OVERVIEW OF WHAT I HAVE DONE:

- I implemented the project on two architectures, deep neural network(DNN) and convolutional neural network(CNN) in colab using pytorch.
- For both of these models I trained the networks with training data and verified on validation set(part of training data).
- After trying for different model parameters for each network to improve results I saved my model.
- To test the saved model and to generate output files I wrote another code which I run it from cmd.

HOW I CONSIDERED ARCHITECTURES:

- The motivation going for DNN is better extraction of pattern(features) from the input compared with Neural Network.
- First I implemented DNN and by changing the number of layers and hidden units observed the validation accuracy, didn't get better results even for ReLu activation and ADAM optimizer. Because we are flattening the image in the DNN
- So to extract the features of the image as it is I went for CNN, it has shift invariant property (because of convolution) and will preserve the edges and orientation of the image.
- CNN has different models but I saved my model on simple LeNet structure because other models have many layers inside, so they are not running on my cpu, but I verified some models(AlexNet, VGG-16) in colab.

HOW I CHOOSED HYPERPARAMETERS:

- For better extraction of the input, Instead of having one layer with many hiiden units I considered enough number of layers with enough hidden units in DNN for the corresponding input.
- For CNN as I mentioned I consider LeNet structure(2 convolutional and 3 fully connected) but other models AlexNet, VGG -16 are slightly increasing the accuracy(around 3-4 %) but they have more layers compared to LeNet.
- I used ReLu as my activation function because it will reduce the likelihood of gradient to vanish, it is computationally more efficient(because of linearity in the positive part) and it gives zero output for negative values makes the output sparse.
- I used input data normalization and batch normalization . This will ensure that inputs to the first layer have zero mean and come from the same distribution, while Batch Normalization on subsequent layers will ensure that inputs to those layers have zero mean in expectation and that their distributions do not drift over time.
- I changed number of epochs, batchsize but didn't see much difference in results, but rate of convergence becoming slow when I increased epochs.

- I used ADAM optimizer because it is formed by considering the advantages of the previous methods(RmsProp,AdaGrad).
- I considered crossentropy loss because it gives best results by comparing the distributions of a classification problem.

VALIDATION PROCESS:

- Once the training is done I am applying my validation data to the model and taking the maximum value index of the returned information from the model.
- After that I am comparing the true labels, predicted labels and summing the truths.
- Finding accuracy by considering total labels and correctly classified labels.

RESULTS:

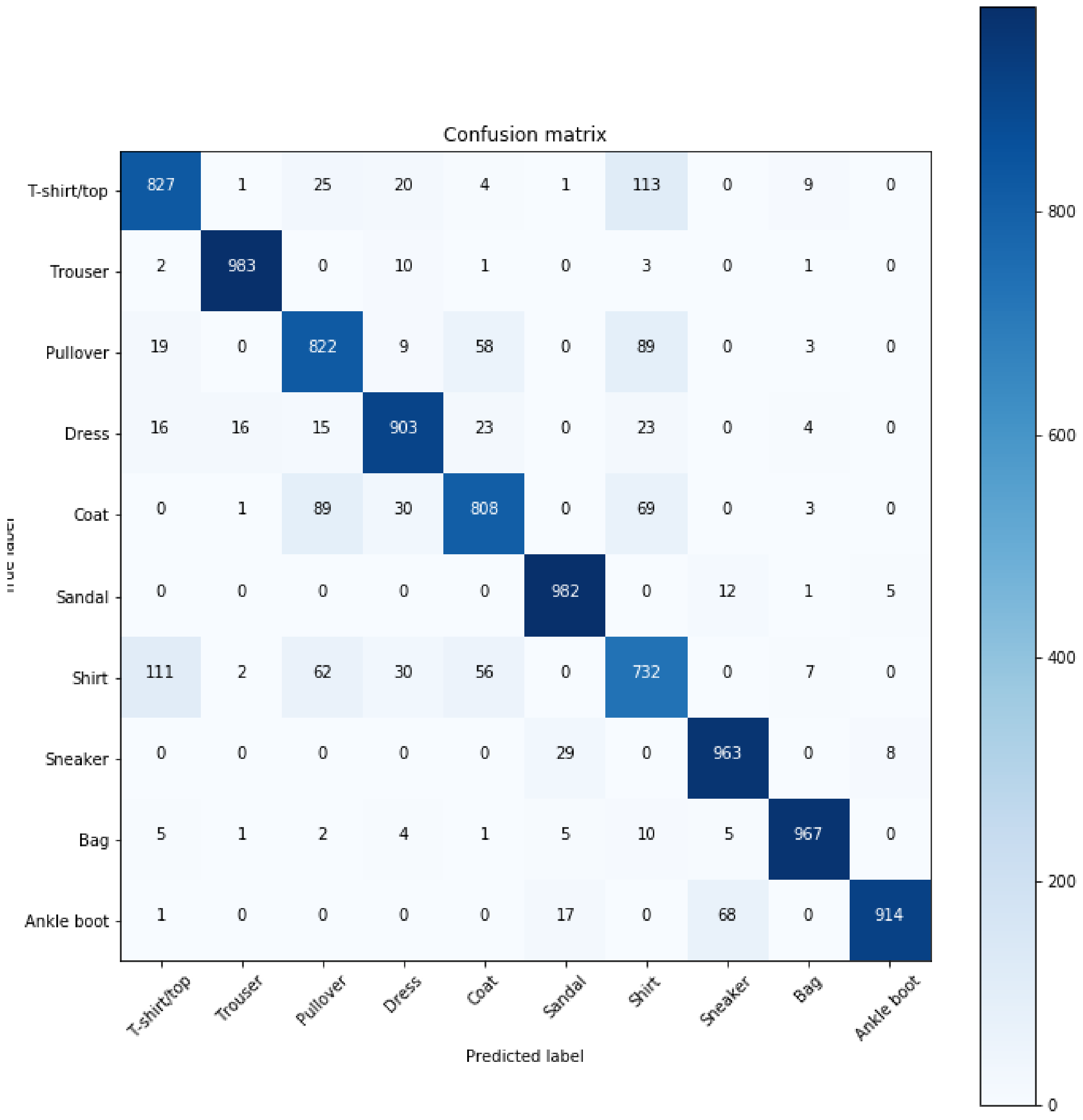
I got 89 to 90 percent accuracy on DNN & 90 to 93.5 percent accuracy on CNN.

Following are the few numbers I noted for CNN

Number of convolutional layers	Kernel size, padding	Epochs	Batchsize	Train accuracy(%)	Test accuracy(%)
2	3*3,1	20	128	95	90.12
2	3*3,1	40	64	96.12	90.97
4	3*3,1	40	128	98.1	92.04
5	3*3,1	40	64	97.374	93.36
5	3*3,1	40	128	97.46	93.28

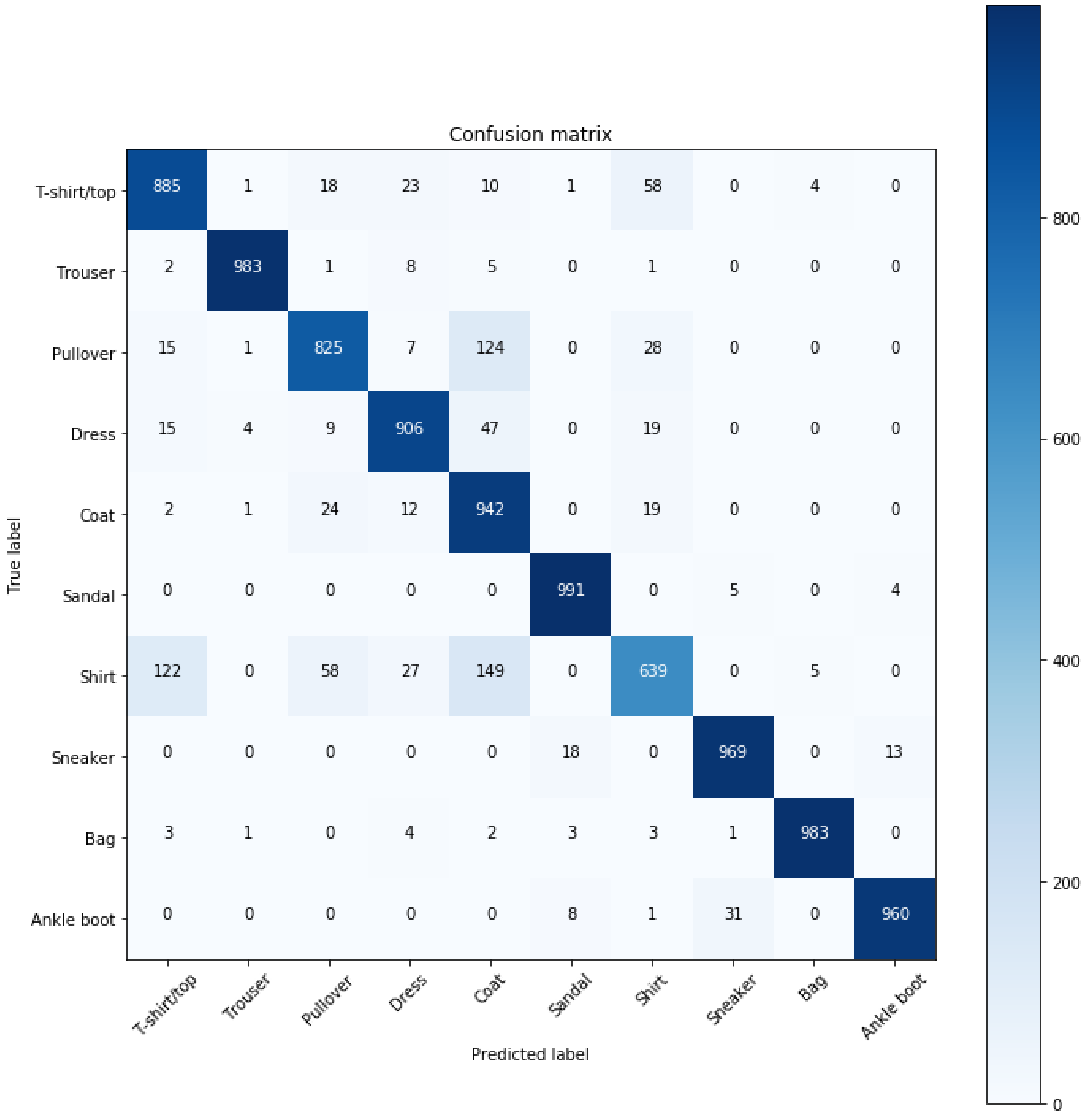
- When I used more number of layers with more epochs, train accuracy increasing (purposefully overfitting the model) and I put dropout for two layers with 25% probability,I observed slight increase in accuracy because it will help to leave some nodes to regularize overfitting problem.
- When I put dropout for all layers the model is underfitting.

CONFUSION MATRIX FOR DNN:



	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt/top	827	1	25	20	4	1	113	0	9	0
Trouser	2	983	0	10	1	0	3	0	1	0
Pullover	19	0	822	9	58	0	89	0	3	0
Dress	16	16	15	903	23	0	23	0	4	0
Coat	0	1	89	30	808	0	69	0	3	0
Sandal	0	0	0	0	0	982	0	12	1	5
Shirt	111	2	62	30	56	0	732	0	7	0
Sneaker	0	0	0	0	0	29	0	963	0	8
Bag	5	1	2	4	1	5	10	5	967	0
Ankle boot	1	0	0	0	0	17	0	68	0	914

CONFUSION MATRIX FOR CNN:



	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt/top	885	1	18	23	10	1	58	0	4	0
Trouser	2	983	1	8	5	0	1	0	0	0
Pullover	15	1	825	7	124	0	28	0	0	0
Dress	15	4	9	906	47	0	19	0	0	0
Coat	2	1	24	12	942	0	19	0	0	0
Sandal	0	0	0	0	0	991	0	5	0	4
Shirt	122	0	58	27	149	0	639	0	5	0
Sneaker	0	0	0	0	0	18	0	969	0	13
Bag	3	1	0	4	2	3	3	1	983	0
Ankle boot	0	0	0	0	0	8	1	31	0	960