**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.

**train\_images**

* 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.

**train\_labels**

* 60,000 samples of 10 classes for the images in the given train\_images.
* Class details are mentioned below.

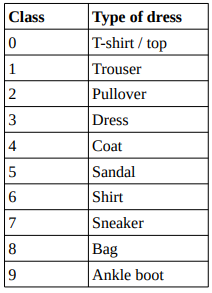
**test\_images**

* 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.

**test\_labels**

* 10,000 samples from 10 classes for the images in the given test\_images.
* Class details are listed below.

**Class Labels**



Build a Convolution Neural network with the following specification:

* conv1: Convolution layer having 2feature detectors, with kernel size 3 x 3, and sigmoid as the activation function, with stride 1 and no-padding.
* Pool1: A max-pooling layer with pool size 2x2.
* conv2: Convolution layer having 2 feature detectors, with kernel size 3 x 3, and rectified linear unit as the activation function, with stride 1 and no-padding.
* Pool2: A max-pooling layer with pool size 2x2.
* FC1: Fully connected layer with 50 neurons, and hyperbolic tangent as the activation
* function.
* Output: Output layer containing 10 neurons, softmax as activation function.

Split the given training dataset into 80% training & 20% test (a.k.a., validation set)

Build your CNN training algorithm in such a way that it saves the weights (i.e., feature detectors) in every single epoch so that you can retrieve best set of feature detectors when the simulation is over.

Fit the model above with the training dataset, with 20 epochs, minibatch size of 200.

Print Epoch-loss, epoch-accuracy plot for training and validation. (Note: validation data is not test data).

From the plot/history of every epoch, determine the best model parameters (i.e., the weights

(kernels, filters, feature detectors). Evaluate that model on the given test dataset, and

* print classification (base) error and accuracy in your report.
* Print the classification report.
* Print the confusion matrix, and also plot the confusion matrix which kind of look like heatmap.