Programming Assignment 2: Classifying clothes

You might have noticed Dr. B is wearing T-shirts for the last couple of weeks; luckily not the same T-shirt! As the season is kind-of chilling out over time, you may notice a very little change in his fashion sense (or not). He is becoming lethargic to picking out a trendy cloth (even though he has lots of it in his closet, haha...), because he does not have that sense yet. He needs your help in properly classifying types of clothes possible to wear.

In this programming assignment, you are going to build an apparel classifier from images. Besides assisting Dr. B to enlightening him about cloth types, it has many practical applications in e-commerce and advertising.

Dataset

The following four data files are in compressed format. Be sure to use **LoadDataModule.py** to uncompress them and get the corresponding dataset into your workspace.

1. train_images.zip
    - 60,000 samples of 28 x 28 grayscale image. The data is of size 60000 x 784. Each pixel has a single intensity value which is an integer between 0 and 255.
2. train_labels.zip
    - 60,000 samples of 10 classes for the images in the given train_images. Class details are mentioned below.
3. test_images.zip
    - 10,000 samples of 28 x 28 grayscale image. The data is of size 10000 x 784. Each pixel has a single intensity value which is an integer between 0 and 255.
4. test_labels.zip
    - 10,000 samples from 10 classes for the images in the given test_images. Class details are listed below.

Class labels

Each training and test samples are assigned to one of the following labels:

Class	Type of dress
0	T-shirt / top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Task 1.1: Your own “Vanilla” Artificial Neural Network to classify clothes

1. Build a feed forward Neural network from scratch (i.e., without Tensorflow or Keras or similar computation frameworks) with the following specification:
    - Hidden layer 1: 784 neurons with hyperbolic tangent activation function in each neuron.
    - Hidden layer 2: 512 neurons, with sigmoid activation function in each of the neuron.
    - Hidden layer 3: 100 neurons, with rectified linear activation function in each of the neuron.
    - Output layer: 10 neurons representing the 10 classes, and obviously softmax activation function for each of the 10 neurons.
2. Seed the random number generator using 554433. Then shuffle the order of training samples.
3. Using Min-Max scaling to scale the training dataset and using the same Min and Max values from the training set scale the test dataset.
4. Using mini-batch gradient descent to optimize the loss function: “categorical cross-entropy” on the training dataset. Please run 50 epochs with batch size of 200. Also record the loss value for each of the 50 epochs. Record the wall time for each epoch. Generate a epoch-loss plot, and loss-wall time plot.
5. Evaluate the model on the test dataset, and

    (a) print classification accuracy in your report.

    (b) Print the 10-class confusion matrix and then print classwise accuracy, precision, recall and F1-measure.

Task 1.2: Repeat Task 1.1 with Keras.

Task 2.1: Your own Small Convolution Neural Network to classify clothes

1. Build a Convolution Neural network from scratch (i.e., without Tensorflow or Keras or similar computation frameworks) with the following specification:
    - Convolution layer having 40 feature detectors, with kernel size 5 x 5, and rectifier as the activation function, with stride 1 and no-padding.
    - A max-pooling layer with pool size 2x2.
    - Fully connected layer with 100 neurons, and rectifier as the activation function.
    - Output layer containing 10 neurons, softmax as activation function.
2. Seed the random number generator using 554433. Then shuffle the order of training samples.
3. Using Min-Max scaling to scale the training dataset and using the same Min and Max values from the training set scale the test dataset.
4. Using mini-batch gradient descent to optimize the loss function: “categorical cross-entropy” on the training dataset. Please run 50 epochs with batch size of 200. Also record the loss value for each of the 50 epochs. Record the wall time for each epoch. Generate a epoch-loss plot, and loss-wall time plot.
5. Evaluate the model on the test dataset, and
    (a) print classification accuracy in your report.
    (b) Print the 10-class confusion matrix and then print classwise accuracy, precision, recall and F1-measure.

Task 2.2: Repeat Task 2.1 with Keras

(Bonus, 10 pts possible) Task 3.1: Bigger convolution neural network to classify clothes

1. Build another Convolution Neural network from scratch (i.e., without Tensorflow or Keras or similar computation frameworks) with the following specification:
   (a) Convolution layer having 48 feature detectors, with kernel size 3 x 3, and rectifier as the activation function, with stride 1 and no padding.
   (b) A max-pooling layer with pool size 2x2.
   (c) Convolution layer having 96 feature detectors, with kernel size 3 x 3, and rectifier as the activation function, with stride 1 and no padding.
   (d) A max-pooling layer with pool size 2x2.
   (e) Fully connected layer with 100 neurons, and rectifier as the activation function.
   (f) Output layer containing 10 neurons, softmax as activation function.
2. Seed the random number generator using 554433. Then shuffle the order of training samples.
3. Using Min-Max scaling to scale the training dataset and using the same Min and Max values from the training set scale the test dataset.
4. Using mini-batch gradient descent to optimize the loss function: “categorical cross-entropy” on the training dataset. Please run 50 epochs with batch size of 200. Also record the loss value for each of the 50 epochs. Record the wall time for each epoch. Generate a epoch-loss plot, and loss-wall time plot.
5. Evaluate the model on the test dataset, and
   (a) print classification accuracy in your report.
   (b) Print the 10-class confusion matrix and then print classwise accuracy, precision, recall and F1-measure.

(Bonus, 5 pts possible) Task 3.2: Repeat Task 3.1 with Keras

Reference materials

- ConvNets at http://cs231n.github.io/convolutional-networks/
- Building classifiers for the MNIST (grayscale) dataset. (Jupyter notebook workspace file: test-example-MNIST.zip)
- Building classifiers for the CIFAR-10 (RGB) dataset. (Jupyter notebook workspace file: test-example-CIFAR10.zip)