**Design and implement a CNN model (with 2+ layers of convolutions) to classify multi category image datasets. Use the concept of padding and Batch Normalization while designing the CNN model. Record the accuracy corresponding to the number of epochs. Use the Fashion MNIST/MNIST/CIFAR10 datasets.**

# Import necessary libraries

import tensorflow as tf

import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras import layers, models

from tensorflow.keras.datasets import fashion\_mnist # Use MNIST or CIFAR-10 as needed

from tensorflow.keras.utils import to\_categorical

# Load Fashion MNIST dataset (Use mnist/cifar10 as needed)

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

# Reshape dataset for CNN input (28x28 grayscale images with 1 channel)

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

# Convert labels to one-hot encoding

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

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

# Build CNN model with padding and batch normalization

def build\_cnn\_model():

model = models.Sequential([

layers.Conv2D(32, (3,3), activation='relu', padding='same', input\_shape=(28,28,1)),

layers.BatchNormalization(),

layers.MaxPooling2D((2,2)),

layers.Conv2D(64, (3,3), activation='relu', padding='same'),

layers.BatchNormalization(),

layers.MaxPooling2D((2,2)),

layers.Conv2D(128, (3,3), activation='relu', padding='same'),

layers.BatchNormalization(),

layers.MaxPooling2D((2,2)),

layers.Flatten(),

layers.Dense(128, activation='relu'),

layers.BatchNormalization(),

layers.Dropout(0.5), # Dropout to reduce overfitting

layers.Dense(10, activation='softmax') # Output layer for 10 categories

])

# Compile model

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

return model

# Create CNN model

cnn\_model = build\_cnn\_model()

# Train CNN model and record accuracy over epochs

history = cnn\_model.fit(X\_train, y\_train, epochs=20, batch\_size=64, validation\_data=(X\_test, y\_test), verbose=1)

# Evaluate model performance

test\_loss, test\_acc = cnn\_model.evaluate(X\_test, y\_test, verbose=0)

print(f"Test Accuracy: {test\_acc:.4f}")

# Plot Accuracy vs. Epochs

plt.figure(figsize=(10,5))

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.title('CNN Model: Accuracy over Epochs')

plt.legend()

plt.show()