DL 8 MNIST FASHION

CODE:

```
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow import keras
import numpy as np
import pandas as pd

(x_train, y_train), (x_test,y_test) = keras.datasets.fashion_mnist.load_data()
plt.imshow(x_train[1])
plt.imshow(x_train[2])
plt.imshow(x_train[0])
x_train=x_train.astype('float32')/255.0
x_test=x_test.astype('float32')/255.0

x_train=x_train.reshape(60000,28,28,1)
x_test=x_test.reshape(10000,28,28,1)

x_train.shape
x_test.shape

y_train.shape
y_test.shape


model= keras.Sequential([
   keras.layers.Conv2D(32,(3,3), activation='relu',input_shape=(28,28,1)),
   keras.layers.MaxPooling2D((2,2)),
   keras.layers.Dropout(0.25),
   keras.layers.Conv2D(64, (3,3), activation='relu'),
   keras.layers.MaxPooling2D((2,2)),
   keras.layers.Dropout(0.25),
   keras.layers.Conv2D(128, (3,3), activation='relu'),
   keras.layers.Flatten(),
   keras.layers.Dense(128, activation='relu'),
   keras.layers.Dropout(0.25),
   keras.layers.Dense(10,activation='softmax')
])

model.summary()
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
history=model.fit(x_train,y_train,epochs=10,validation_data=(x_test,y_test))

test_loss,test_acc=model.evaluate(x_test,y_test)
print('Test accuracy:',test_acc)
```

EXPLANATION

Certainly! Let's go through the code step by step:

The code provided demonstrates the training of a convolutional neural network (CNN) model for fashion item classification using the Fashion MNIST dataset. Here's an explanation of the code:

1. Import libraries: The necessary libraries are imported, including TensorFlow, Keras, NumPy, Pandas, and matplotlib.

2. Load the Fashion MNIST dataset: The Fashion MNIST dataset is loaded using `keras.datasets.fashion_mnist.load_data()`. It provides 60,000 training images and 10,000 testing images of fashion items. The data is divided into features (images) and labels (class/category).

3. Visualize images: Three images from the training set (`x_train[1]`, `x_train[2]`, `x_train[0]`) are displayed using `plt.imshow()`.

4. Preprocess the data: The pixel values of the images are normalized by dividing them by 255 to scale them between 0 and 1. This is done for both the training and testing data.

5. Reshape the data: The shape of the images is reshaped to include a single channel. The original shape of (60000, 28, 28) becomes (60000, 28, 28, 1) for the training data, and the original shape of (10000, 28, 28) becomes (10000, 28, 28, 1) for the testing data.

6. Define the model architecture: A sequential model is created using `keras.Sequential()`. It consists of multiple layers, including convolutional layers (`Conv2D`), max pooling layers (`MaxPooling2D`), dropout layers (`Dropout`), and dense layers (`Dense`). The last layer has 10 units with softmax activation for multi-class classification.

7. Compile the model: The model is compiled with the Adam optimizer, sparse categorical cross-entropy loss (as the labels are integers), and accuracy metric.

8. Train the model: The model is trained on the training data (`x_train` and `y_train`) using `fit()`. The training is performed for 10 epochs, and the validation data (`x_test` and `y_test`) is used for monitoring the performance during training.

9. Evaluate the model: The model is evaluated on the testing data using `evaluate()`, and the test loss and accuracy are stored in `test_loss` and `test_acc`, respectively.

10. Print the test accuracy: The test accuracy of the trained model is printed.

In summary, this code trains a CNN model to classify fashion items using the Fashion MNIST dataset. The model is trained, evaluated, and its test accuracy is reported.

ORAL QUESTION

1. What is classification - Classification is a type of supervised learning in machine learning that involves categorizing data into predefined classes or categories based on a set of features or characteristics. It is used to predict the class of new, unseen data based on the patterns learned from the labeled training data.
2. CNN - A CNN is a type of deep neural network that is commonly used for image classification tasks.
3. Dataset ??? - The MNIST Fashion dataset is a collection of 70,000 grayscale images of 28x28 pixels, representing 10 different categories of clothing and accessories. The categories include T-shirts/tops, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and ankle boots. The dataset is often used as a benchmark for testing image classification algorithms, and it is considered a more challenging version of the original MNIST dataset which contains handwritten digits. The MNIST Fashion dataset was released by Zalando Research in 2017 and has since become a popular dataset in the machine learning community. he MNIST Fashion dataset is a collection of 70,000 grayscale images of 28x28 pixels each. These images represent 10 different categories of clothing and accessories, with each category containing 7,000 images. The categories are as follows:
   T-shirt/tops
   Trousers
   Pullovers
   Dresses
   Coats
   Sandals
   Shirts
   Sneakers
   Bags
   Ankle boots
   The images were obtained from Zalando's online store and are preprocessed to be normalized and centered. The training set contains 60,000 images, while the test set contains 10,000 images. The goal of the dataset is to accurately classify the images into their respective categories.

4. Deep neural networks using CNNs work on classification tasks by learning to automatically extract features from input images and using those features to make predictions.
   Here's how it works:
   a) Input layer: The input layer of the network takes in the image data as input.
   b) Convolutional layers: The convolutional layers apply filters to the input images to extract relevant features. Each filter produces a feature map that highlights areas of the image that match the filter.
   c) Activation functions: An activation function is applied to the output of each convolutional layer to introduce non-linearity into the network. Pooling layers: The pooling layers downsample the feature maps to reduce the spatial dimensions of the data.
   d) Dropout layer: Dropout is used to prevent overfitting by randomly dropping out a percentage of the neurons in the network during training.
   e) Fully connected layers: The fully connected layers take the flattened output from the last pooling layer and perform a classification task by outputting a probability distribution over the possible classes.