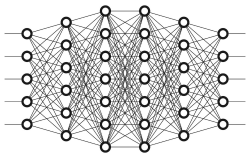


Massachusetts Institute of Technology
Dept. of Electrical Engineering and Computer Science
Fall Semester, 2018
MIT 6.S198: Deep Learning Practicum

Assignment 2: Multilayer Networks

1 - Building Models with Model Builder

Task 1.1: You should see in the leftmost column of Model Builder that the initial model is marked “Invalid model”. Why is the model invalid? Write up your answer on your webpage for this assignment.

Answer 1.1: The model is invalid because the dimensions do not match. The input image is [28,28,1] and the input for Softmax is [10] so the dimensions need to be adjusted

Task 1.2: We can make the model valid by adding two more layers. Click on the “Op type” menu in the center of the first new layer, and select “Flatten”. The flatten layer takes its input, which is is a 3-D tensor of shape [28,28,1], and flattens it into a 1-D vector x=[x1,x2,...,x784] of shape 28×28×1=[784]. The second added later is a Fully Connected layer with 10 hidden units. “Hidden” means that the units are neither input nor output. “Fully connected” means that each unit is connected to all 784 inputs and all 10 outputs.

The classifications you are seeing are almost always wrong. Why is this? What performance should you expect from this particular network, i.e., how often should you expect it to be correct? Is this what you observe? *Hint:* The weights were initialized to random values. Write up your answers to these questions in your webpage.

Answer 1.2: The weights are initialized to random values. As such we should expect the performance to match that of a random guesser which is about 10% accuracy. The software does not give statistics before being trained but after 1 run it has an accuracy of 17% so the initial accuracy should have been around 10%

2 - Training Models

Task 2.1: What accuracy do you observe in training MNIST? How many inferences per second does the demo perform? How many examples per second does it train? Then try the same thing with Fashion MNIST and document your findings.

Answer 2.1: The MNIST data set has an accuracy of 90% and does 2270 inferences per second. It trains 1760 examples per second. The Fashion MNIST data set has an accuracy of 82% and does 2120 inferences per second. It trains 1620 examples per second.

Task 2.2: Change the Dataset to CIFAR-10. This will take about 30 seconds to load, due to the large number of images. Change the model back to Custom and add the flatten and fully connected layers as above. What accuracy do you observe in training CIFAR-10 after letting it train for a minute or two. You should find that it's a lot worse than for MNIST. We'll talk about why performance is bad when we discuss convolutional networks.

Answer 2.2: The CIFAR 10 data set has an accuracy of 38% and does 1120 inferences per second. It trains 1700 examples per second.

Task 2.3: Changing back to MNIST, let's consider a simple idea for improving accuracy, which turns out not to work: just add more fully connected units, one on top of the other. Add a new layer to the model so that the sequence of layers becomes: Input → Flatten → FC(10) → FC(10) → Softmax → Label. Start training and you should see the accuracy plummet to zero, with terrible results. What's going on? Hint: Notice that many of the probabilities will print as *Nan%*. Document these results and write up your ideas for why this happens.

Answer 2.3: The train accuracy is very low because the double stacked fully connected layers lead to exploding gradients. This ruins the accuracy of the network as it causes the updates to be too large.

3 - Activation Layers

Task 3: Return to the MNIST model so that the sequence of layers becomes: Input → Flatten → FC(10) → ReLU → FC(10) → Softmax → Label. Train the new model. How well does it perform? Then make the first FC model wider by increasing the number of units to 100. Does this make a difference? Document the results for these questions on your webpage.

Answer 3: The MNIST dataset with an extra FC(10) has an accuracy of 8% and does 1750 inferences per second. It trains 1240 examples per second. With the updated FC(100), the MNIST dataset has an accuracy of 97% and does 1450 inferences per second. It trains 1170 examples per second.

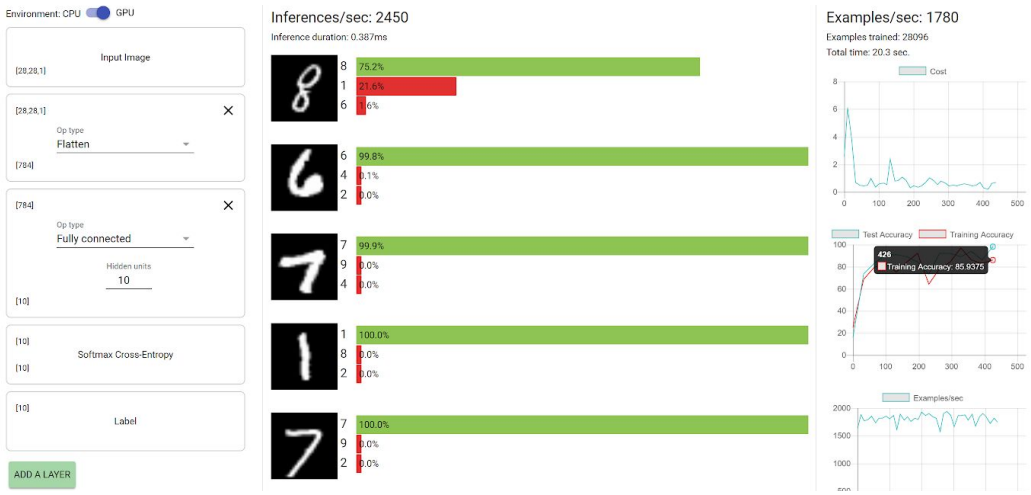
4 - Exploring with Model Builder

Testing vs. Validation

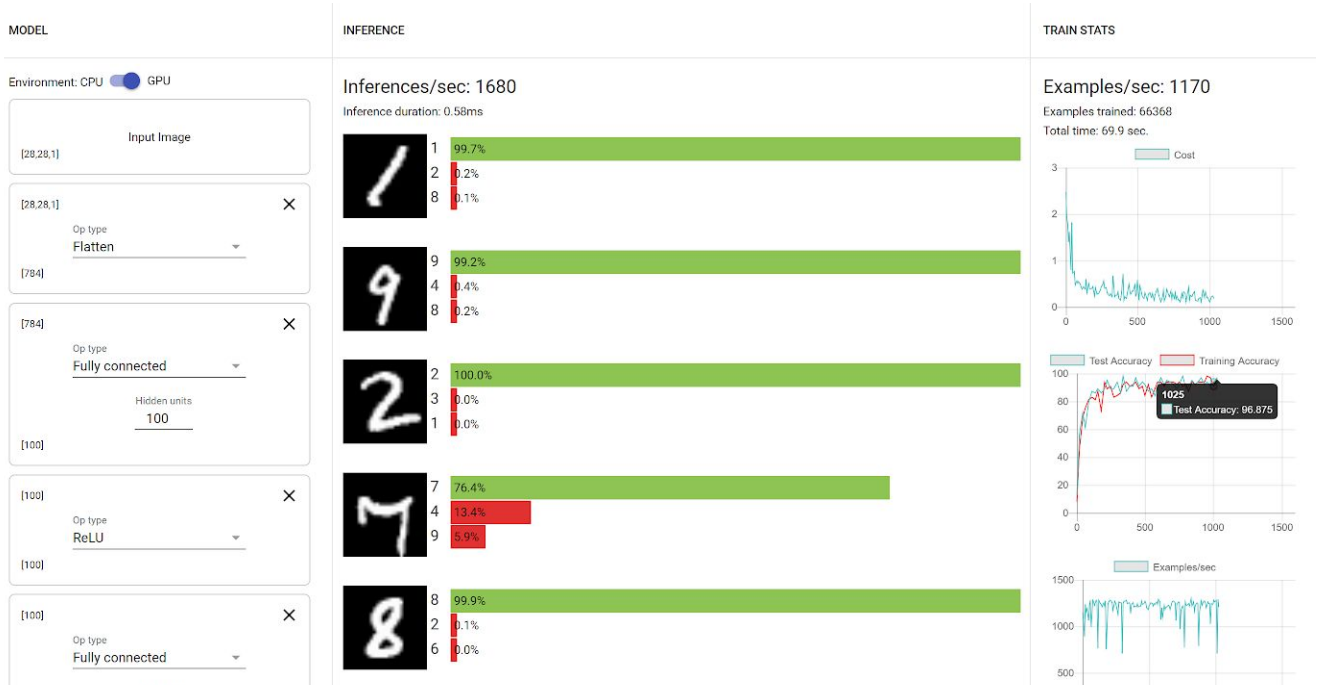
Appendix 2 (optional) has some general comments on training that you might find helpful.

Task 4-1: Train your MNIST model with 1,2,3,4, and 5 FC layers, with ReLU between them. For each, use the same hyperparameters, and the same number of hidden units (except for the last layer). What were the training times and accuracy? Do you see any overfitting? What can you conclude about how many layers to use? Include screenshots of the Training Stats for each of your examples.

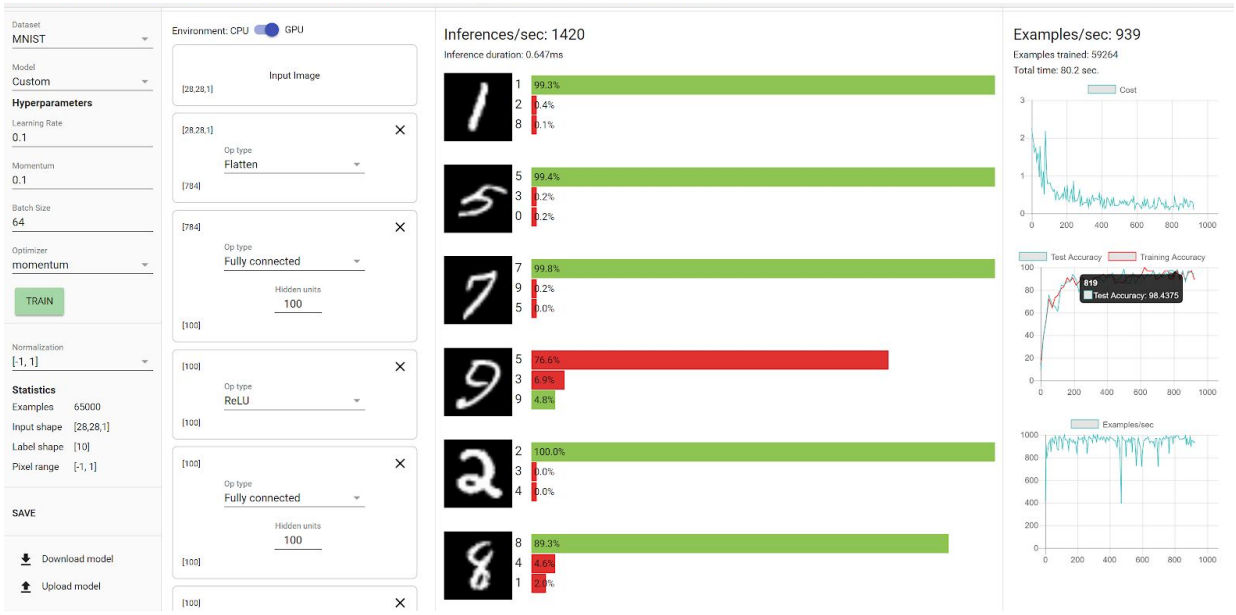
Answer 4-1: With 1xFC, the MNIST dataset has an accuracy of 85% and is trained in about 20.3 seconds. I do not see any overfitting.



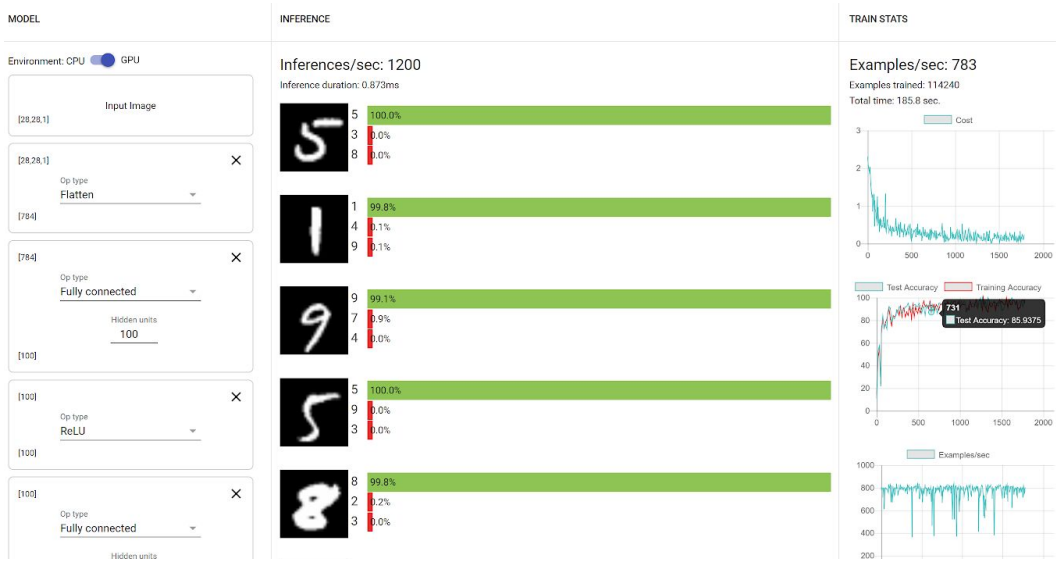
With 2xFC, the MNIST dataset has an accuracy of 96% and is trained in about 70 seconds. I do not see any overfitting.



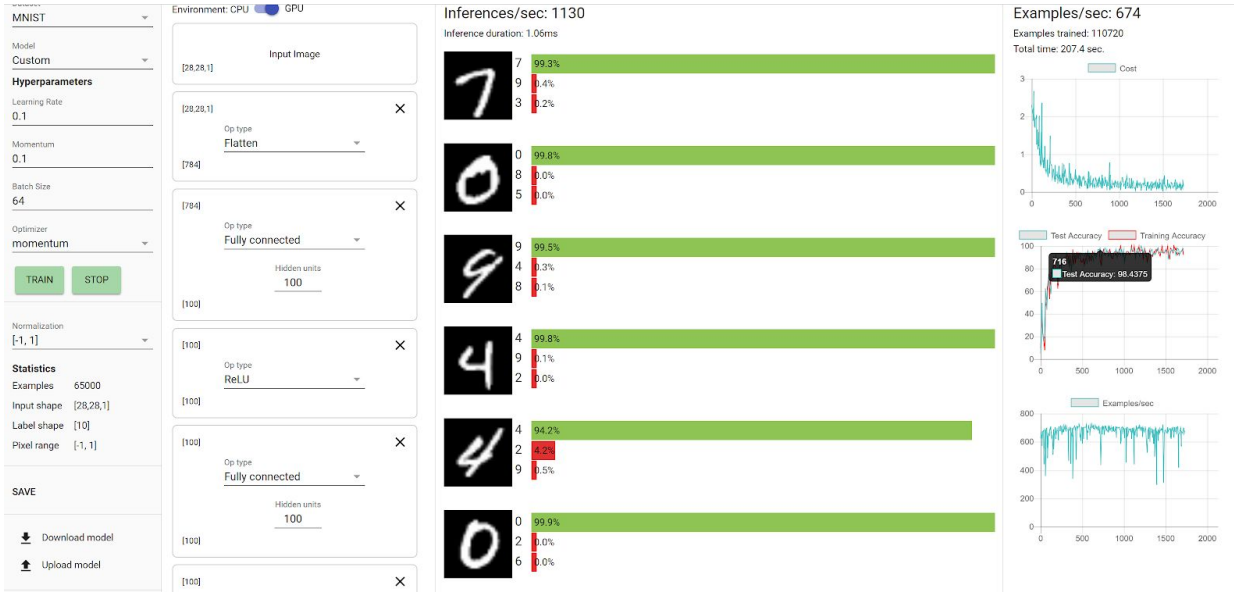
With 3xFC, the MNIST dataset has an accuracy of 98% and is trained in about 80 seconds. I do not see any overfitting.



With 4xFC, the MNIST dataset has an accuracy of 85% and is trained in about 180 seconds. I am not sure if there are mild signs of overfitting.



With 5xFC, the MNIST dataset has an accuracy of 98% and is trained in about 200 seconds. I do not think there is over fitting.

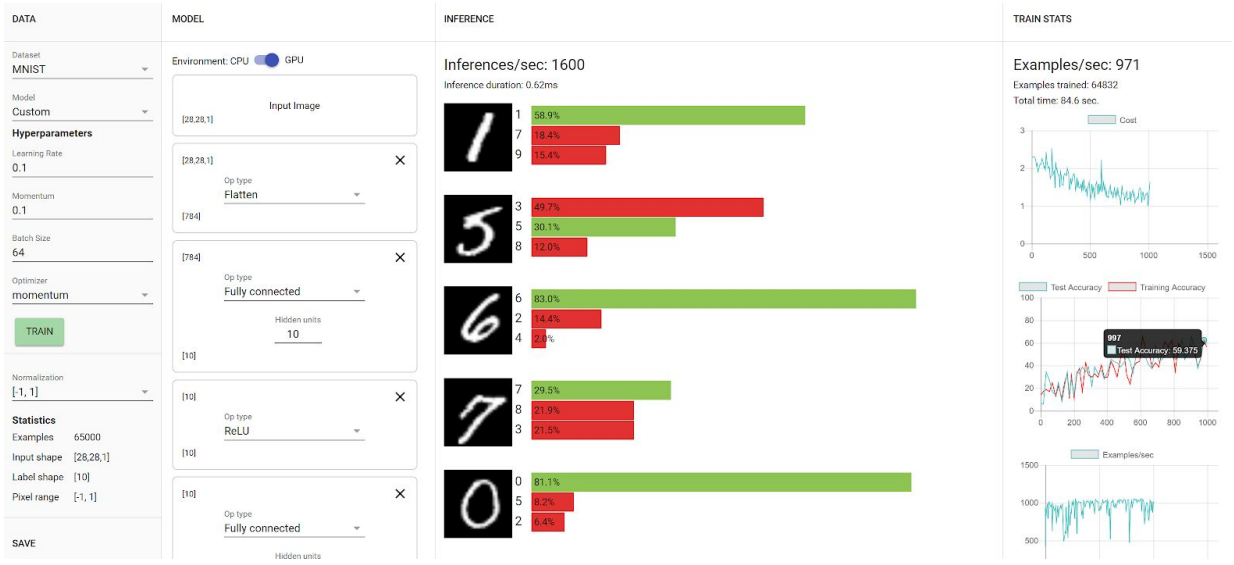


Overall I do not think there are real signs of overfitting but because of the speed differences I would use FCx2.

Task 4-2: Build a model with 3 FC layers, with ReLU between them. Try making the first layer wide and the second narrow, and vice versa, using the same hyperparameters as before. Which performs better? Why do you think this is?

With FC(100), FC(10), FC(10) accuracy is 95%

With FC(10), FC(100), FC(10) accuracy is 60%



I think that the first performs better because after the flatten layer there are 784 data points and FC(10) shrinks that to 10 data points so it does not make sense to then raise that to 100 when there was less information in the previous level.

Task 4-3: Try the same experiments with Fashion MNIST and CIFAR-10. Do you get similar results?

Answer 4-3: With 1xFC, the Fashion MNIST dataset has an accuracy of 90% and is trained in about 30 seconds. I do not see any overfitting.

With 2xFC, the Fashion MNIST dataset has an accuracy of 85% and is trained in about 30 seconds. I do not see any overfitting.

With 3xFC, the Fashion MNIST dataset has an accuracy of 82% and is trained in about 30 seconds. I do not see any overfitting.

With 4xFC, the Fashion MNIST dataset has an accuracy of 85% and is trained in about 30 seconds. I do not see any overfitting.

With 5xFC, the Fashion MNIST dataset has an accuracy of 80% and is trained in about 30 seconds. I do not see any overfitting.

The results here are interesting and show that we do not have a great variation in results as we increase the number of FC layers. If anything there is a slight decrease in accuracy as the number of layers increase.

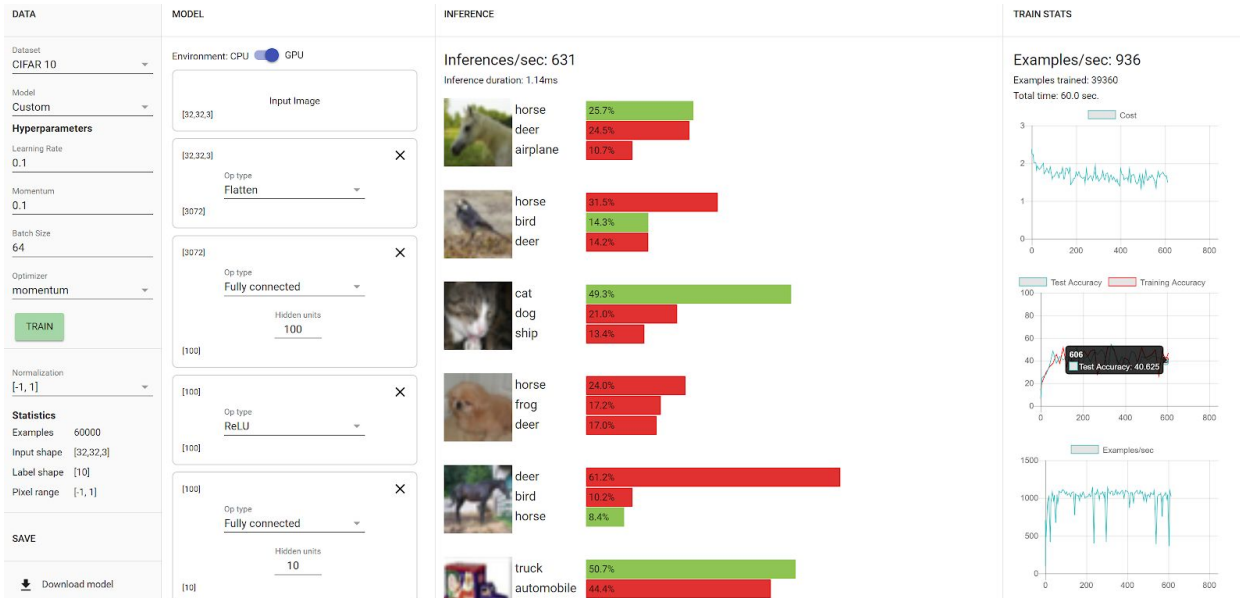
With FC(100), FC(10), FC(10) the accuracy of Fashion MNIST is 82%

With FC(10), FC(100), FC(10) the accuracy of Fashion MNIST is 82%

The results show now difference which is very interesting and I do not have any reason why this may be true.

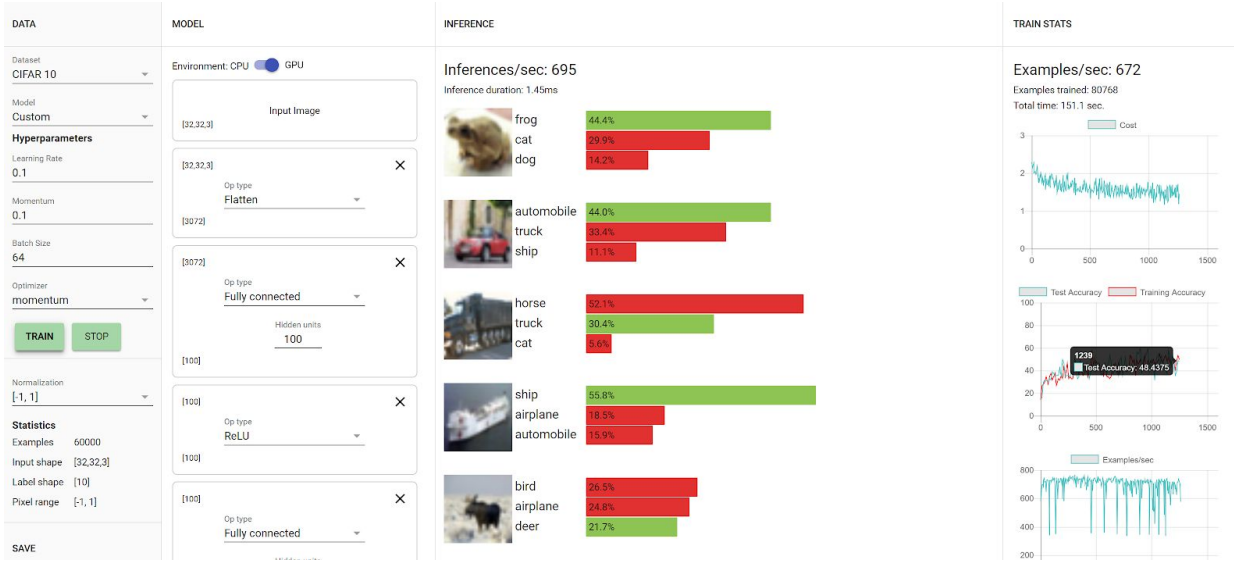
With 1xFC, the Fashion CIFAR 10 dataset has an accuracy of 40%.

With 2xFC, the Fashion CIFAR 10 dataset has an accuracy of 40%.

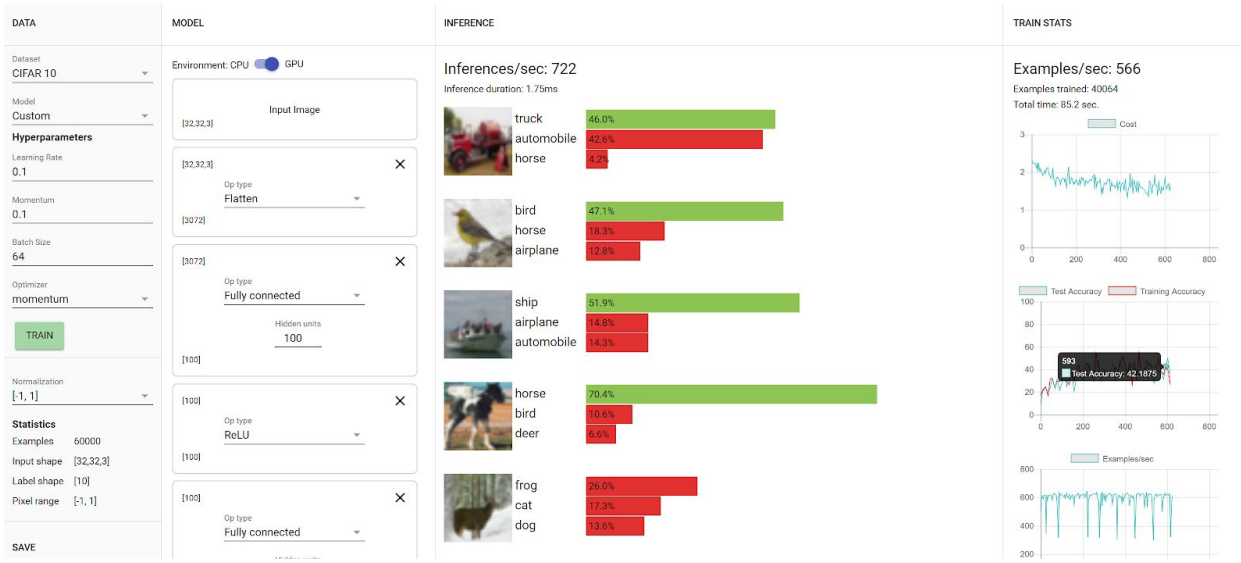


With 3xFC, the Fashion CIFAR 10 dataset has an accuracy of 38%.

With 4xFC, the Fashion CIFAR 10 dataset has an accuracy of 40%.



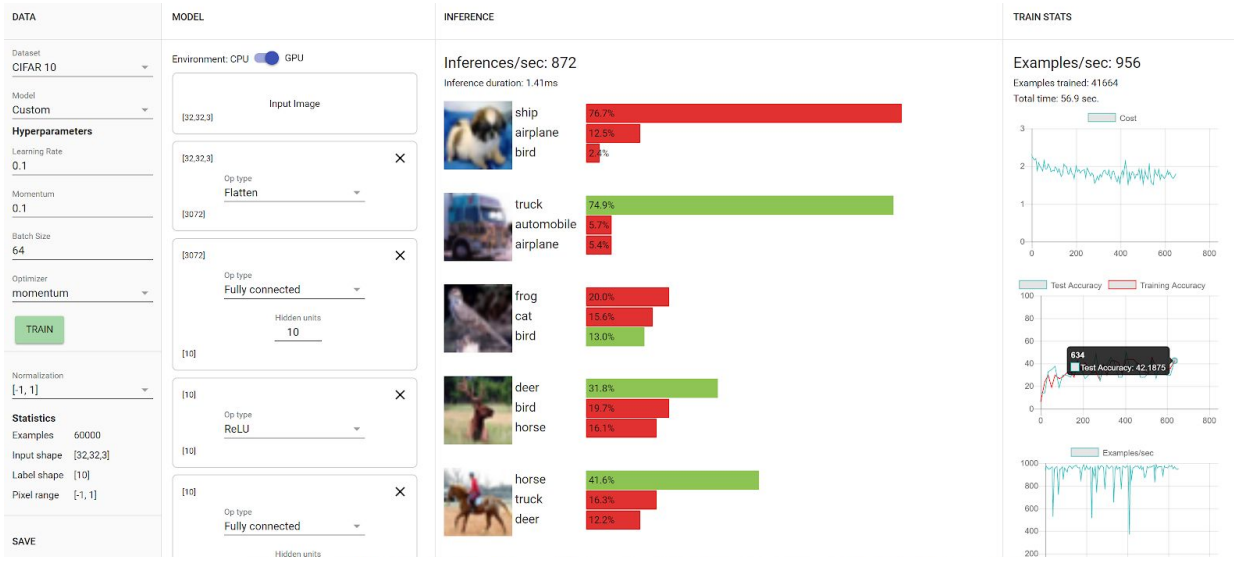
With 5xFC, the Fashion CIFAR 10 dataset has an accuracy of 38%.



Here the data is really interesting as there is no difference in the accuracy scores of the different models. This may also be a result of not training for enough time.

With FC(100), FC(10), FC(10) the accuracy of CIFAR 10 is 42%

With FC(10), FC(100), FC(10) the accuracy of CIFAR 10 is 42%



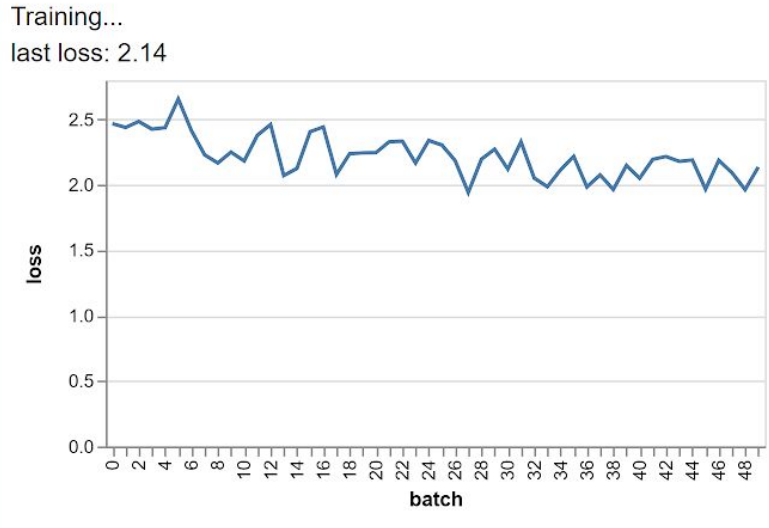
Here the data is really interesting as there is no difference in the accuracy scores of the different models. This may also be a result of not training for enough time.

5 - Multilayer Models

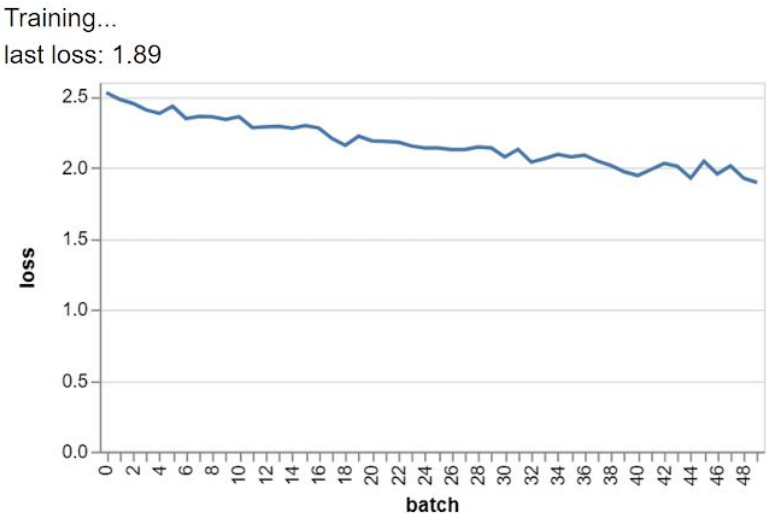
Task 5-1: Effect of Batch_Size and Num_Batches

Answer 5-1: Increasing both decreases the loss

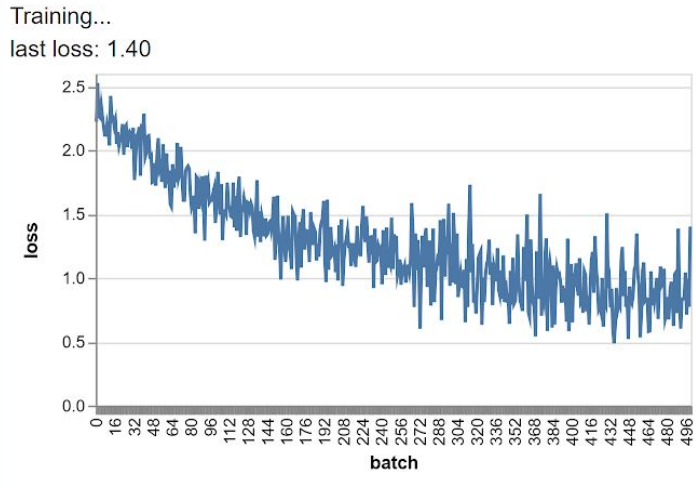
BATCH_SIZE = 10; NUM_BATCHES = 50;



BATCH_SIZE = 100; NUM_BATCHES = 50;



BATCH_SIZE = 10; NUM_BATCHES = 500



Task 5-2: Look at some of the testing results and try to find examples of classifications where the system does poorly and is even wrong. When you see interesting results, document them on your webpage.

Answer 5-2:

Class	Prob.
0	0.072
1	0.112
2	0.075
3	0.097
4	0.089
5	0.102
6	0.094
7	0.121
8	0.140
9	0.098

Class	Prob.
0	0.120
1	0.085
2	0.101
3	0.133
4	0.062
5	0.091
6	0.105
7	0.100
8	0.164
9	0.040

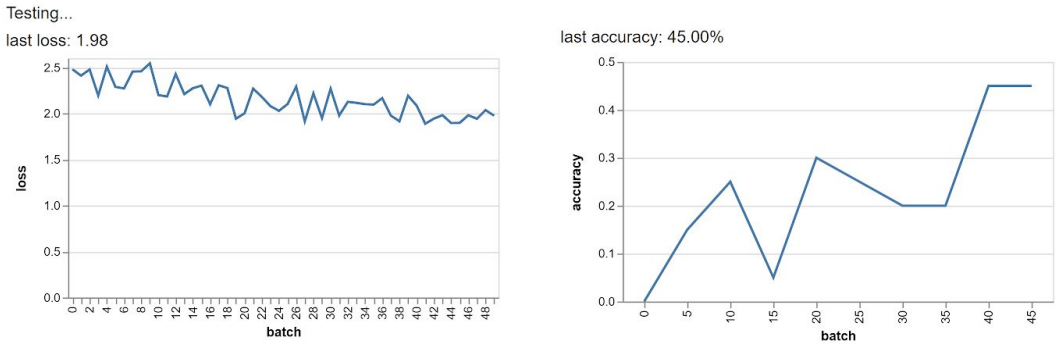
Class	Prob.
0	0.049
1	0.126
2	0.077
3	0.064
4	0.105
5	0.097
6	0.086
7	0.115
8	0.144
9	0.137

Class	Prob.
0	0.121
1	0.089
2	0.087
3	0.089
4	0.141
5	0.071
6	0.116
7	0.083
8	0.103
9	0.100

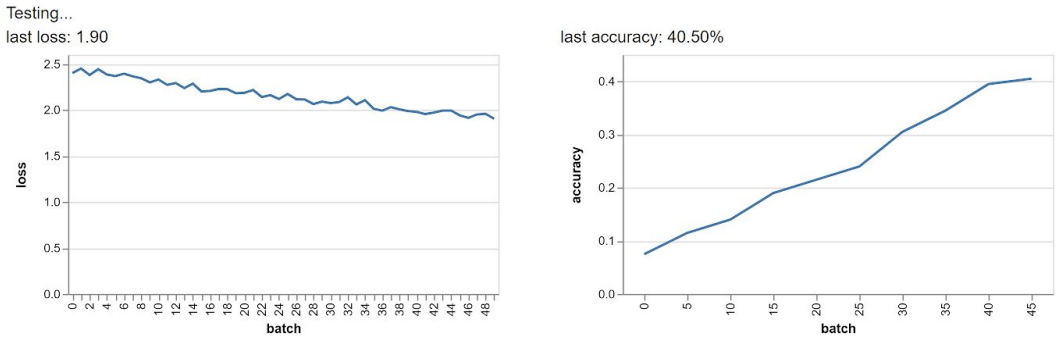
Task 5-3: Experiment with changing the batch size and the number of batches to try to improve the testing results. Give a brief description of what you tried, and the results.

Answer 5-3:

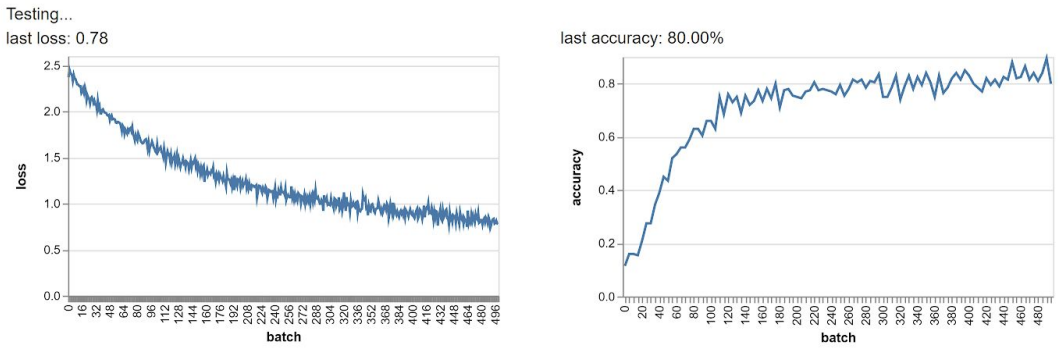
BATCH_SIZE = 20; NUM_BATCHES = 50



BATCH_SIZE = 200; NUM_BATCHES = 50

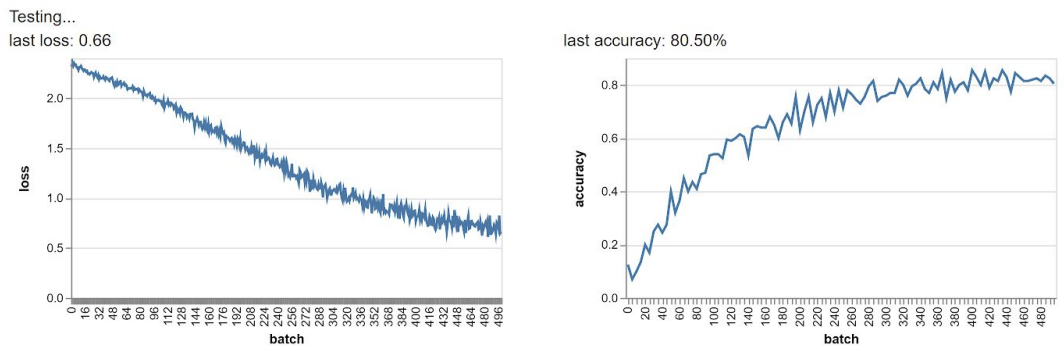


BATCH_SIZE = 200; NUM_BATCHES = 500

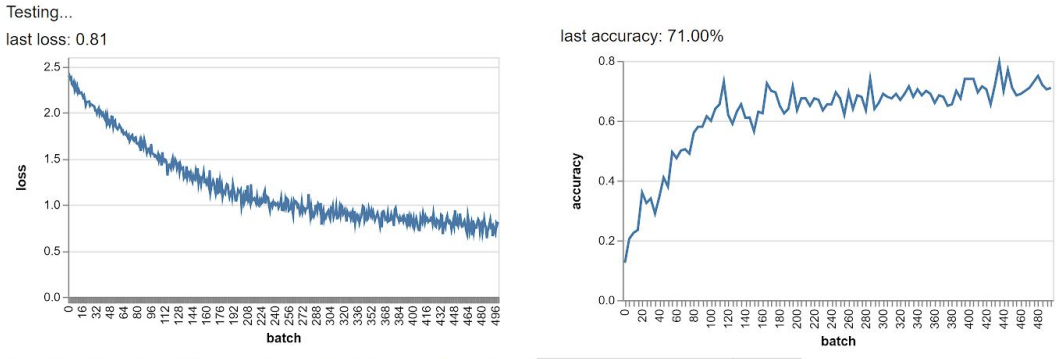


Task 5-4: Experiment to see how your new model does and briefly report on the results.Now that you've explored using MNIST, try some similar explorations with Fashion MNIST. Add links to your code files on your website (please make these are text files and not screenshots so that the graders can run your code if necessary).

Answer 5-4:



TensorFlow.js: Train Fashion_MNIST with the Layers API.



2.3: Style transfer examples

Create some interesting examples of style transfer. Place these on your Web page for sharing with the class on Wednesday. Make sure to allow time for DeepArt to generate the images and notify you.

Use This form to hand in Assignment 2
Due Monday September 17, at 10AM.