prediction))

return {"prediction": pred\_class}

**Step 3: requirements.txt**

fastapi

uvicorn

tensorflow

pillow

**Step 4: Dockerfile**

FROM python:3.10-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY app/ ./app

EXPOSE 8000

CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port", "8000"]

**Step 5: Build the Docker Image**

docker build -t tf-cnn-api .

**Step 6: Run the Docker Container**

docker run -p 8000:8000 tf-cnn-api

**Step 7: Test the API (Python client)**

import requests

import base64

with open("digit.png", "rb") as f:

img\_base64 = base64.b64encode(f.read()).decode("utf-8")

response = requests.post("http://localhost:8000/predict/", json={"image\_base64": img\_base64})

print(response.json())

Make sure digit.png is a 28x28 grayscale image or resized to that format.

**Expected Output**

{"prediction": 7}