GROUP 12 PHASE 3

CHOICE OF THE PRE-PROCESSING METHODS

All the images we have taken for this phase are square-shaped, i.e., they are in a 1: 1 ratio. We have observed that the images we used in the earlier phases had 28 x 28 resolution. So initially, since we could not take pictures that are exactly 28 by 28, we just made sure that the pictures that are taken are square, which means that they are taken in a 1:1 ratio, as mentioned above. But given that the neural network has been trained on 28 x 28-pixel images, our images had to be converted to 28 by 28; otherwise, the network would not work since those dimensions were kept in mind while computing derivatives in backward pass and weights update. So, after reading the images, we rescale them using cv2 to 28 by 28, and then we flattened them to get the shape of my_X_train as 50 by 784, where 50 is the number of images and 784 is the number of pixels of 1 image.

Moreover, we observed that the images provided in the earlier phases consistently had a black background and the objects in front were white in color. Our images were not in the same pattern. Most of them had a white background, and the objects themselves were arbitrarily colored. For this, we used grayscale conversion and inversion to make sure that the images are consistent with the ones in the earlier phases. We could have used cv2 to read as grayscale and invert them, but we simply converted them to grayscale and changed their backgrounds beforehand, so we did not have to do that in our code.

CHOICE OF THE HYPERPARAMETERS AND NETWORK

Here, we tried several different possibilities, and we kept our learning rate rather high initially. At this, the entropy loss started to increase, and we know that in order to make sure that it decreases, we had to decrease the learning rate. At first, we started from lower EPOCHS, and then we kept increasing them since we noticed that the loss curve was going downwards and the accuracy was increasing. In phase 2, we ran our Neural Network at 140 EPOCHS and 0.001 learning rate with 2 hidden layers with nodes 150 and 75, and we got an accuracy slightly higher than 85. But in this phase, due to time constraints, we only ran it for 20 epochs, and a 0.01 learning rate with 2 hidden layers of 150 and 75 nodes each, and we got 85% accuracy for these parameters as well.

In short, we chose our hyperparameters randomly, through the trial-and-error method. Afterward, we looked at the entropy loss curve and adjusted the hyperparameters accordingly

RESULTS – PHASE-3

The accuracy we got for our own dataset was quite low, 10% to be exact, and there are various reasons for that. First of all, the FashionMNIST dataset has images with a perfectly black background and very clear objects. On the other hand, the pictures on our dataset are shot in a non-optimal environment and do not have a clear background. Therefore, after converting them to greyscale and inverting them, the images have a tint of white on their background, and the objects, on some pictures, are not clearly visible. We also took pictures from our phones in 1:1 mode, which doesn't ensure that the images are 28 by 28, so when we rescaled them, the image got pixelated and slightly blurred. As our NN is trained on fashionMINST dataset, it struggles to recognize pictures in our dataset. Neural Networks, in general, are not very robust and can be easily fooled, so provided that the images were a blur and not clearly visible and not exactly similar in terms of properties to the fashion-MNIST dataset, the neural network fails to recognize most of them. So, in short, since the test images are not exactly matchable to the train dataset, the NN fails to recognize those images.

To improve our results, we could have shot pictures with a plain background. Moreover, we could have used a professional camera and tweaked the white balance, noise, etc., of the images using applications like photoshop to make our images similar to the ones in fashionMINST. We could have also made sure that no pixelation occurs in our pictures. Apart from that, since Neural Networks are very basic and not very robust, we could've used a CNN instead, which is a perfect option for image classification and can provide accuracies up to 99.7% on training data and hopefully a higher accuracy on our data as well.