Assignment 4

Both tasks share common script 'script_load_data.m' which was used for loading the data.

In the script I just load the data and permute it randomly.

For 'nrptool task' I also one-hot encode the data. I also create 3x time larger dataset were 2/3 has added gaussian noise.

I also thought about flipping or rotating the images, but because they are just represented as array, I thought it would make no difference...

Task 1: nftool

I tried mulitple number of neurons in the single hidden layer - 10, 15, 20, 30, 50 and 100.

At first I used Levenberg-Marquardt backpropagation, but it was extremely slow an I was not able to run it on more than 30 neurons.

Then I switched to the Scaled conjugate gradient backpropagation and it was very fast...

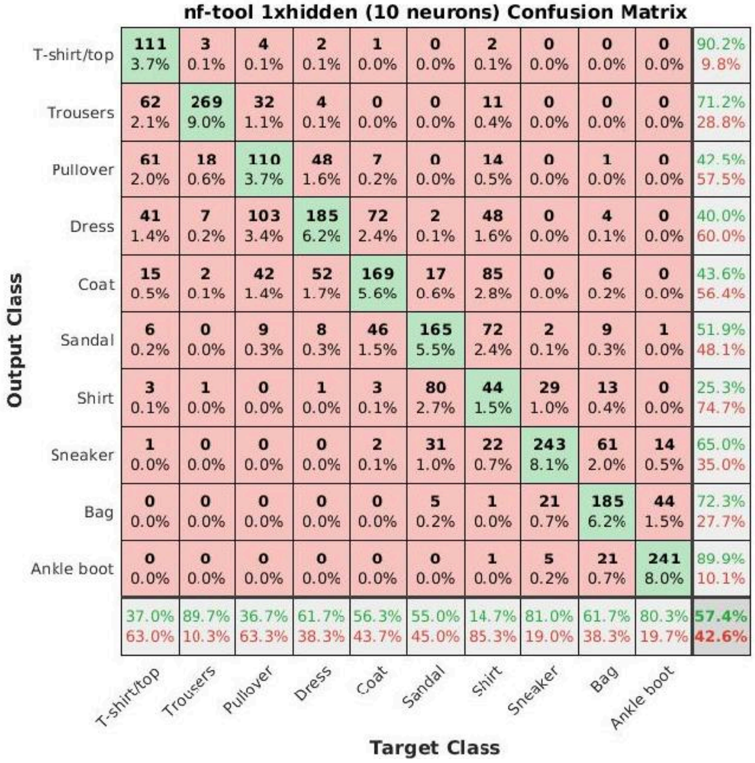
This table sumarizes the accuracy results on the test data set:

neurons	**trainscg**	**trainlm**
10	**0.5740**	0.5343
15	0.5050	0.4877
20	0.5240	0.3460
30	0.4913	0.3500
50	0.5523	-
100	0.5172	-

For some reason trainscg led to much better results even when it is simpler method.

We can see that best resutls were achieved with 10 neruons and Scaled conjugate gradient backpropagation.

Confusion matrix for the default (20 neurons) and for the best (10 neurons) is in the files '**nf-tool-20.jpg**' and '**nf-tool-10.jpg**' respectively. Best network is saved in the file '**nf-tool-10.mat**'.

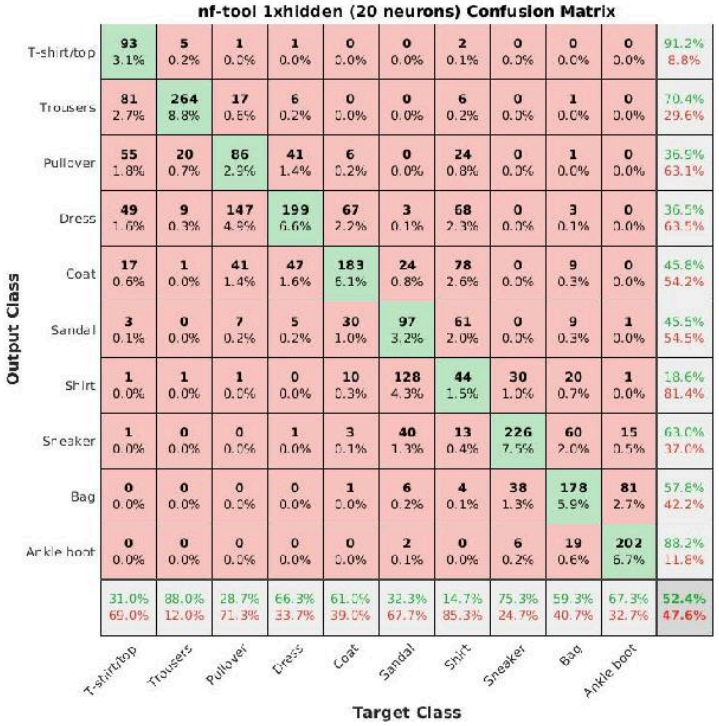


nf-tool 1xhidden (10 neurons) Confusion Matrix

Output Class

	T-shirt/top	Trousers	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	
T-shirt/top	111 3.7%	3 0.1%	4 0.1%	2 0.1%	1 0.0%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	90.2% 9.8%
Trousers	62 2.1%	269 9.0%	32 1.1%	4 0.1%	0 0.0%	0 0.0%	11 0.4%	0 0.0%	0 0.0%	0 0.0%	71.2% 28.8%
Pullover	61 2.0%	18 0.6%	110 3.7%	48 1.6%	7 0.2%	0 0.0%	14 0.5%	0 0.0%	1 0.0%	0 0.0%	42.5% 57.5%
Dress	41 1.4%	7 0.2%	103 3.4%	185 6.2%	72 2.4%	2 0.1%	48 1.6%	0 0.0%	4 0.1%	0 0.0%	40.0% 60.0%
Coat	15 0.5%	2 0.1%	42 1.4%	52 1.7%	169 5.6%	17 0.6%	85 2.8%	0 0.0%	6 0.2%	0 0.0%	43.6% 56.4%
Sandal	6 0.2%	0 0.0%	9 0.3%	8 0.3%	46 1.5%	165 5.5%	72 2.4%	2 0.1%	9 0.3%	1 0.0%	51.9% 48.1%
Shirt	3 0.1%	1 0.0%	0 0.0%	1 0.0%	3 0.1%	80 2.7%	44 1.5%	29 1.0%	13 0.4%	0 0.0%	25.3% 74.7%
Sneaker	1 0.0%	0 0.0%	0 0.0%	0 0.0%	2 0.1%	31 1.0%	22 0.7%	243 8.1%	61 2.0%	14 0.5%	65.0% 35.0%
Bag	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5 0.2%	1 0.0%	21 0.7%	185 6.2%	44 1.5%	72.3% 27.7%
Ankle boot	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	5 0.2%	21 0.7%	241 8.0%	89.9% 10.1%
	37.0% 63.0%	89.7% 10.3%	36.7% 63.3%	61.7% 38.3%	56.3% 43.7%	55.0% 45.0%	14.7% 85.3%	81.0% 19.0%	61.7% 38.3%	80.3% 19.7%	**57.4%** **42.6%**

Target Class



nf-tool 1xhidden (20 neurons) Confusion Matrix

Output Class

	T-shirt/top	Trousers	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	
T-shirt/top	93 3.1%	5 0.2%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	91.2% 8.8%
Trousers	81 2.7%	264 8.8%	17 0.6%	6 0.2%	0 0.0%	0 0.0%	6 0.2%	0 0.0%	1 0.0%	0 0.0%	70.4% 29.6%
Pullover	55 1.8%	20 0.7%	86 2.9%	41 1.4%	6 0.2%	0 0.0%	24 0.8%	0 0.0%	1 0.0%	0 0.0%	36.9% 63.1%
Dress	49 1.6%	9 0.3%	147 4.9%	199 6.6%	67 2.2%	3 0.1%	68 2.3%	0 0.0%	3 0.1%	0 0.0%	36.5% 63.5%
Coat	17 0.6%	1 0.0%	41 1.4%	47 1.6%	183 6.1%	24 0.8%	78 2.6%	0 0.0%	9 0.3%	0 0.0%	45.8% 54.2%
Sandal	3 0.1%	0 0.0%	7 0.2%	5 0.2%	30 1.0%	97 3.2%	61 2.0%	0 0.0%	9 0.3%	1 0.0%	45.5% 54.5%
Shirt	1 0.0%	1 0.0%	1 0.0%	0 0.0%	10 0.3%	128 4.3%	44 1.5%	30 1.0%	20 0.7%	1 0.0%	18.6% 81.4%
Sneaker	1 0.0%	0 0.0%	0 0.0%	1 0.0%	3 0.1%	40 1.3%	13 0.4%	226 7.5%	60 2.0%	15 0.5%	63.0% 37.0%
Bag	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	6 0.2%	4 0.1%	38 1.3%	178 5.9%	81 2.7%	57.8% 42.2%
Ankle boot	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	6 0.2%	19 0.6%	202 6.7%	88.2% 11.8%
	31.0% 69.0%	88.0% 12.0%	28.7% 71.3%	66.3% 33.7%	61.0% 39.0%	32.3% 67.7%	14.7% 85.3%	75.3% 24.7%	59.3% 40.7%	67.3% 32.7%	**52.4%** **47.6%**

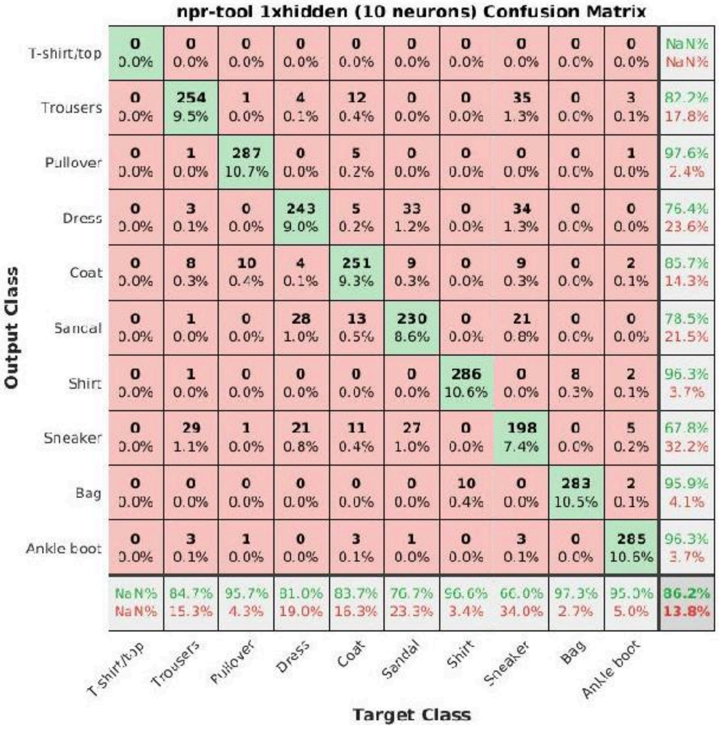
Target Class

Task 2: nrptool

I tried a lot of variants of the neurons and layers. Finally the best accuracy **0.8625** was acheived with two hidden layers with 150 and 100 neuros.

As described above, I also modified the dataset by adding noise. Though it did helped only just a little bit. Maybe I should add more noise...

Confusion matrix for the best network is in the file '**nrp-tool-150-100.jpg**' trained network is in the file '**nrp-tool-150-100.mat**'



npr-tool 1xhidden (10 neurons) Confusion Matrix

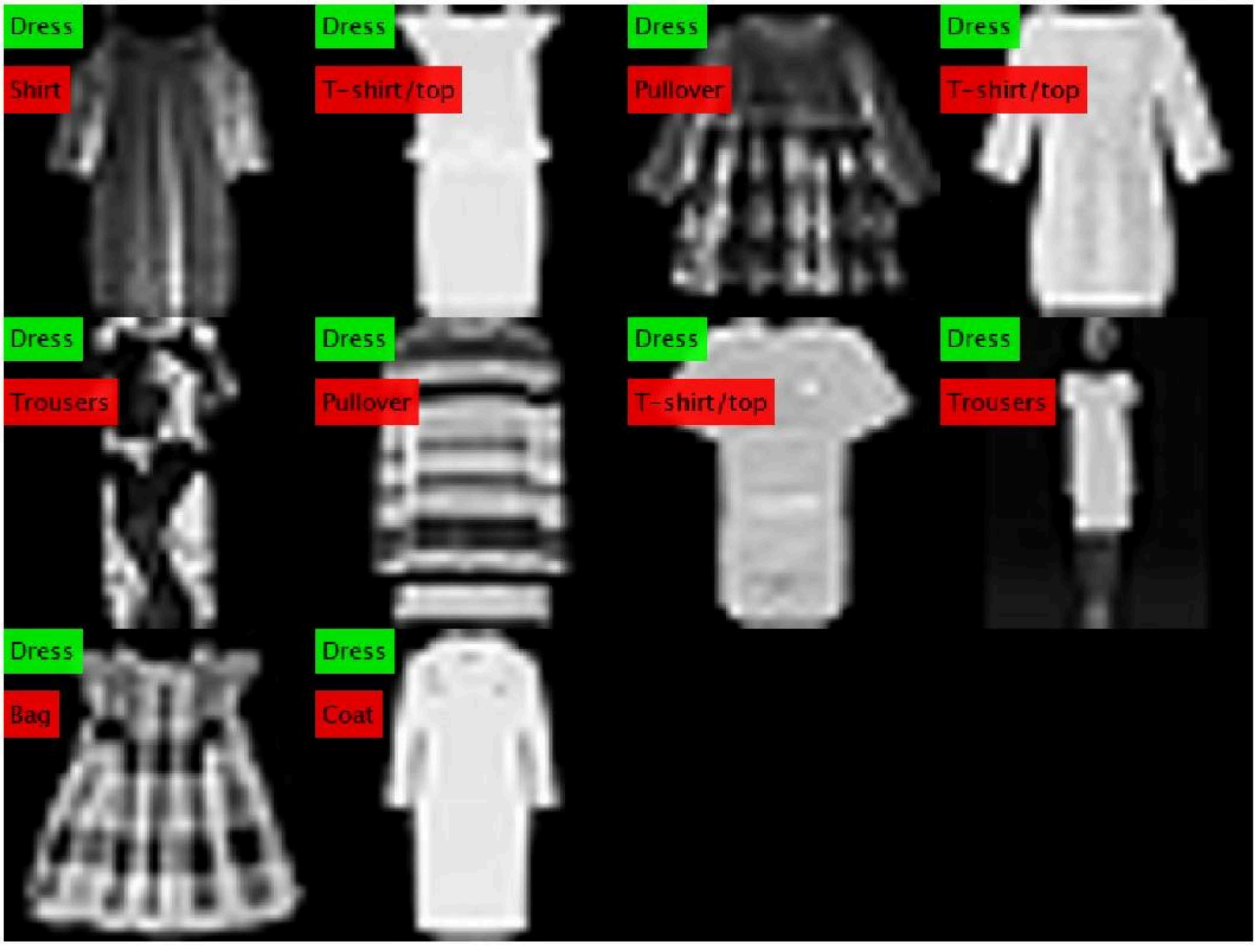
Output Class \ Target Class	T-shirt/top	Trousers	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	
T-shirt/top	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
Trousers	0 0.0%	254 9.5%	1 0.0%	4 0.1%	12 0.4%	0 0.0%	0 0.0%	35 1.3%	0 0.0%	3 0.1%	82.2% 17.8%
Pullover	0 0.0%	1 0.0%	287 10.7%	0 0.0%	5 0.2%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	97.6% 2.4%
Dress	0 0.0%	3 0.1%	0 0.0%	243 9.0%	5 0.2%	33 1.2%	0 0.0%	34 1.3%	0 0.0%	0 0.0%	76.4% 23.6%
Coat	0 0.0%	8 0.3%	10 0.4%	4 0.1%	251 9.3%	9 0.3%	0 0.0%	9 0.3%	0 0.0%	2 0.1%	85.7% 14.3%
Sandal	0 0.0%	1 0.0%	0 0.0%	28 1.0%	13 0.5%	230 8.6%	0 0.0%	21 0.8%	0 0.0%	0 0.0%	78.5% 21.5%
Shirt	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	286 10.6%	0 0.0%	8 0.3%	2 0.1%	96.3% 3.7%
Sneaker	0 0.0%	29 1.1%	1 0.0%	21 0.8%	11 0.4%	27 1.0%	0 0.0%	198 7.4%	0 0.0%	5 0.2%	67.8% 32.2%
Bag	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	10 0.4%	0 0.0%	283 10.5%	2 0.1%	95.9% 4.1%
Ankle boot	0 0.0%	3 0.1%	1 0.0%	0 0.0%	3 0.1%	1 0.0%	0 0.0%	3 0.1%	0 0.0%	285 10.6%	96.3% 3.7%
	NaN% NaN%	84.7% 15.3%	95.7% 4.3%	81.0% 19.0%	83.7% 16.3%	76.7% 23.3%	96.6% 3.4%	66.0% 34.0%	97.3% 2.7%	95.0% 5.0%	**86.2%** **13.8%**

Here are 10 examples of wrongly classified images (green is correct class, red is prediction...)

We can see, that one can be easily confused because the image is simaler to both correct and predicted labels.

Dres and Shirts are very similar.

Quite funny is missclassification of Dress and trousers. I gues it is because it is very long.


Dress
Shirt
Dress
T-shirt/top
Dress
Pullover
Dress
T-shirt/top
Dress
Trousers
Dress
Pullover
Dress
T-shirt/top
Dress
Trousers
Dress
Bag
Dress
Coat