CMSC510 - Assignment 4

Steven M. Hernandez, V00528395

November 2020

In this assignment, I created two CNN model scripts using PyTorch and Tensorflow respectively. Both models were generated to be similar with $c \in \{0, 1, 2\}$ as the number of convolutional layers and $m \in \{3, 5, 9\}$ as the filter size per convolutional layer.

1 Results

In Table 1 we can see the time taken and final validation accuracy for each model parameters. We repeat the experiment parameters for both pytorch as well as tensorflow. We can see that adding a convolutional layer significantly increases the the to compute the final results when a GPU is not used. We can also see that increasing the filter size m increases the Validation Accuracy as well. The figures on the following pages show also the validation accuracy and validation loss at the end of each epoch.

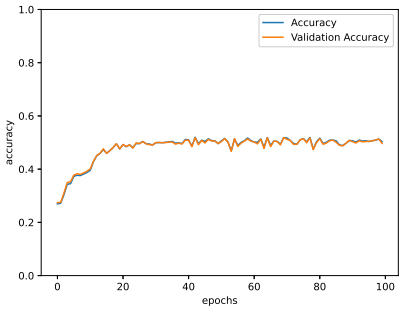
I will note, Tensorflow took a considerably higher time to train. For Pytorch, the times written are for 100 epochs while Tensorflow was trained for only 10 epochs. Even so, Tensorflow achieves higher accuracy overall. One different I tried in the approaches was to use Adam optimizer for Pytorch versus Stochastic Gradient Descent (SGD) for Tensorflow. Because of the high difference in execution times for each optimizer, I tried $c = 1$, $m = 3$ and $num\_epochs = 10$ for each combination of ML-library and optimizer which can be seen in Table 2. We can see that the optimizer does not appear to cause any changes, but Tensorflow is still significantly more time consuming than Pytorch. However, I will also note that I performed these experiments on a different day (after shutting down the server). Thus, we can see Table 2 shows lower total times than in Table 1. So maybe there were some other processes running which slowed down the computation significantly.

Library	c	m	Time (minutes)	Val. Accuracy	# Epochs	Optimizer
Pytorch	0	N/A	1.35	49.6%	100	Adam
Pytorch	1	3	39.6	66.2%	100	Adam
Pytorch	1	5	37.0	76.3%	100	Adam
Pytorch	1	9	27.9	94.6%	100	Adam
Pytorch	2	3	93.6	47.3%	100	Adam
Pytorch	2	5	80.6	86.4%	100	Adam
Pytorch	2	9	49.4	90.7%	100	Adam
Tensorflow	0	N/A	0.45	54.2%	10	SGD
Tensorflow	1	3	56.7	96.8%	10	SGD
Tensorflow	1	5	53.2	97.01%	10	SGD
Tensorflow	1	9	44.0	97.00%	10	SGD
Tensorflow	2	3	153.64	96.7%	10	SGD
Tensorflow	2	5	198.50	97.5%	10	SGD
Tensorflow	2	9	199.37	97.1%	10	SGD

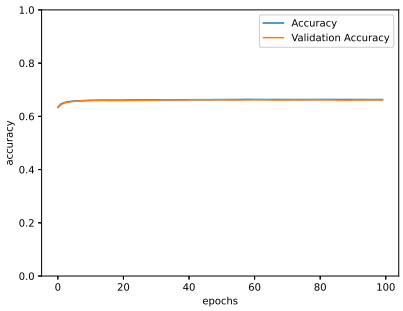
Table 1: Results of each experiment

	Adam	SGD
Pytorch	1.69	1.59
Tensorflow	11.57	11.93

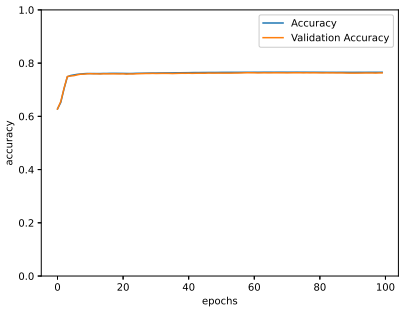
Table 2: Comparing Time (minutes) to Train 10 epochs



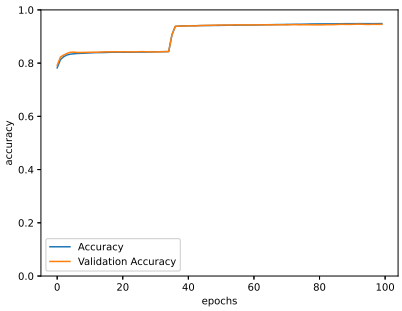
(a) $c = 0$, $m = 0$



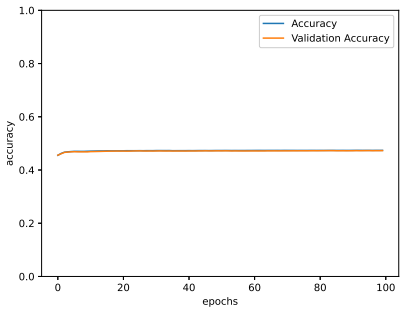
(b) $c = 1$, $m = 3$



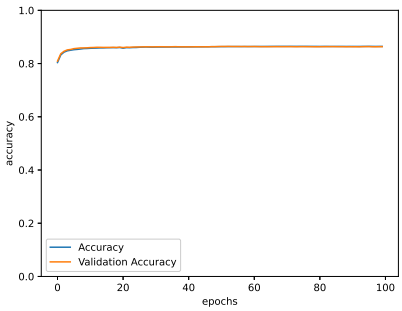
(c) $c = 1$, $m = 5$



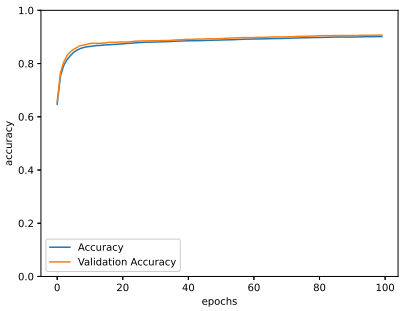
(d) $c = 1$, $m = 9$



(e) $c = 2$, $m = 3$

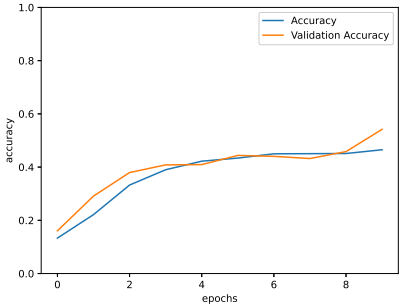


(f) $c = 2$, $m = 5$

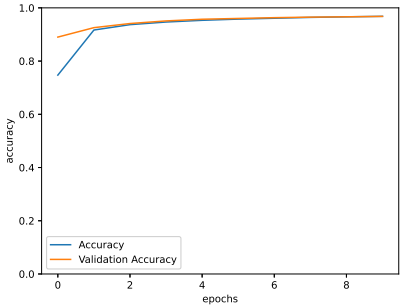


(g) $c = 2$, $m = 9$

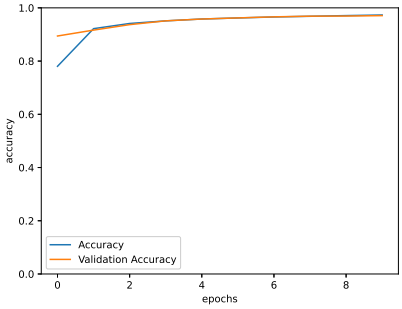
Figure 1: Accuracy for PyTorch



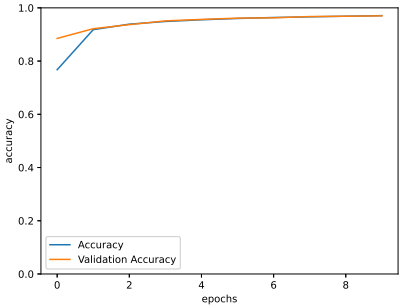
(a) $c = 0$, $m = 0$



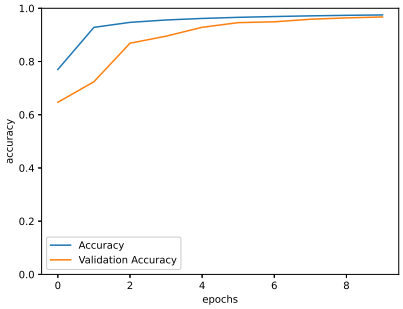
(b) $c = 1$, $m = 3$



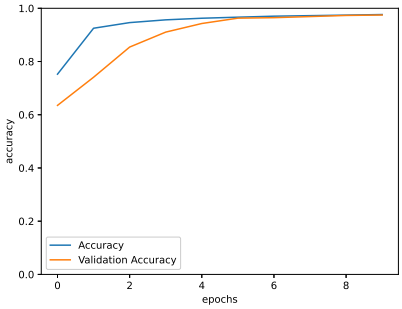
(c) $c = 1$, $m = 5$



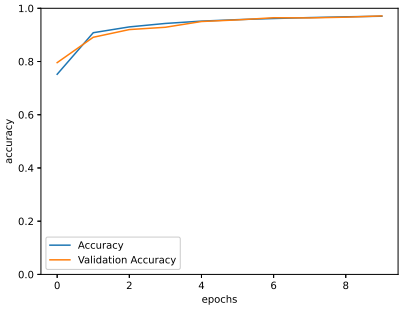
(d) $c = 1$, $m = 9$



(e) $c = 2$, $m = 3$

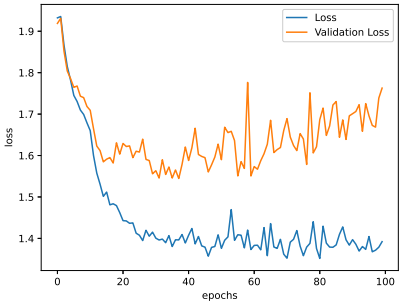


(f) $c = 2$, $m = 5$

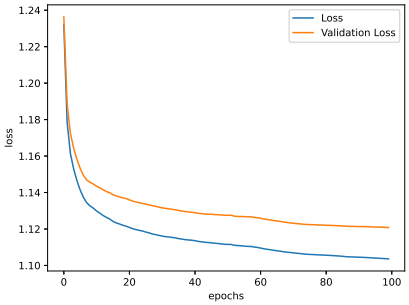


(g) $c = 2$, $m = 9$

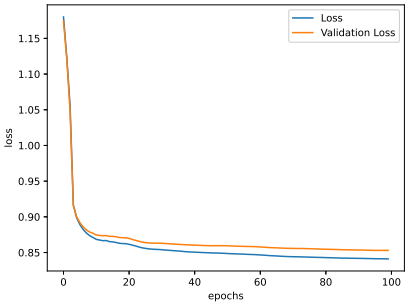
Figure 2: Accuracy for Tensorflow



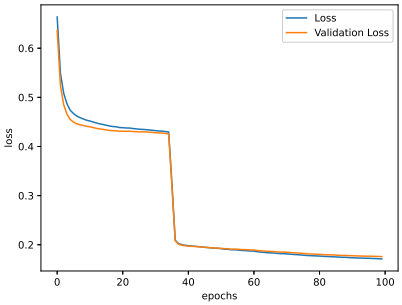
(a) $c = 0$, $m = 0$



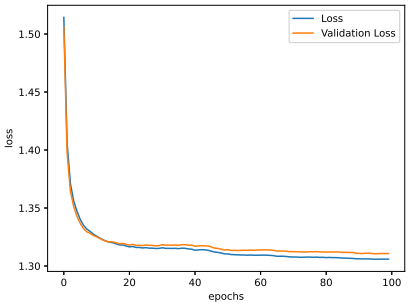
(b) $c = 1$, $m = 3$



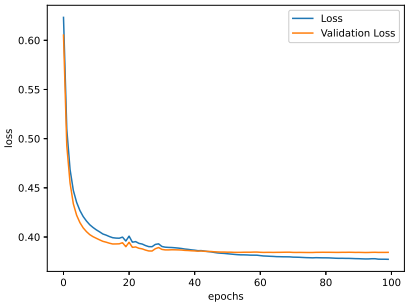
(c) $c = 1$, $m = 5$



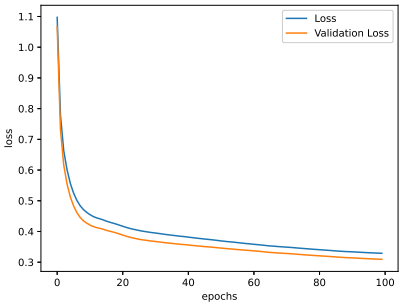
(d) $c = 1$, $m = 9$



(e) $c = 2$, $m = 3$

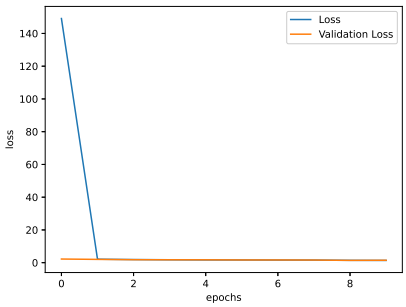


(f) $c = 2$, $m = 5$

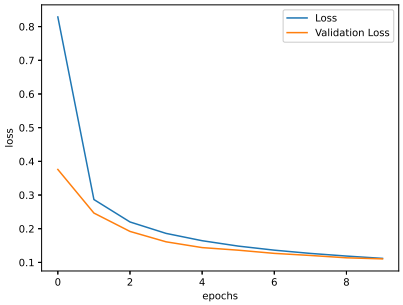


(g) $c = 2$, $m = 9$

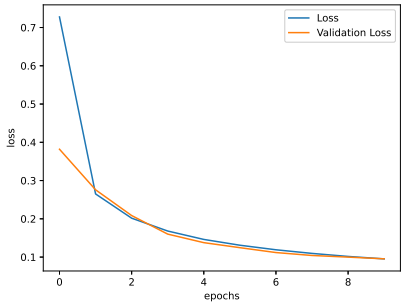
Figure 3: Loss for PyTorch



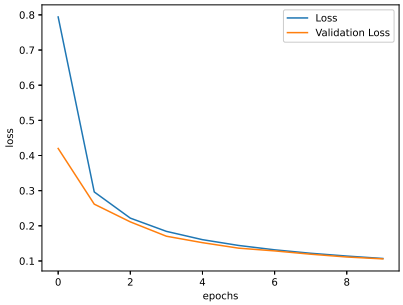
(a) $c = 0$, $m = 0$



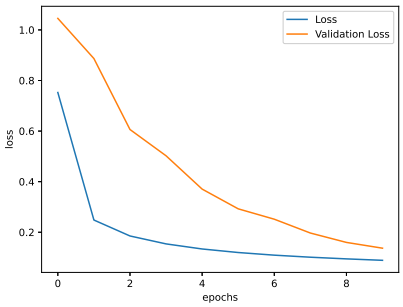
(b) $c = 1$, $m = 3$



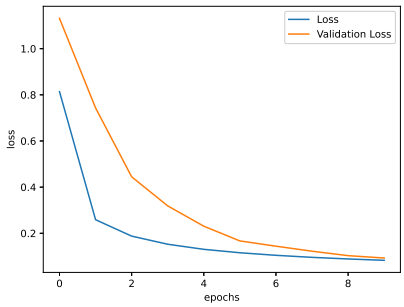
(c) $c = 1$, $m = 5$



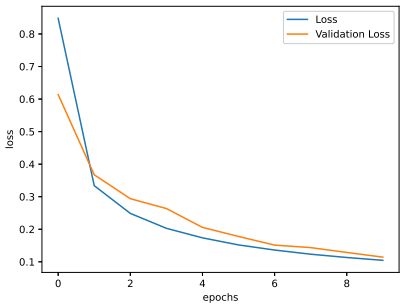
(d) $c = 1$, $m = 9$



(e) $c = 2$, $m = 3$



(f) $c = 2$, $m = 5$



(g) $c = 2$, $m = 9$

Figure 4: Loss for Tensorflow