FashionMNIST Documentation

This code implements a simple convolutional neural network using numpy and scipy libraries. It defines several classes for different types of layers such as fully connected layer, convolutional layer, and activation layer. The network is trained using mean squared error loss and backpropagation algorithm.

The code starts by importing the necessary libraries such as numpy, scipy, and keras. It then defines several classes for different layers such as FullyConnected, Convolutional, Activation, Tanh, Sigmoid, and Reshape.

The FullyConnected class defines a fully connected layer, which takes the input from the previous layer, multiplies it with randomly initialized weights and adds a bias term to produce the output. The forward and backward methods of this class implement the forward and backward pass, respectively.

The Convolutional class defines a convolutional layer, which takes an image as input and produces a feature map by applying several filters (kernels) to the image. The forward and backward methods of this class implement the forward and backward pass, respectively.

The Activation class defines an activation layer, which applies an activation function to the output of the previous layer. The Tanh and Sigmoid classes inherit from the Activation class and implement the tanh and sigmoid activation functions, respectively.

The Reshape class is used to reshape the output of the convolutional layer to be compatible with the fully connected layer.

The predict and train functions are used to predict the labels and train the network, respectively.

The code then loads the fashion_mnist dataset using keras and preprocesses it by reshaping the data, normalizing it, and categorizing the labels. It then creates a network using several layers and trains it on the preprocessed data using the mean squared error loss and backpropagation algorithm.

Finally, the code predicts the labels of the test data using the trained network and calculates the accuracy.