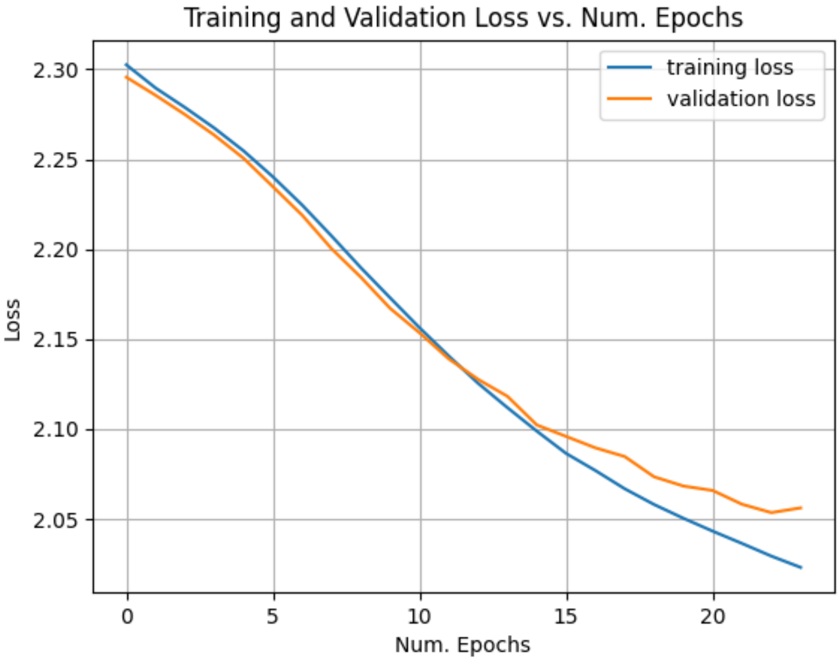
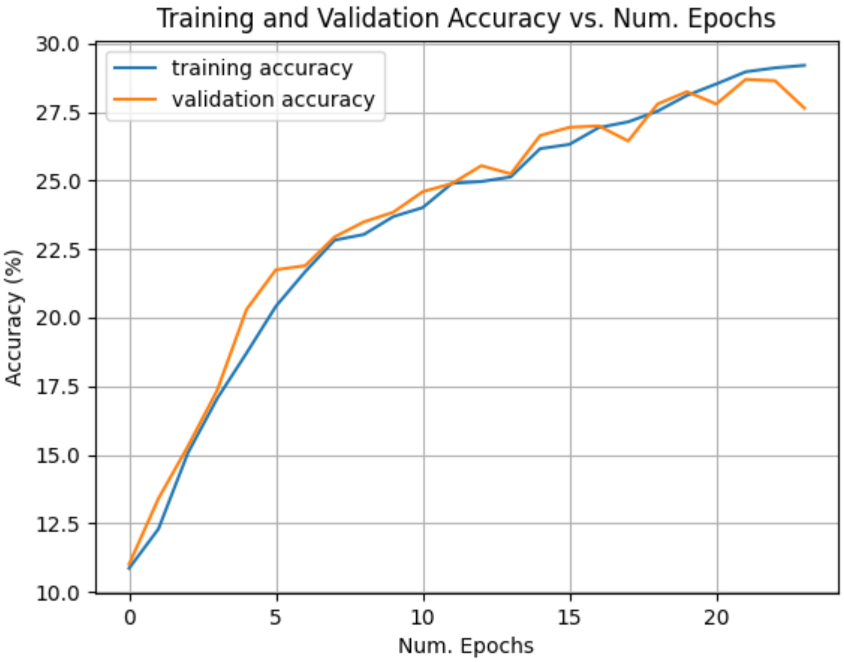
Christopher Tsai

CS 349 Final Project Free Response Questions

Convolutional Neural Networks:

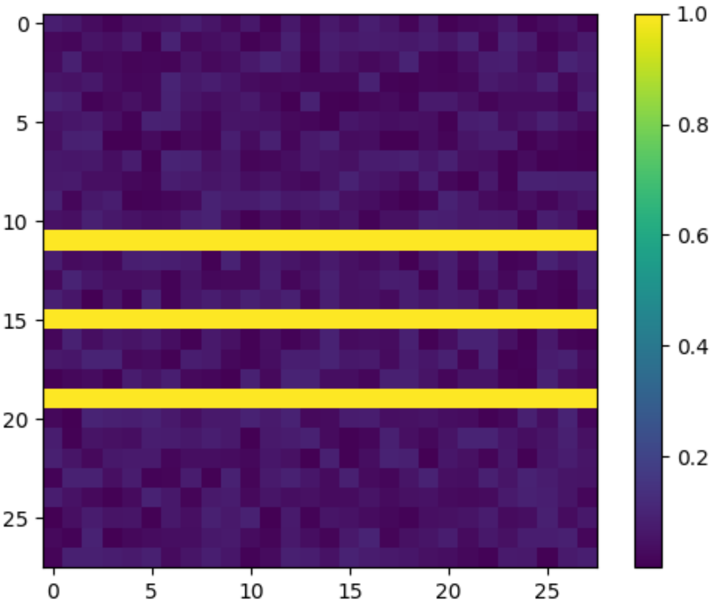
10. The output size is [16, 3, 17, 23]. I've acquired these answers two ways. The first way is through PyTorch code, which you can see in my *fr_answers.py* file's *problem_10()* function.
    The second way is through the formula $O = \frac{W-K+2P}{S} + 1$, where O is the output height/length, W is the input height/length, K is the filter size, P is the padding, and S is the stride (source: https://www.quora.com/How-can-I-calculate-the-size-of-output-of-convolutional-layer). For the first image dimension, $O = \frac{32-8+2*4}{2} + 1 = 17$. For the second, $O = \frac{32-4+2*8}{2} + 1 = 23$. The 3 in [16, 3, 17, 23] corresponds to the number of output channels. The 16 corresponds to the batch size.
11. Kernel sizes: [(5, 5), (5, 5)]. Strides: [(1, 1), (1, 1)]. Total num. of weights: 68138 (see *fr_answers.py* for code for acquiring total weights).
12. Number of epochs before terminating: 24. Accuracy on testing set: 28.11%.



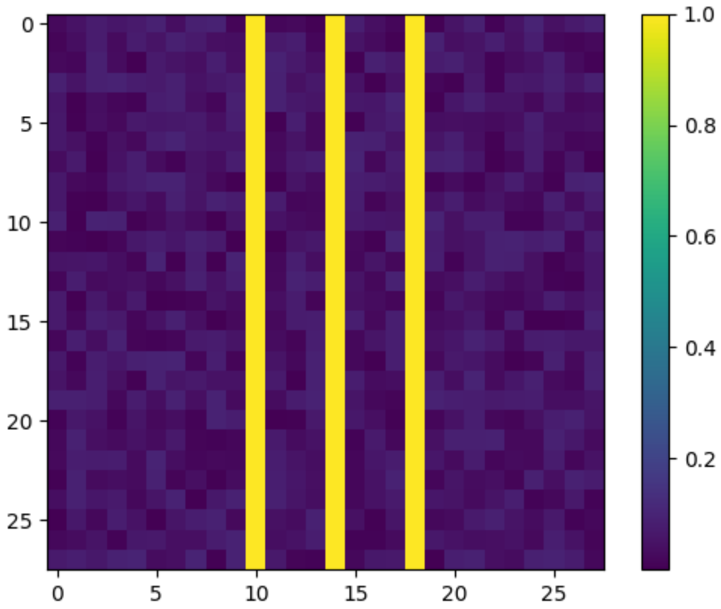


13. The first 5000 images in the data set belong to the first class, which contains images of three horizontal stripes. The last 5000 images belong to the second class, which contains images of three vertical stripes.

    Example of first class:



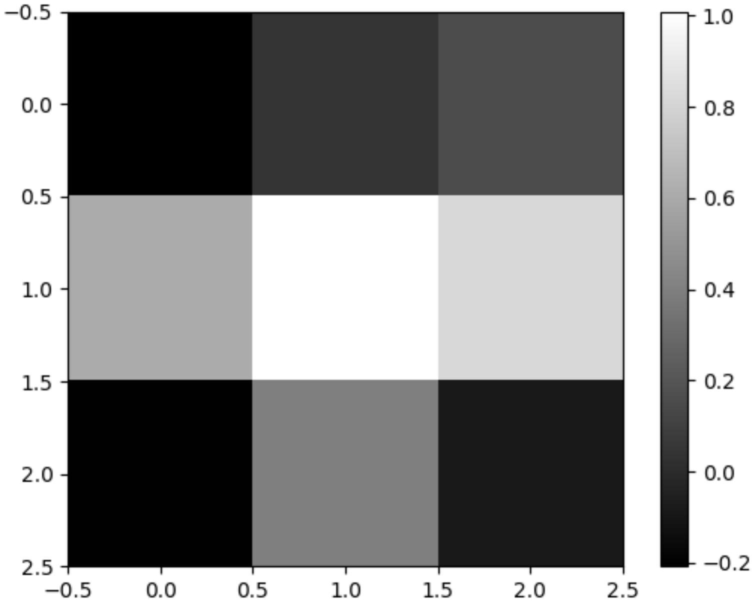
Example of second class:



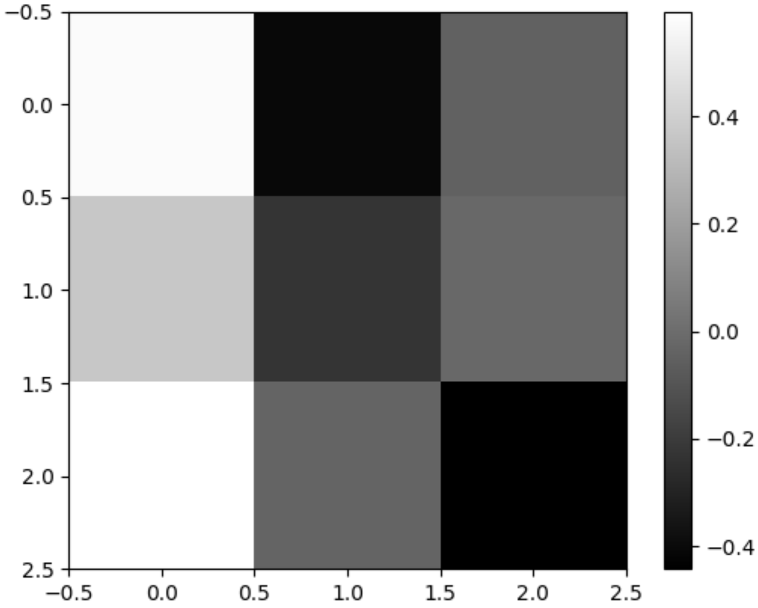
Within each class, what differences each sample to the next is the location of the bars. For the first class, the three bars can go along any y-coordinate (within the range of the image, [0,30]), and for the second class, the three bars can go along any x-coordinate (again, within the range of the image). The distance between each bar remains constant throughout.

To classify these images, a model must be shift invariant. This means that it should be able to detect the horizontal/vertical-ness of the bars regardless of their location. The classifier must rely on the information of the window that contains the bars.

14. I had to increase the learning rate to 1e-3 to see some patterns. In general, the first kernel usually contains a horizontal "line" of high value elements, like so:



The second kernel usually has a vertical “line”:



Clearly, the horizontal “line” in the first kernel corresponds to the horizontal lines in the first image class and the vertical “line” in the second kernel corresponds to the vertical lines in the second image class. However, when ran multiple times these results were not always consistent (because of the randomness of the initial guess). In fact, a good number of times they are flipped (so vertical for kernel 1 and horizontal for kernel 2). But there is always some “line” of high-valued elements in the kernels.

Text Classification: