Exp 10

Design and implement a CNN model (with 4+ layers of convolutions) to classify multi category image datasets. Use the MNIST, Fashion MNIST, CIFAR-10 datasets. Set the No. of Epoch as 5 and 10 . Make the necessary changes whenever required. Record the accuracy corresponding to the number of epochs. Record the time required to run the program, using CPU as well as using GPU in Colab.

import time

import tensorflow as tf

from tensorflow.keras import layers, models

from tensorflow.keras.datasets import mnist, fashion\_mnist, cifar10

from tensorflow.keras.utils import to\_categorical

# Detect device

device\_name = tf.test.gpu\_device\_name()

if device\_name:

    print(f" GPU is available: {device\_name}")

else:

    print(" GPU not found. Using CPU...")

# Smaller CNN (4 conv layers)

def build\_small\_cnn\_model(input\_shape, num\_classes):

    model = models.Sequential([

        layers.Conv2D(16, (3, 3), activation='relu', padding='same', input\_shape=input\_shape),

        layers.Conv2D(16, (3, 3), activation='relu', padding='same'),

        layers.MaxPooling2D((2, 2)),

        layers.Conv2D(32, (3, 3), activation='relu', padding='same'),

        layers.Conv2D(32, (3, 3), activation='relu', padding='same'),

        layers.MaxPooling2D((2, 2)),

        layers.Flatten(),

        layers.Dense(64, activation='relu'),

        layers.Dense(num\_classes, activation='softmax')

    ])

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

    return model

# Data loader and preprocessor

def prepare\_dataset(name):

    if name == "mnist":

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

        x\_train = x\_train.reshape(-1, 28, 28, 1)

        x\_test = x\_test.reshape(-1, 28, 28, 1)

    elif name == "fashion\_mnist":

        (x\_train, y\_train), (x\_test, y\_test) = fashion\_mnist.load\_data()

        x\_train = x\_train.reshape(-1, 28, 28, 1)

        x\_test = x\_test.reshape(-1, 28, 28, 1)

    elif name == "cifar10":

        (x\_train, y\_train), (x\_test, y\_test) = cifar10.load\_data()

    else:

        raise ValueError("Unknown dataset")

    x\_train = x\_train.astype('float32') / 255.0

    x\_test = x\_test.astype('float32') / 255.0

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

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

    # Reduce data for faster CPU training

    x\_train = x\_train[:10000]

    y\_train = y\_train[:10000]

    return x\_train, y\_train, x\_test, y\_test, x\_train.shape[1:]

# Training and evaluation

def train\_model(dataset\_name, epochs\_list=[5, 10]):

    print(f"\n Dataset: {dataset\_name}")

    x\_train, y\_train, x\_test, y\_test, input\_shape = prepare\_dataset(dataset\_name)

    results = []

    for epochs in epochs\_list:

        model = build\_small\_cnn\_model(input\_shape, 10)

        start = time.time()

        history = model.fit(x\_train, y\_train, epochs=epochs, batch\_size=64,

                            validation\_split=0.1, verbose=0)

        end = time.time()

        test\_loss, test\_acc = model.evaluate(x\_test, y\_test, verbose=0)

        duration = end - start

        print(f" Epochs: {epochs} | Accuracy: {test\_acc:.4f} | Time: {duration:.2f}s")

        results.append((epochs, test\_acc, duration))

    return results

# Run for each dataset

all\_results = {}

datasets = ["mnist", "fashion\_mnist"]  # You can add "cifar10" if running on GPU

for name in datasets:

    all\_results[name] = train\_model(name)

# Print final summary

print("\n Summary (CPU Optimized):")

print("Dataset\t\tEpochs\tAccuracy\tTime (s)")

for dataset, records in all\_results.items():

    for epochs, acc, dur in records:

        print(f"{dataset}\t{epochs}\t{acc:.4f}\t\t{dur:.2f}")

CPU

GPU not found. Using CPU...

Dataset: mnist

/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)

Epochs: 5 | Accuracy: 0.9766 | Time: 49.26s

/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)

Epochs: 10 | Accuracy: 0.9801 | Time: 92.74s

Dataset: fashion\_mnist

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz>

**29515/29515** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz>

**26421880/26421880** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz>

**5148/5148** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz>

**4422102/4422102** ━━━━━━━━━━━━━━━━━━━━ **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)

🧪 Epochs: 5 | Accuracy: 0.8619 | Time: 44.93s

/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)

Epochs: 10 | Accuracy: 0.8785 | Time: 96.54s

Summary (CPU Optimized):

Dataset Epochs Accuracy Time (s)

mnist 5 0.9766 49.26

mnist 10 0.9801 92.74

fashion\_mnist 5 0.8619 44.93

fashion\_mnist 10 0.8785 96.54

GPU

GPU is available: /device:GPU:0

Dataset: mnist

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

**11490434/11490434** ━━━━━━━━━━━━━━━━━━━━ **2s** 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)

🧪 Epochs: 5 | Accuracy: 0.9791 | Time: 11.39s

/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)

Epochs: 10 | Accuracy: 0.9800 | Time: 11.41s

Dataset: fashion\_mnist

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz>

**29515/29515** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz>

**26421880/26421880** ━━━━━━━━━━━━━━━━━━━━ **2s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz>

**5148/5148** ━━━━━━━━━━━━━━━━━━━━ **0s** 0us/step

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz>

**4422102/4422102** ━━━━━━━━━━━━━━━━━━━━ **1s** 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)

Epochs: 5 | Accuracy: 0.8694 | Time: 9.12s

/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)

Epochs: 10 | Accuracy: 0.8726 | Time: 13.71s

Dataset: cifar10

Downloading data from <https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz>

**170498071/170498071** ━━━━━━━━━━━━━━━━━━━━ **13s** 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)

Epochs: 5 | Accuracy: 0.5161 | Time: 15.84s

/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)

Epochs: 10 | Accuracy: 0.5768 | Time: 11.36s

Summary (CPU Optimized):

Dataset Epochs Accuracy Time (s)

mnist 5 0.9791 11.39

mnist 10 0.9800 11.41

fashion\_mnist 5 0.8694 9.12

fashion\_mnist 10 0.8726 13.71

cifar10 5 0.5161 15.84

cifar10 10 0.5768 11.36