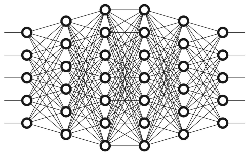


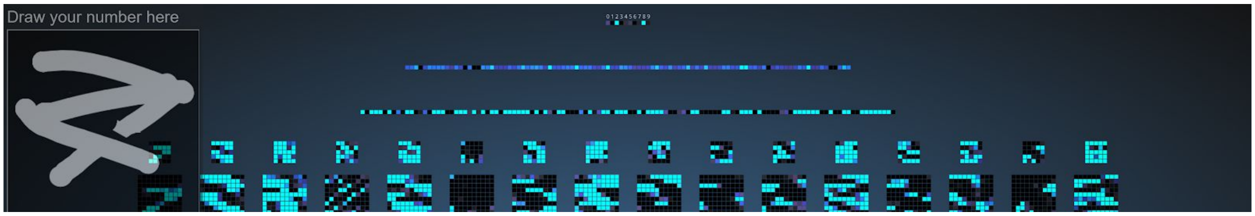
Massachusetts Institute of Technology
Dept. of Electrical Engineering and Computer Science
Fall Semester, 2018
MIT 6.S198: Deep Learning Practicum

Assignment 3: Convolutional Neural Networks

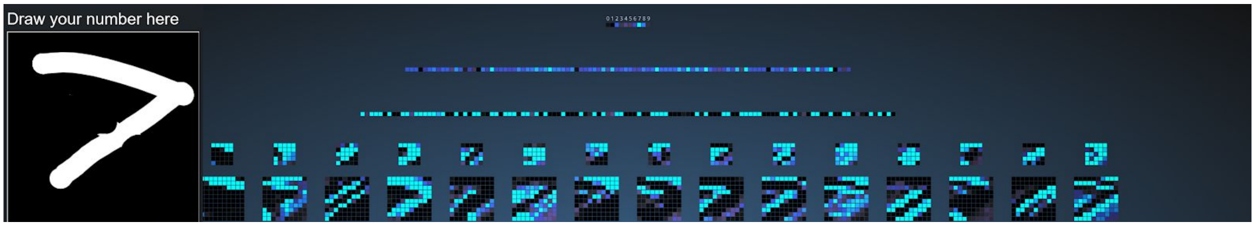
1 - Convolutional Net

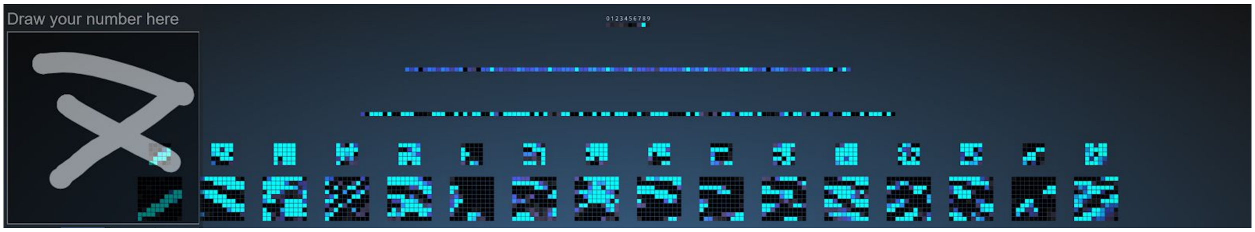
Task 1: Spend some time playing with this demo. Draw an input, modify it, and observe how the results at each layer change as you change the drawing. Create some inputs that look vaguely like digits, but that confuse the network, i.e., where two or more of the labels register. Write up interesting observations about what you see combined with illustrative screenshots.

Answer 1: When numbers are written atop one another the net gets confused.



Furthermore this net is not good at recognizing rotations and struggles to correctly identify digit when rotated if slightly adjusted.

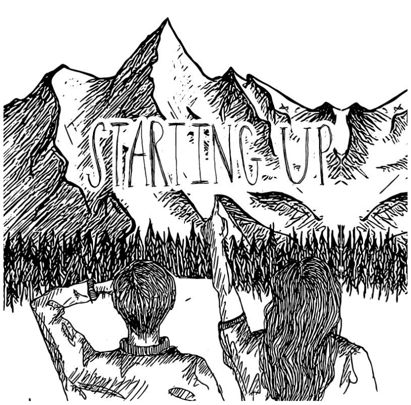


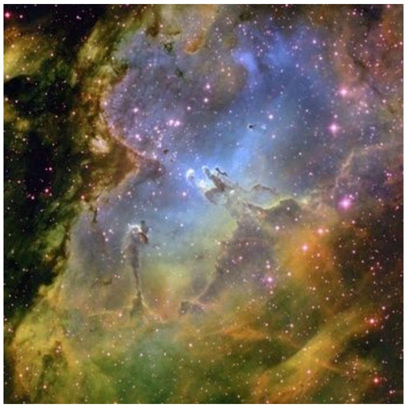


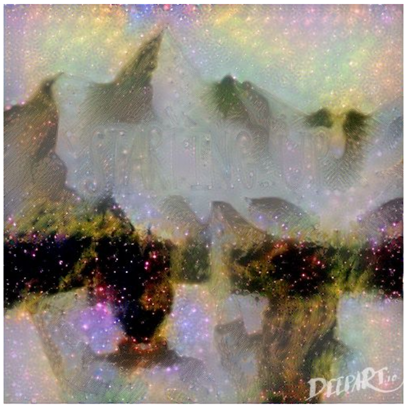
2 - Style Transformation

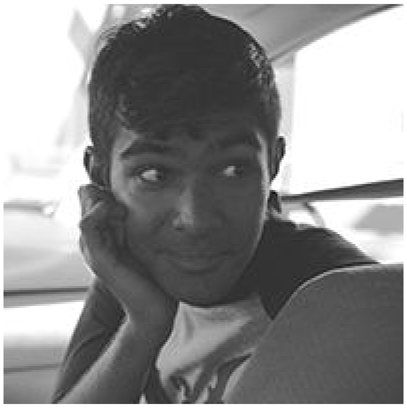
Task 2.1: Show images with style transfers

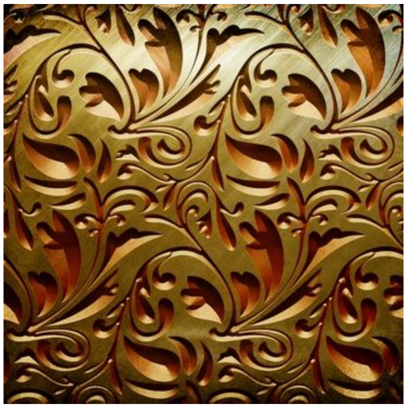
Answer 2.1:

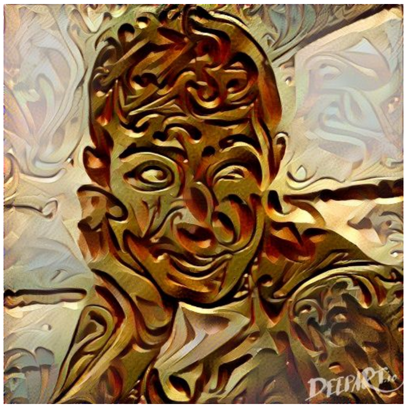








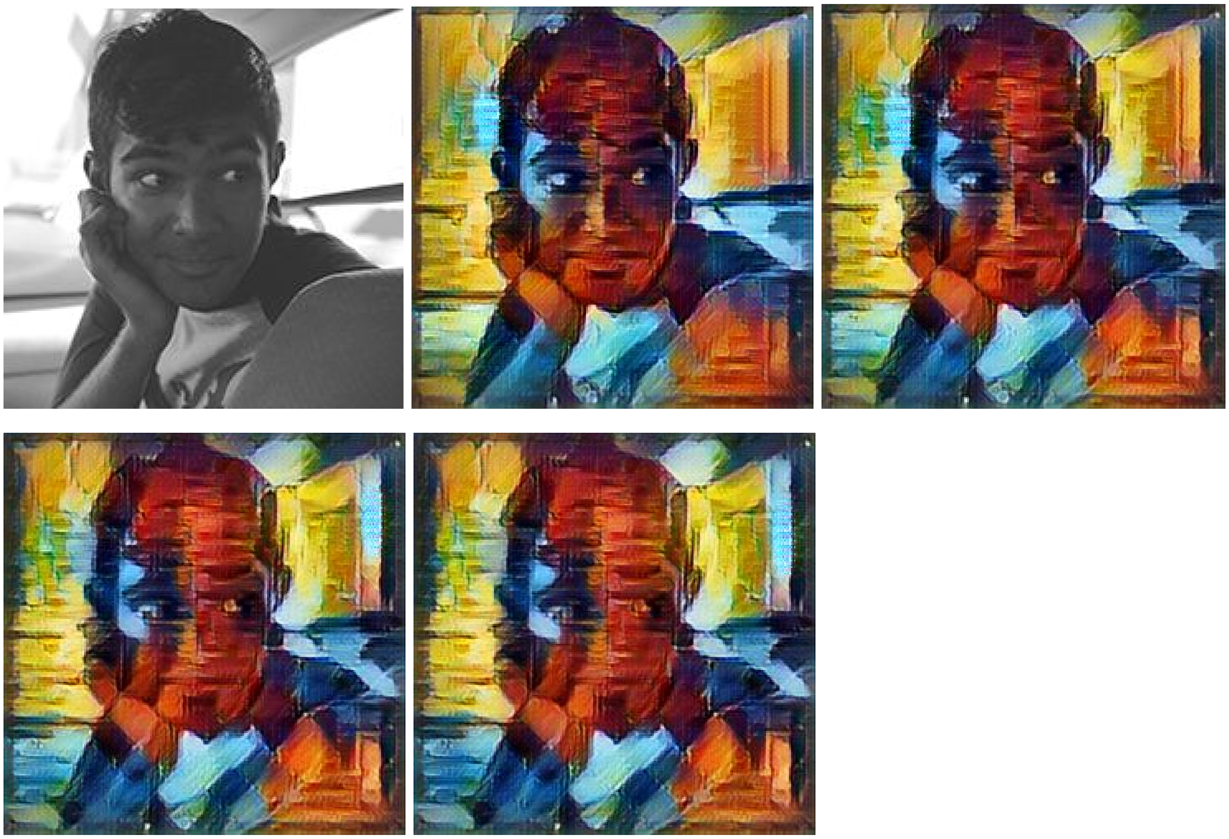




3 - Fast Style Transfer

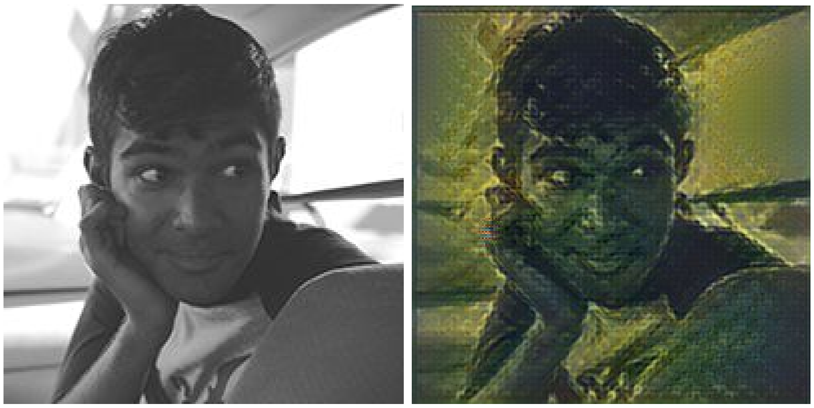
Task 3.1: What happens when you pass an image through the same filter several times? Do you observe any interesting characteristics or padding artifacts?

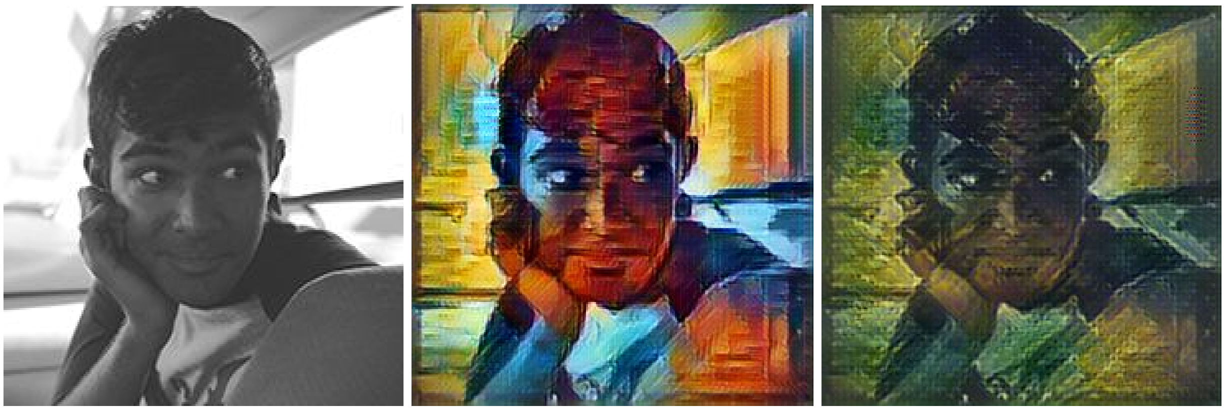
Answer 3.1: Passing the image through multiple times causes degradation of quality as more and more close pixels are pooled together. The edges also darken and the effect grows inward.



Task 3.2: Apply a filter to an image and then apply another filter to that already-filtered image. Is the result similar to what you would get when you apply the second filter to the original image?

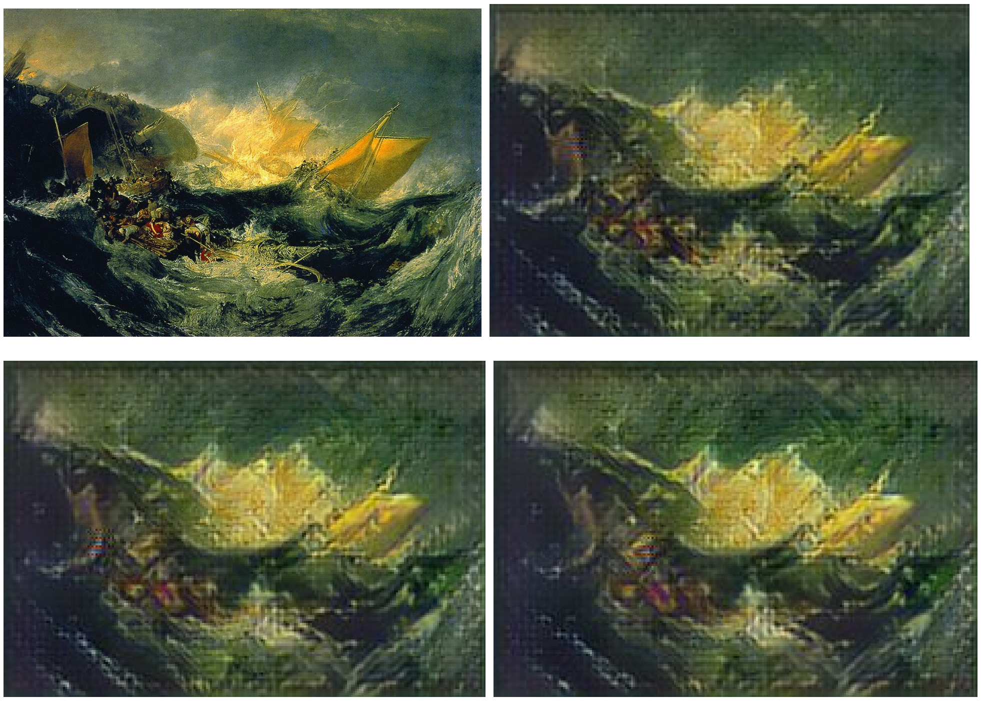
Answer 3.2: The results are similar but not the same. The middle layer leaves artifacts.





Task 3.3: Try different combinations of filters and number of times you filter an image. Note on your website if you have any interesting observations or insights.

Answer 3.3: Interestingly, the images changes even if you tell it to imitate itself.



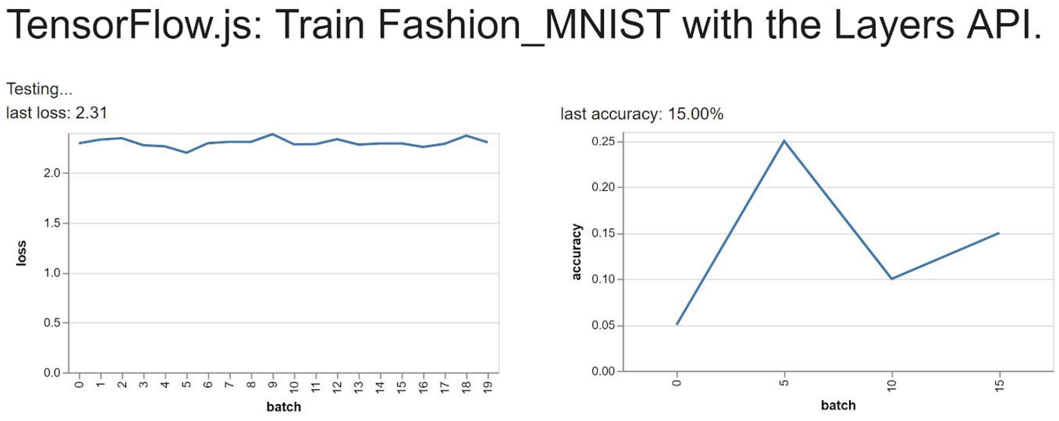
4 - CNNs with Code

For the following 4 questions, describe what architectures you implemented and take screenshots of the results you got for each dataset (MNIST, Fashion MNIST, and CIFAR).

Task 4.1: Try changing the learning rate, batch size, and number of batches. Make sure that these numbers are reasonable to start (i.e. won't take too long to run on your computer).

Answer 4.1:

Batch_Size = 20, Num_Batches = 20



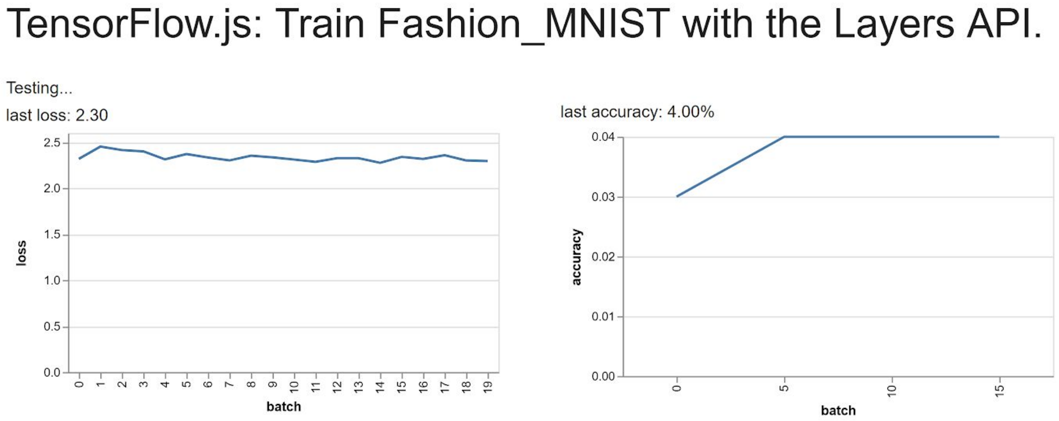
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 2.31

last accuracy: 15.00%

Batch_Size = 100, Num_Batches = 20



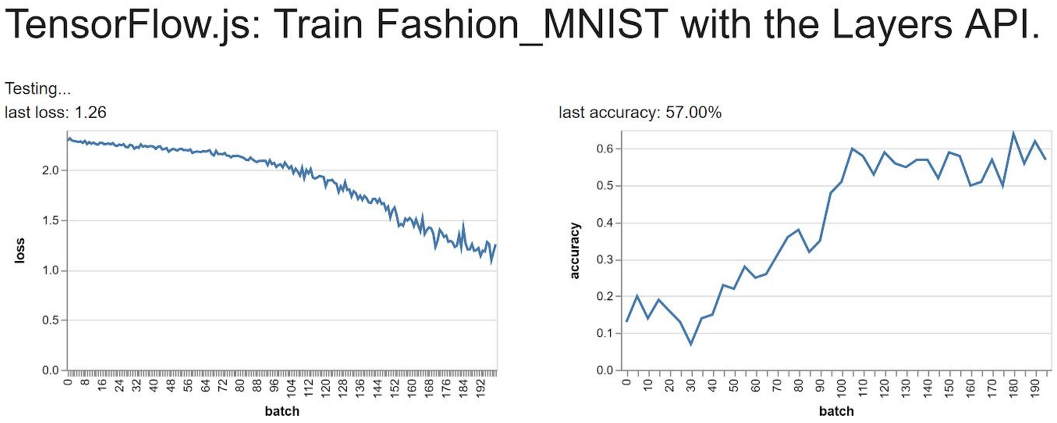
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 2.30

last accuracy: 4.00%

Batch_Size = 100, Num_Batches = 200



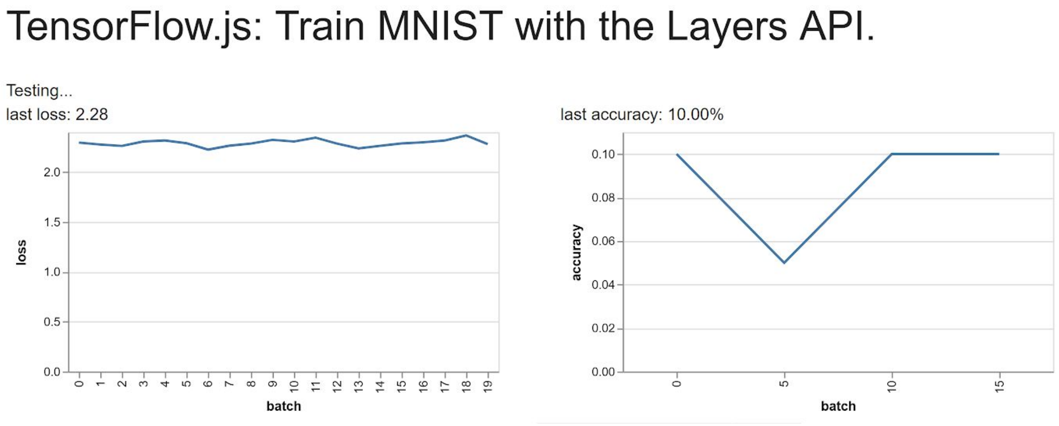
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 1.26

last accuracy: 57.00%

Batch_Size = 20, Num_Batches = 20



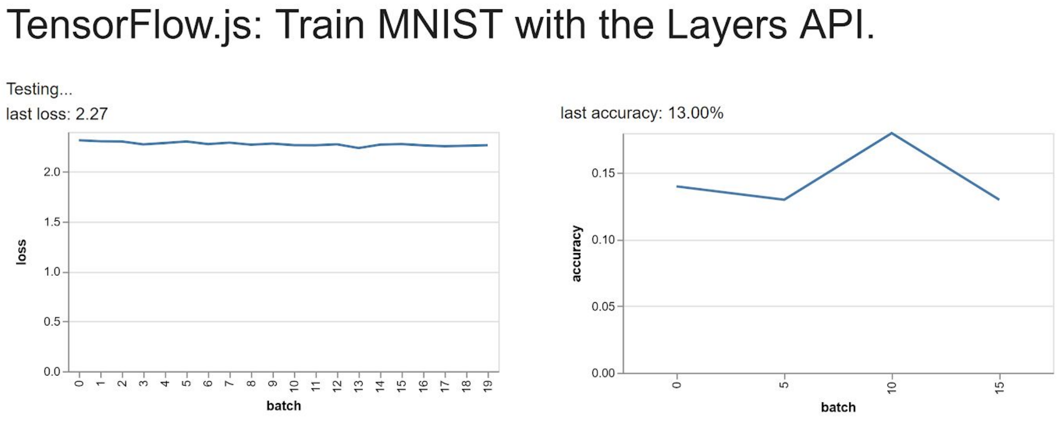
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 2.28

last accuracy: 10.00%

Batch_Size = 100, Num_Batches = 20



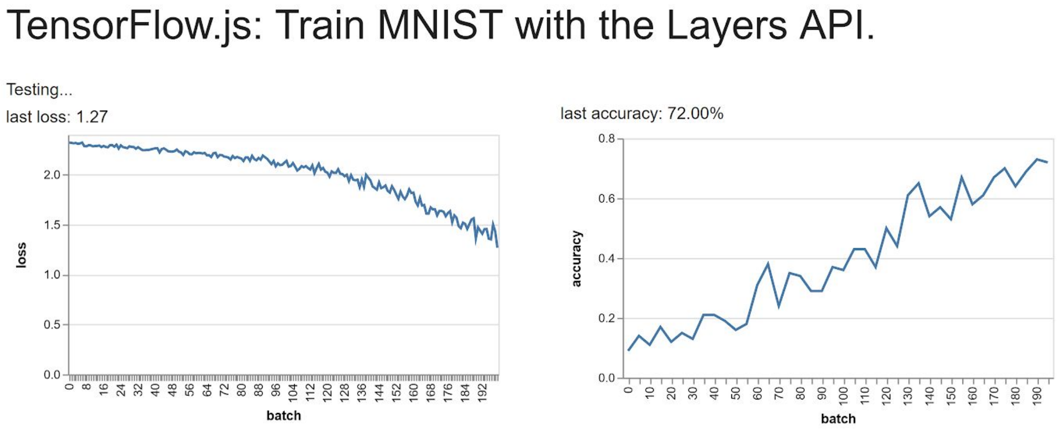
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 2.27

last accuracy: 13.00%

Batch_Size = 100, Num_Batches = 200



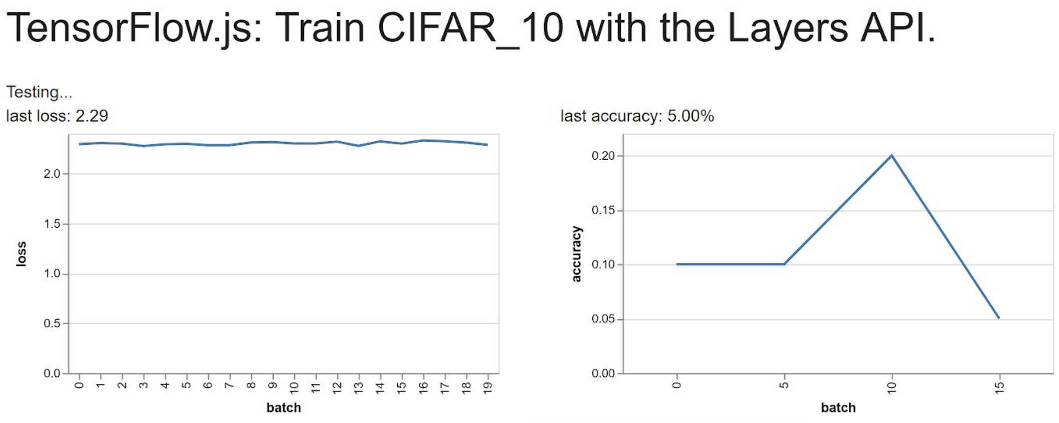
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 1.27

last accuracy: 72.00%

Batch_Size = 20, Num_Batches = 20



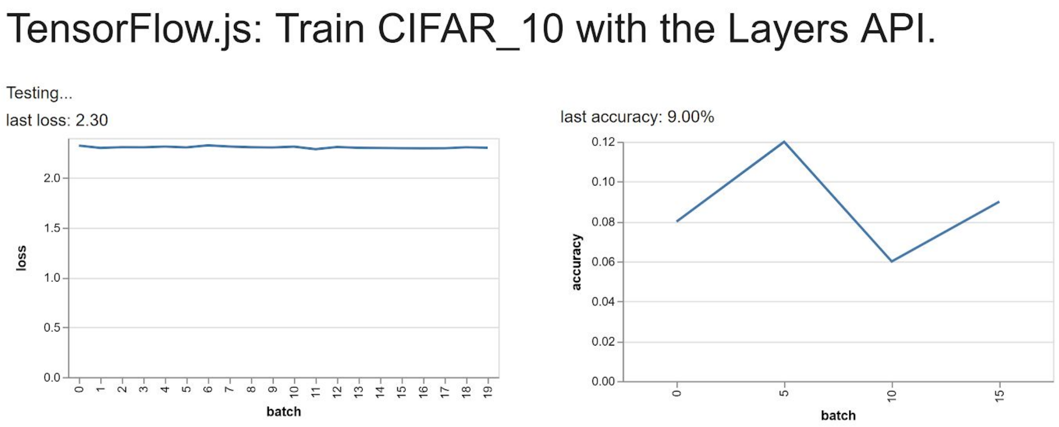
TensorFlow.js: Train CIFAR_10 with the Layers API.

Testing...

last loss: 2.29

last accuracy: 5.00%

Batch_Size = 100, Num_Batches = 20



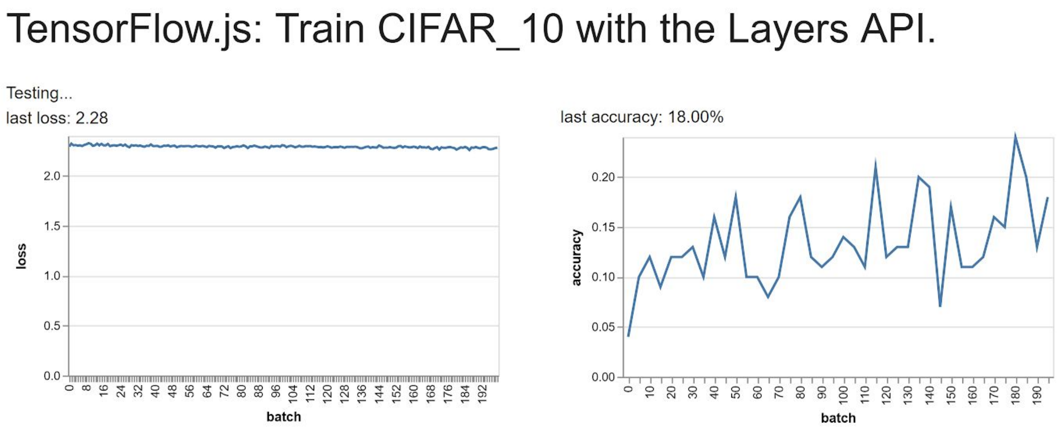
TensorFlow.js: Train CIFAR_10 with the Layers API.

Testing...

last loss: 2.30

last accuracy: 9.00%

Batch_Size = 100, Num_Batches = 200



TensorFlow.js: Train CIFAR_10 with the Layers API.

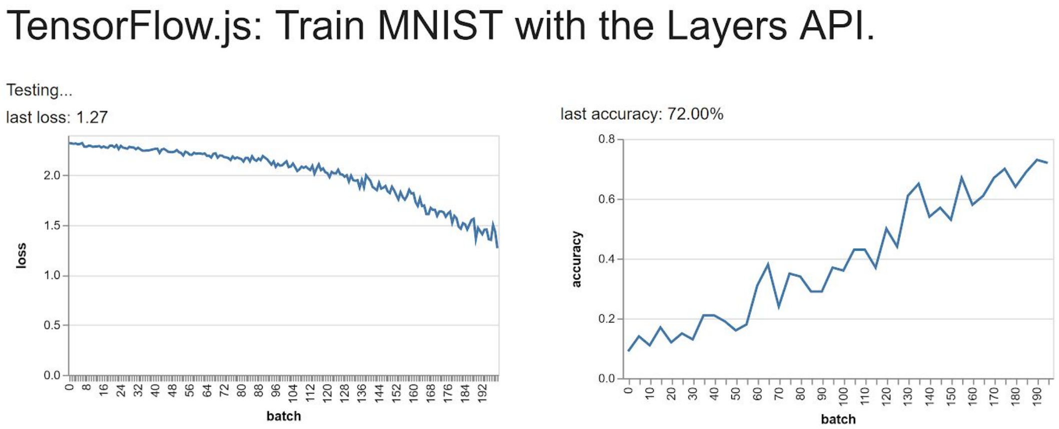
Testing...

last loss: 2.28

last accuracy: 18.00%

Task 4.2: Try changing some of the other parameters like field size, stride, output, …

Answer 4.2: poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu



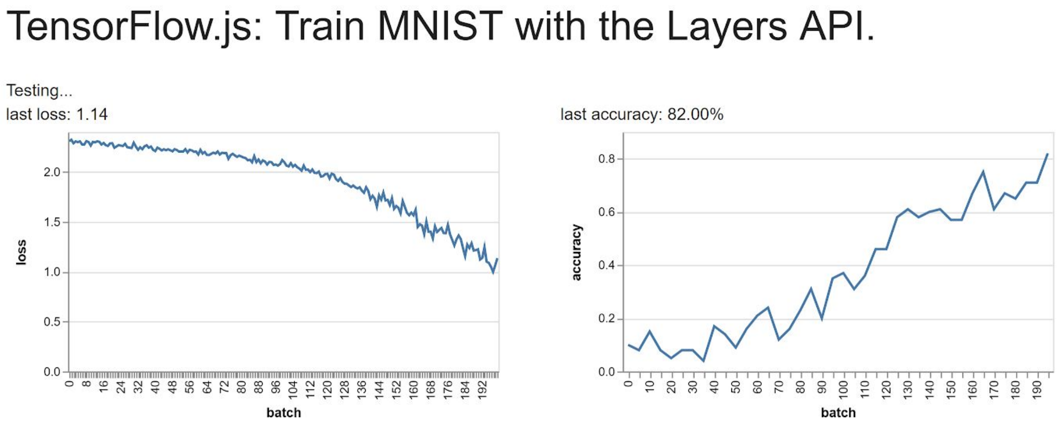
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 1.27

last accuracy: 72.00%

poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu6



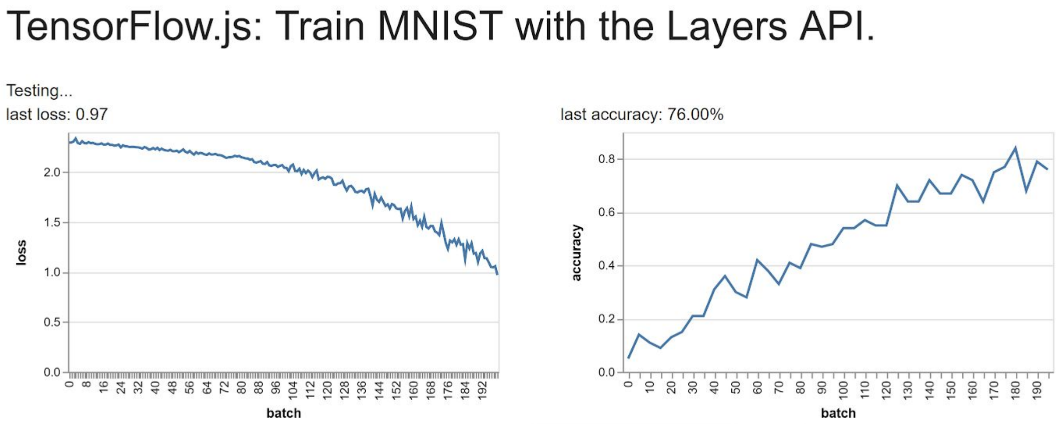
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 1.14

last accuracy: 82.00%

poolSize = [2,2], Stride = [2,2], filters: 32, activation: relu6



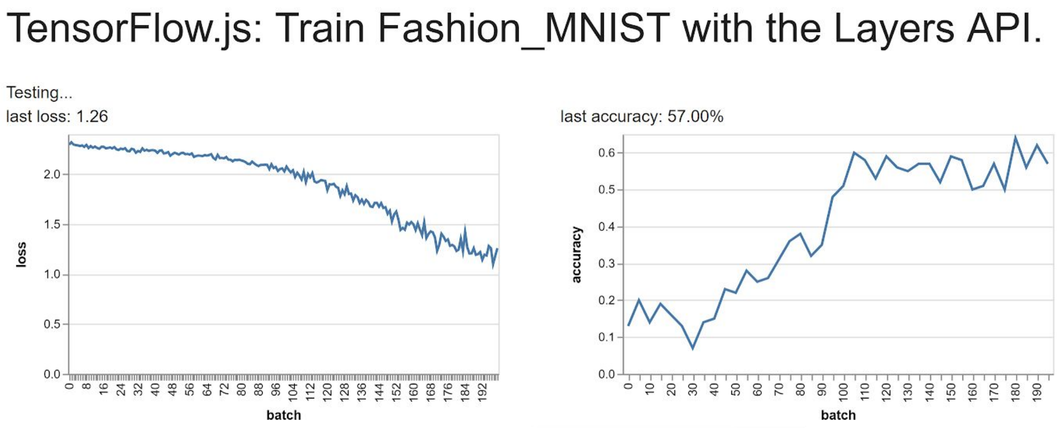
TensorFlow.js: Train MNIST with the Layers API.

Testing...

last loss: 0.97

last accuracy: 76.00%

poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu



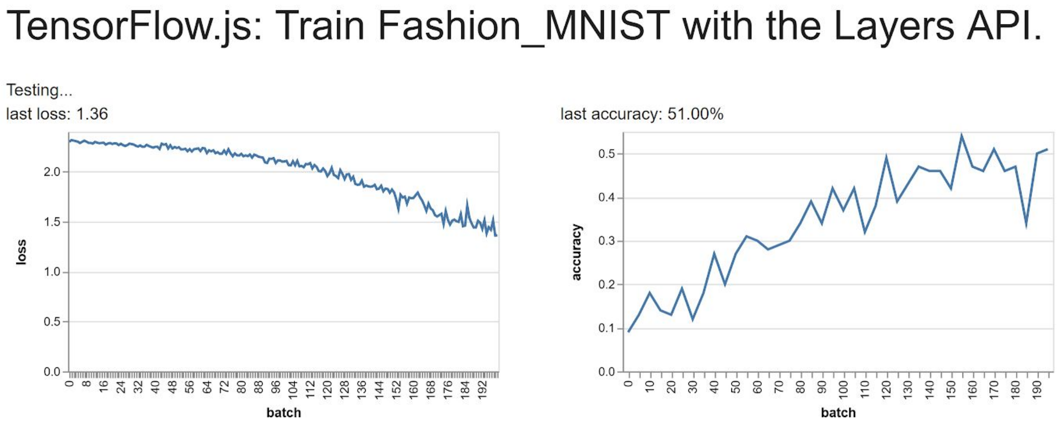
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 1.26

last accuracy: 57.00%

poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu6



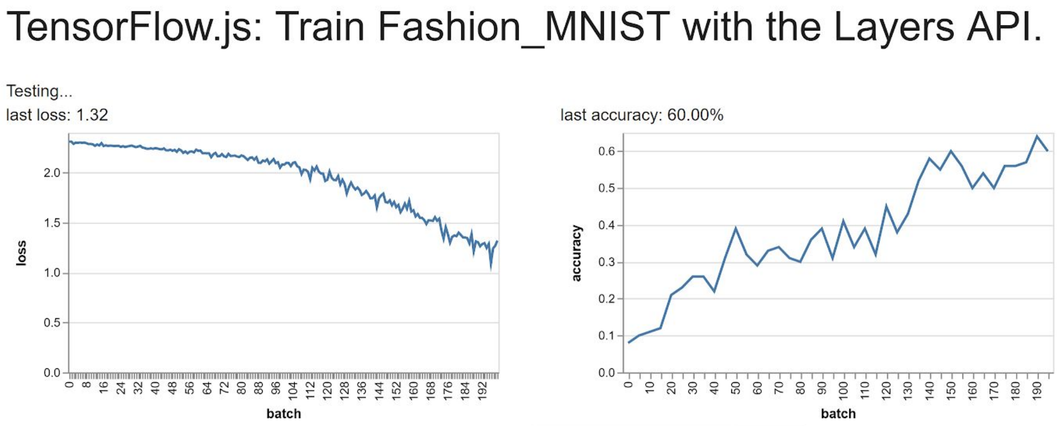
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 1.36

last accuracy: 51.00%

poolSize = [2,2], Stride = [2,2], filters: 32, activation: relu6



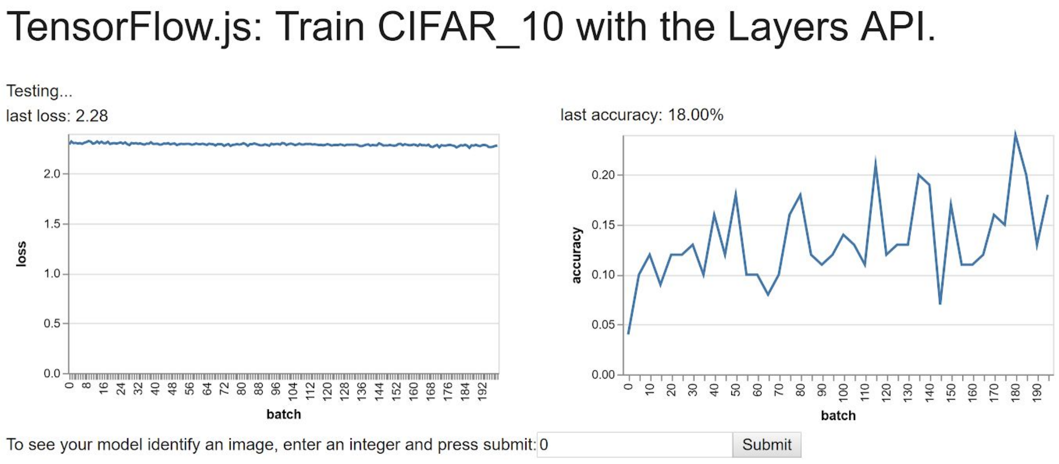
TensorFlow.js: Train Fashion_MNIST with the Layers API.

Testing...

last loss: 1.32

last accuracy: 60.00%

poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu



TensorFlow.js: Train CIFAR_10 with the Layers API.

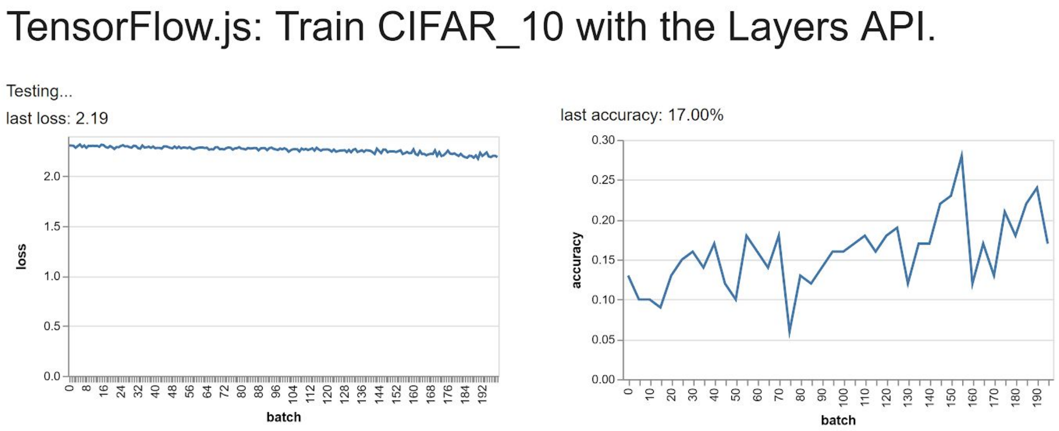
Testing...

last loss: 2.28

last accuracy: 18.00%

To see your model identify an image, enter an integer and press submit: 0

poolSize = [2,2], Stride = [2,2], filters: 16, activation: relu6



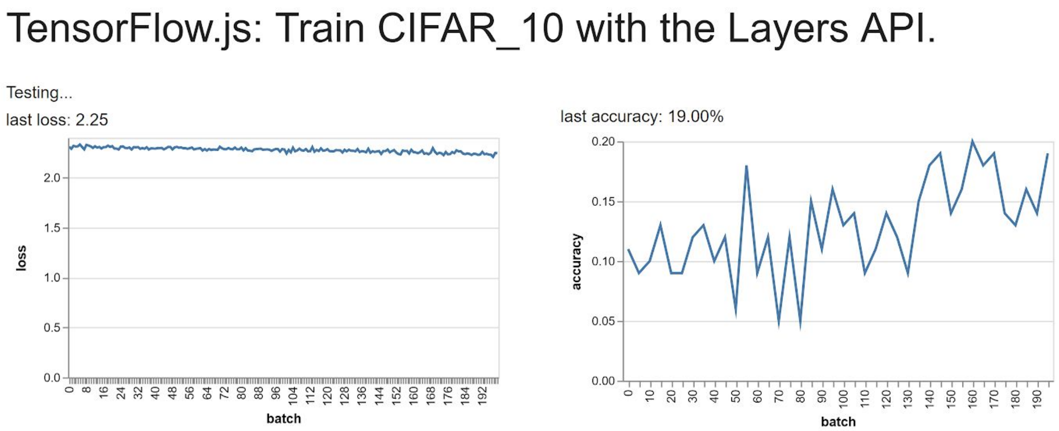
TensorFlow.js: Train CIFAR_10 with the Layers API.

Testing...

last loss: 2.19

last accuracy: 17.00%

poolSize = [2,2], Stride = [2,2], filters: 32, activation: relu6



TensorFlow.js: Train CIFAR_10 with the Layers API.

Testing...

last loss: 2.25

last accuracy: 19.00%

Task 4.3: Do you have a hypothesis for why CIFAR-10 is so much harder to train on than Fashion MNIST and MNIST (i.e. it's more difficult to achieve a 90%+ accuracy) while Fashion MNIST has similar training times to MNIST (even though Fashion MNIST is more complex than MNIST)?

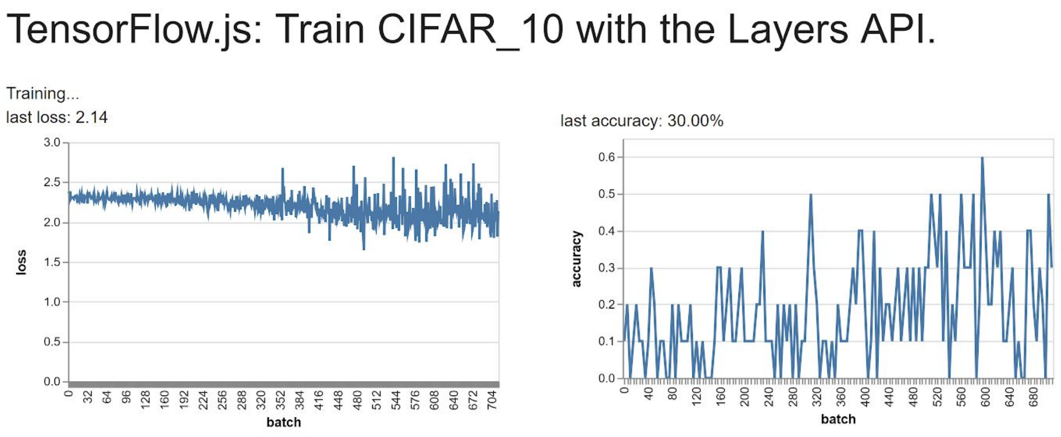
Answer 4.3: CIFAR-10 is RGB so it has three channels and makes a much larger information set than MNIST or Fashion MNIST, making it much harder to train.

Task 4.4: How does adding more convolutional layers relate to accuracy and training speed? Is there a point at which adding more layers plateaus or even decreases the maximum accuracy you are able to achieve with that model?

Answer 4.4: Adding more convolutional layers increases accuracy (but decreases training speed) up to around 3 layers. Afterward the layers plateaus. The max pooling layers are lossy so if you had enough layers of ConvNets you could actually decrease accuracy.

Task 4.5: Challenge: Are you able to find an architecture/combination of techniques that can get you to 60% accuracy on CIFAR-10 within 1 minute of training? 5 minutes? 10 minutes?

Answer 4.5: One quick way to get to 60% accuracy quickly (though unstably) is to do small batches <10 for many runs >1000.



TensorFlow.js: Train CIFAR_10 with the Layers API.

Training...
last loss: 2.14

last accuracy: 30.00%

Task 2.6: Add code that captures some performance statistic from a testing run. For example, you might keep track of the fraction of examples where the prediction was correct , or perhaps do something more detailed that takes the predicted probabilities into account. Hint: Look near the end of the file where the testing results are logged. Using your performance metrics, compare some of the different models you built above. Write a brief report with screenshots showing the choices and the results.

Task 2.7: Add links to your code files on your website (please make sure these are text files and not screenshots so that the graders can run your code if necessary).

Action 2.7: Uploaded final code file with comments about changes.

3: Submission

Create a page for assignment 3 on your class homework submission site. Include your name and email address and the required writeups as indicated above together with code and images, as appropriate.

Use This form to hand in Assignment 3
Due 10AM Monday, 9/24