6135B-Assignment 2-Report

Ethan Kreuzer

March 2025

Question 1.3

To determine the number of learnable parameters in the model, we break the computation down layer by layer.

First Layer (without biases):
The input to the first LSTM layer includes both the input vector $x \in \mathbb{R}^{d_x}$ and the hidden state $h \in \mathbb{R}^{d_h}$. Each of the four gates (input, forget, output, and cell) has weight matrices for both x and h:

$$\mathbf{W}_{[\cdot]}, \mathbf{U}_{[\cdot]} \in \mathbb{R}^{d_h \times (d_x + d_h)}$$

Subsequent Layers (without biases):
All following layers receive inputs only from the previous hidden state (of dimension d_h), assuming no biases. This gives

$$\mathbf{W}_{[\cdot]}, \mathbf{U}_{[\cdot]} \in \mathbb{R}^{d_h \times (2 * d_h)}$$

for the input, output, forget and cell gates for the for the next (L-1) layers
This yields

$$4[d_h \cdot d_x + d_h^2 + 2d_h^2(L-1)]$$

total parameters if no biases

Adding Biases:
Adding biases we now have the additional vectors

$$b_{[i]}, b_{[f]}, b_{[o]}, b_{[c]} \in \mathbb{R}^{d_h}$$

which adds $4Ld_h$ total parameters.

$$\text{Number of Parameters} = \begin{cases} 4d_h d_x + d_h^2 + (L-1) \cdot 2d_h^2 + 4Ld_h & \text{if bias = True} \\ 4d_h d_x + d_h^2 + (L-1) \cdot 2d_h^2 & \text{if bias = False} \end{cases}$$

Question 2.6

The model has to learn the parameters

$$\mathbf{W}_{[Q]}, \mathbf{W}_{[K]}, \mathbf{W}_{[V]}, \mathbf{W}_{[O]} \in \mathbb{R}^{\text{head size} \times \text{number of heads}}$$

this is $4 \times \text{head size}^2 \times \text{number of heads}^2$ parameters.
Now if we add Biases, we need to learn the parameters

$$b_{[Q]}, b_{[K]}, b_{[V]}, b_{[O]} \in \mathbb{R}^{\text{head size} \times \text{number of heads}}$$

$$\text{Number of Parameters} = \begin{cases} 4 \times \text{head size}^2 \times \text{number of heads}^2 + 4 \times \text{head size} \times \text{number of heads} & \text{if bias = True} \\ 4 \times \text{head size}^2 \times \text{number of heads}^2 & \text{if bias = False} \end{cases}$$

Question 2.10

Above in 2.6 we have the result:

$$P = \begin{cases} 4 \times \text{head size}^2 \times \text{number of heads}^2 + 4 \times \text{head size} \times \text{number of heads} & \text{if bias = True} \\ 4 \times \text{head size}^2 \times \text{number of heads}^2 & \text{if bias = False} \end{cases}$$

The MLP at the end of each block does not have a bias. Since the input and ouptut dimensions of the layer are both d_{model} with hidden dimenion d_{ff}. This gives us $2 \times \text{model} \times d_{model}^2$ paramaters to be learned. Therefore,

$$P = \begin{cases} \text{L } (4 \times \text{head size}^2 \times \text{number of heads}^2 + 4 \times \text{head size} \times \text{number of heads} + 2 \times \text{model} \times d_{model}^2) & \text{if bias = True} \\ 4 \times \text{head size}^2 \times \text{number of heads}^2 & \text{if bias = False} \end{cases}$$

Question 4.1

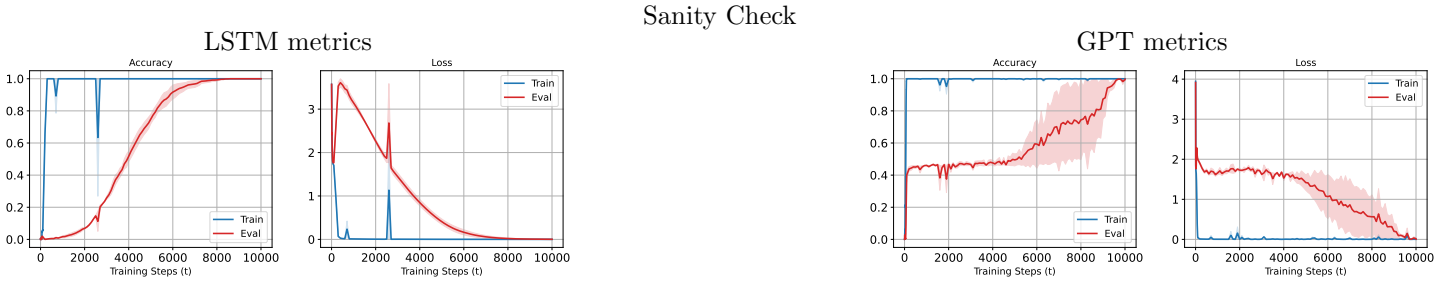


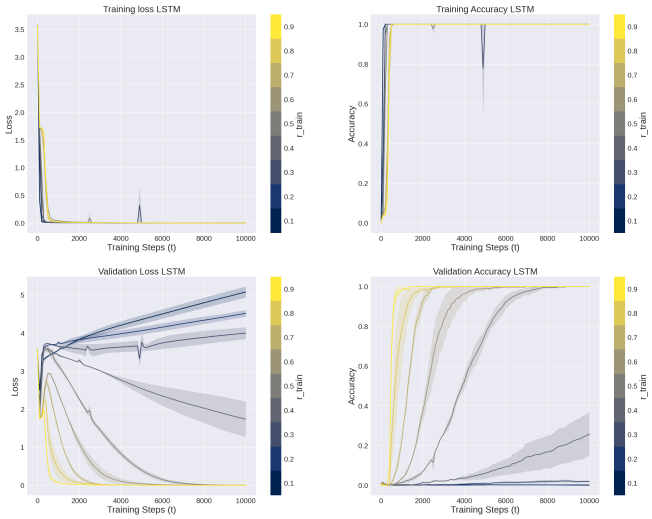
Figure 1: Question 1

Question 4.2

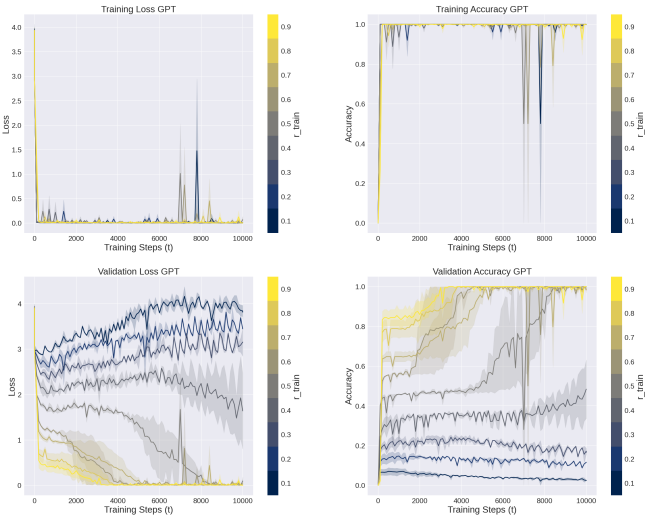
Table 1: Sample Table Caption

Metric	LSTM	GPT
$\mathcal{L}_{\text{train}}$	0.000095 ± 0.000006	0.000171 ± 1.394e-06
$\mathcal{L}_{\text{val}}$	0.002156 ± 0.001363	0.0036328 ± 0.00296579
$\mathcal{A}_{\text{train}}$	1.00 ± 0.0	1.0 ± 0.0
$\mathcal{A}_{\text{val}}$	1.00 ± 0.00	0.99896 ± 0.0010395
$t_f(\mathcal{L}_{\text{train}})$	10001 ± 0.00	8400 ± 800
$t_f(\mathcal{L}_{\text{val}})$	10001 ± 0.00	9800 ± 100
$t_f(\mathcal{A}_{\text{train}})$	300 ± 0.00	100 ± 0.0
$t_f(\mathcal{A}_{\text{val}})$	8100 ± 500	7700 ± 1000
$\Delta t(\mathcal{L})$	0 ± 0	1400 ± 100
$\Delta t(\mathcal{A})$	7800 ± 500	7600 ± 1000

Question 4.3



(a) LSTM Loss and Accuracy



(b) GPT Loss and Accuracy

Figure 2: Question 3 part a

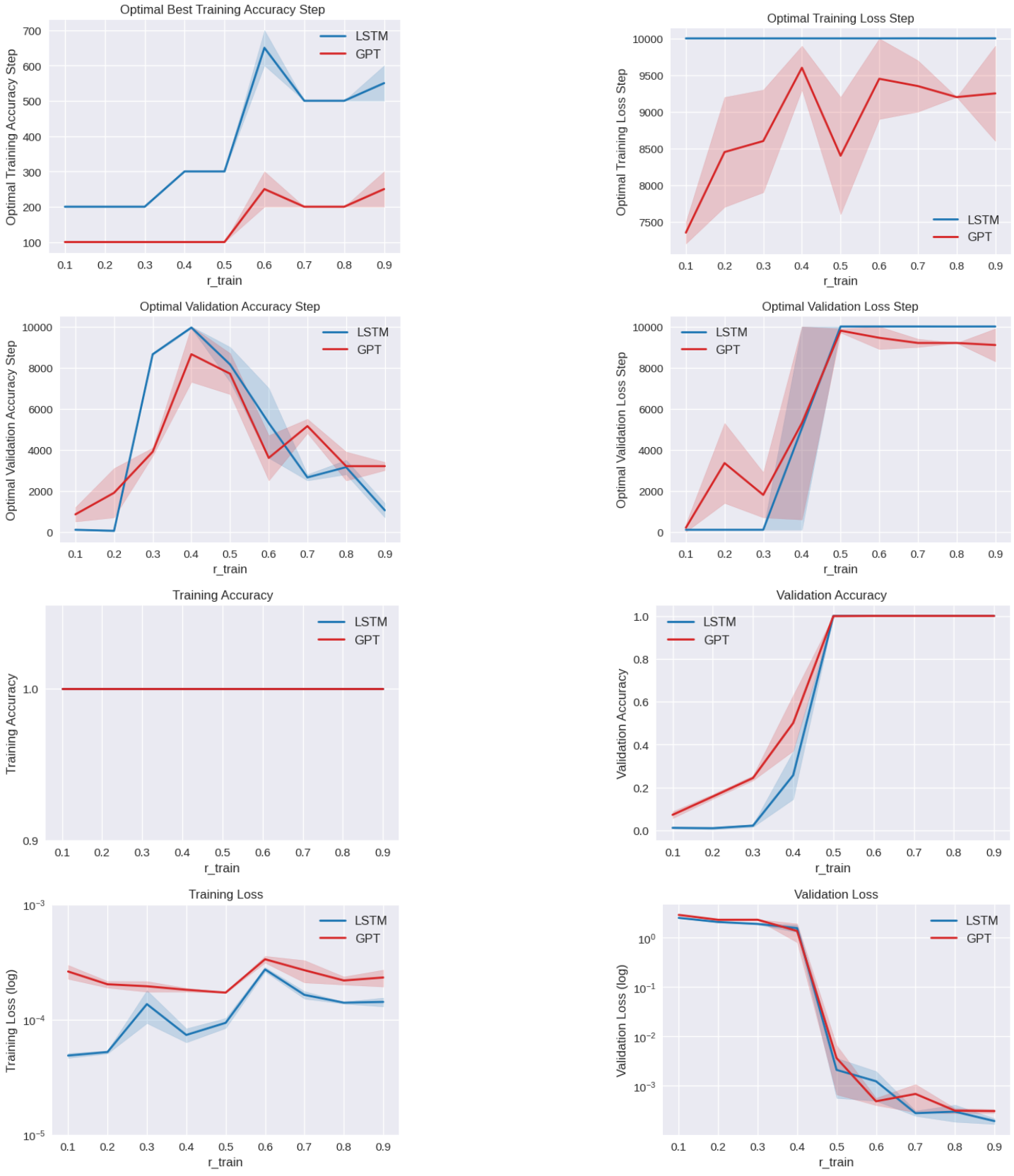


Figure 3: Question 3 part b.

Question 3 part c

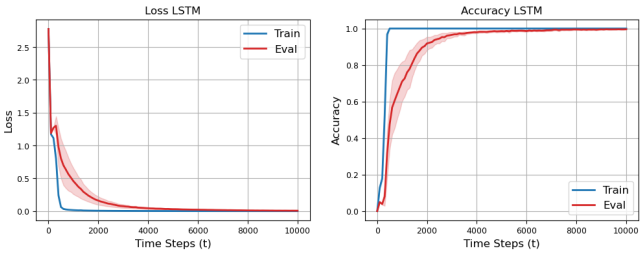
In terms of the best Validation Losses achieved by LSTM and GPT, the metrics are close for most of the r values, but it appears that for high values of r (0.7, 0.8 and 0.9) that the LSTM model achieves better performance.
In terms of Validation Accuracy however, we see that the LSTM model achieves better generalization that GPT for r values

from 0.1 to 0.4 and then at 0.5 both models achieve perfect accuracy.

Question 3 part d

The smallest r value that allows for over 0.90 Validation Accuracy is 0.5 and the largest r value that just overfits the data is 0.4 (this is where validation accuracy is less than 0.5 but train accuracy is perfect).

Question 4.4



(a) LSTM Loss and Accuracy

(b) GPT Loss and Accuracy

Figure 4: Question 4 part a

Metrics over training steps for binary and ternary operations

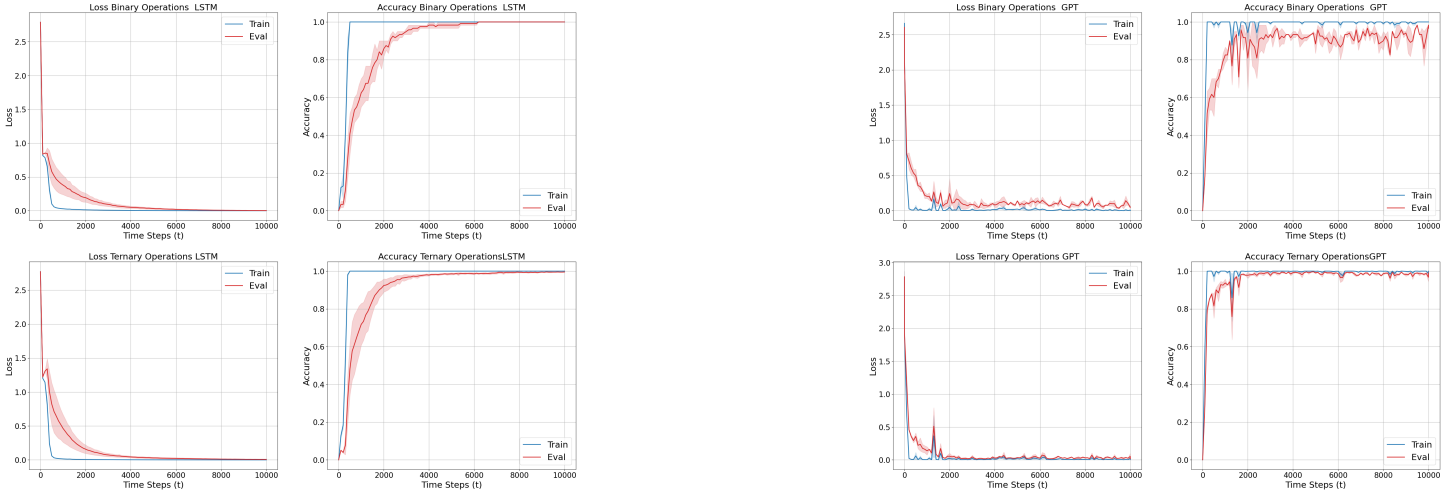


Figure 5: Question 4 part b

Question 4 part c

It appears that ternary operations are predicted more frequently than binary operations in the GPT model, but there is not much difference between the operations when looking at LSTM.

Question 4.5

Acc/Loss metrics for LSTM and GPT as functions of L and time steps

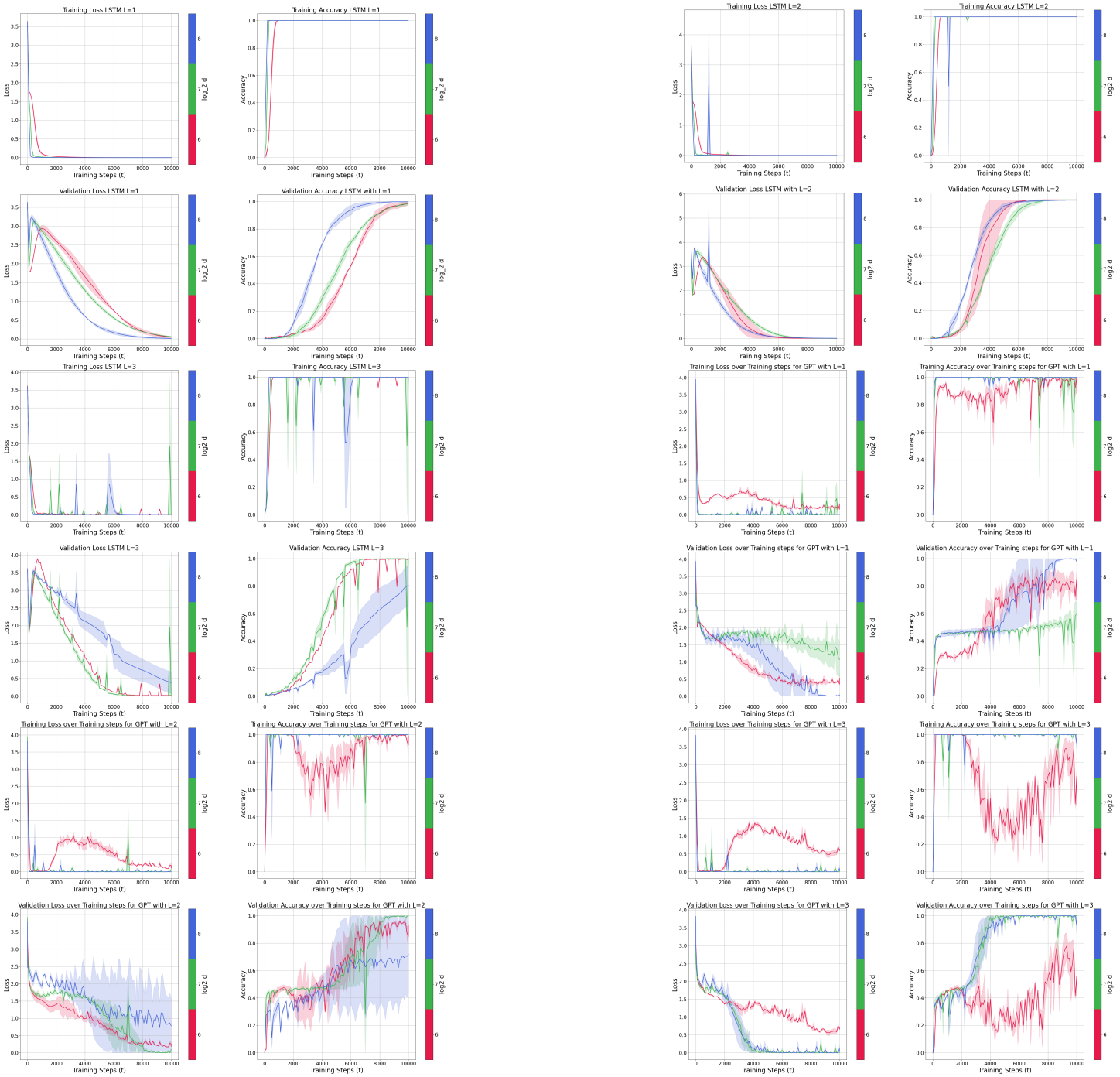


Figure 6: Question 5 part a.

Best Acc/Loss metrics for LSTM functions of L and d

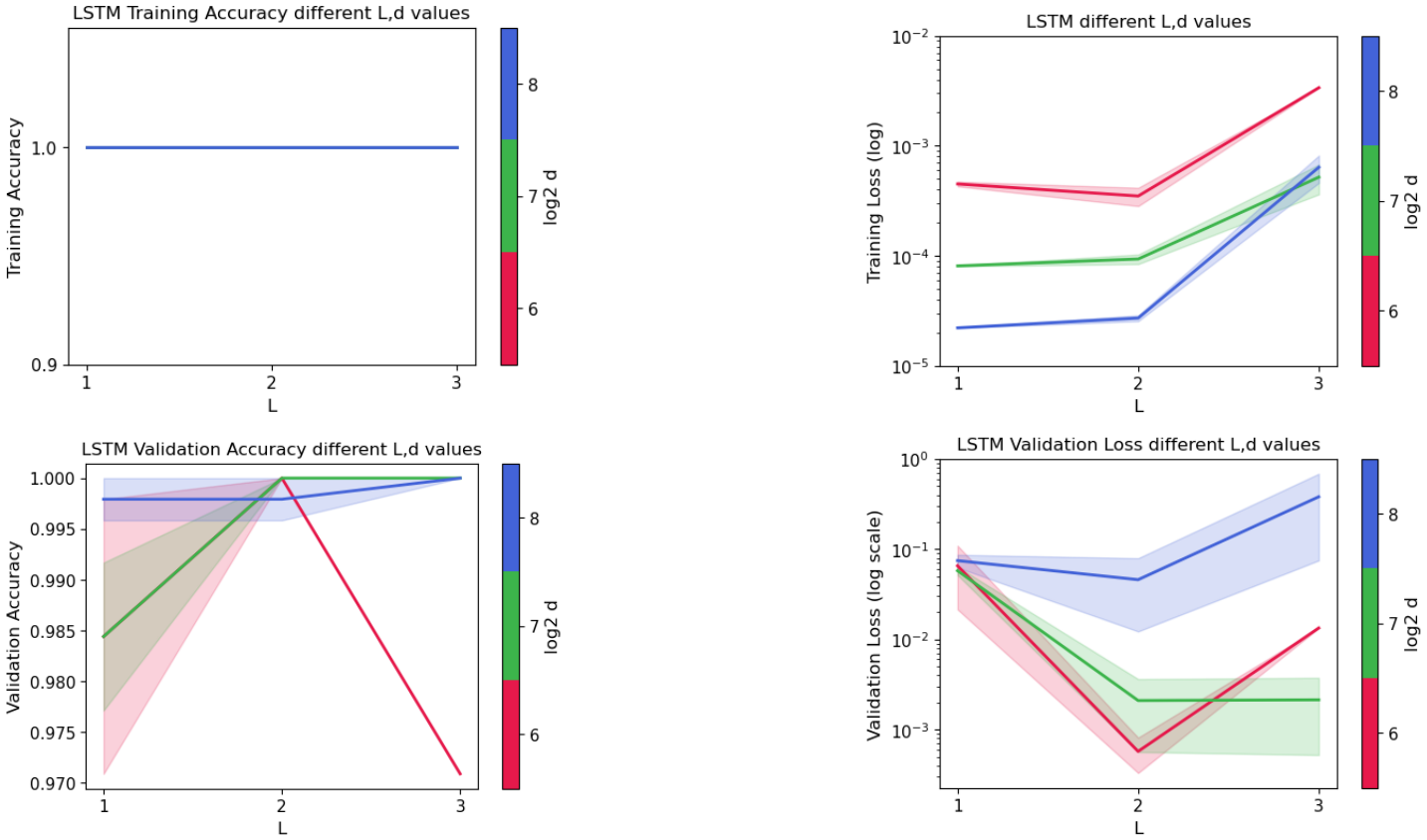


Figure 7: Question 5 part b.

Training step Best Acc/Loss metrics achieved for LSTM functions of L and d

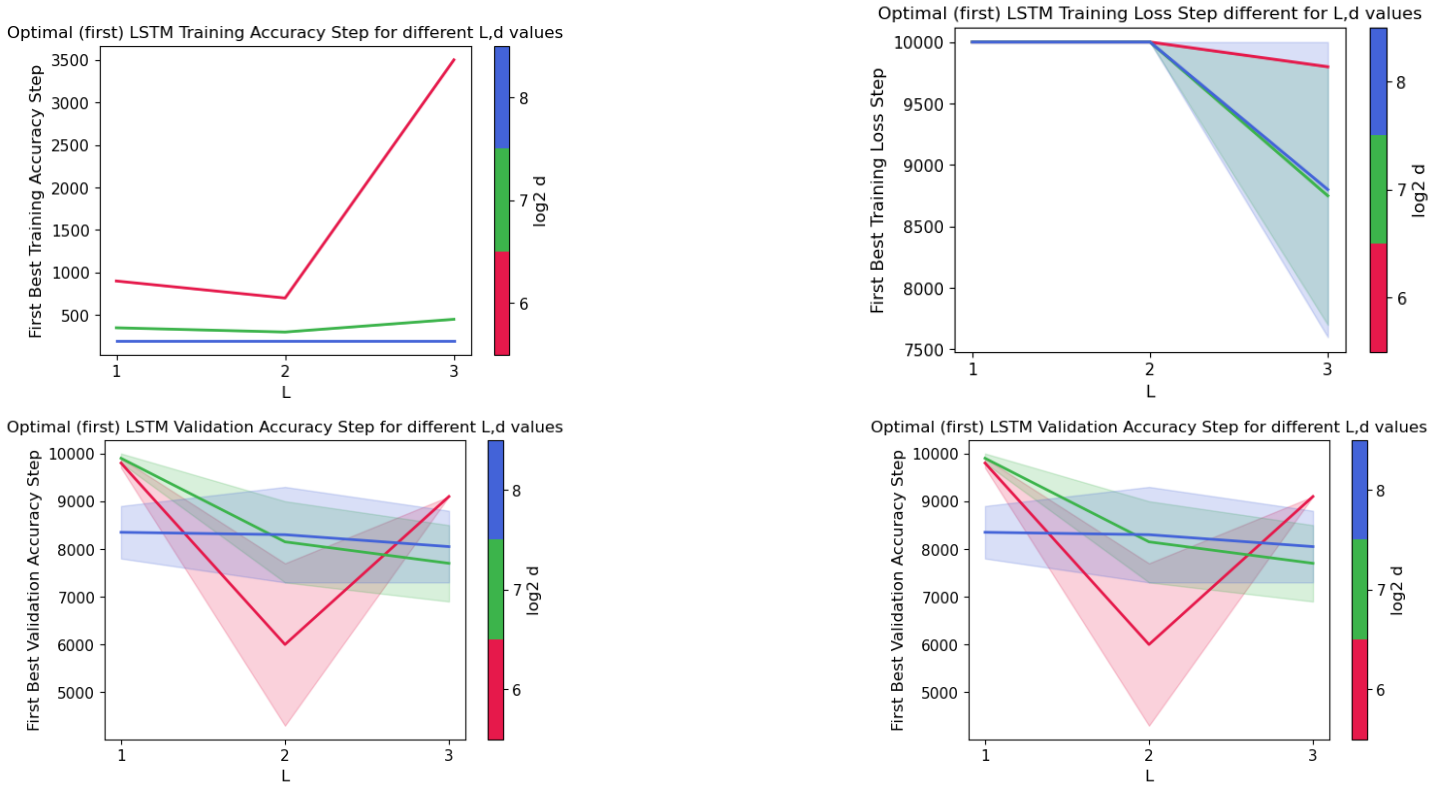


Figure 8: Question 5 part b.

Best Acc/Loss metrics for GPT functions of L and d

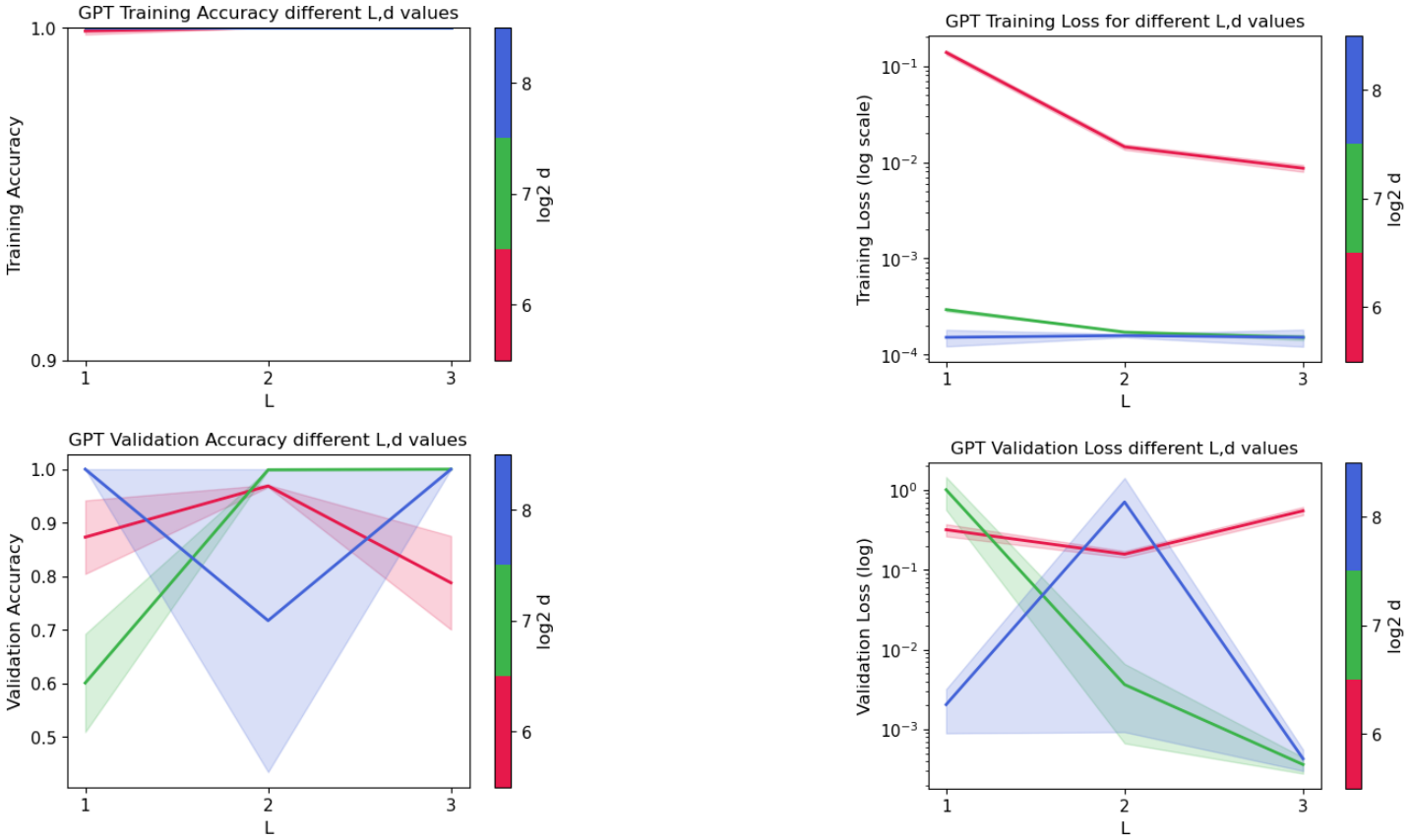


Figure 9: Question 5 part b.

Training step Best Acc/Loss metrics achieved for GPT functions of L and d

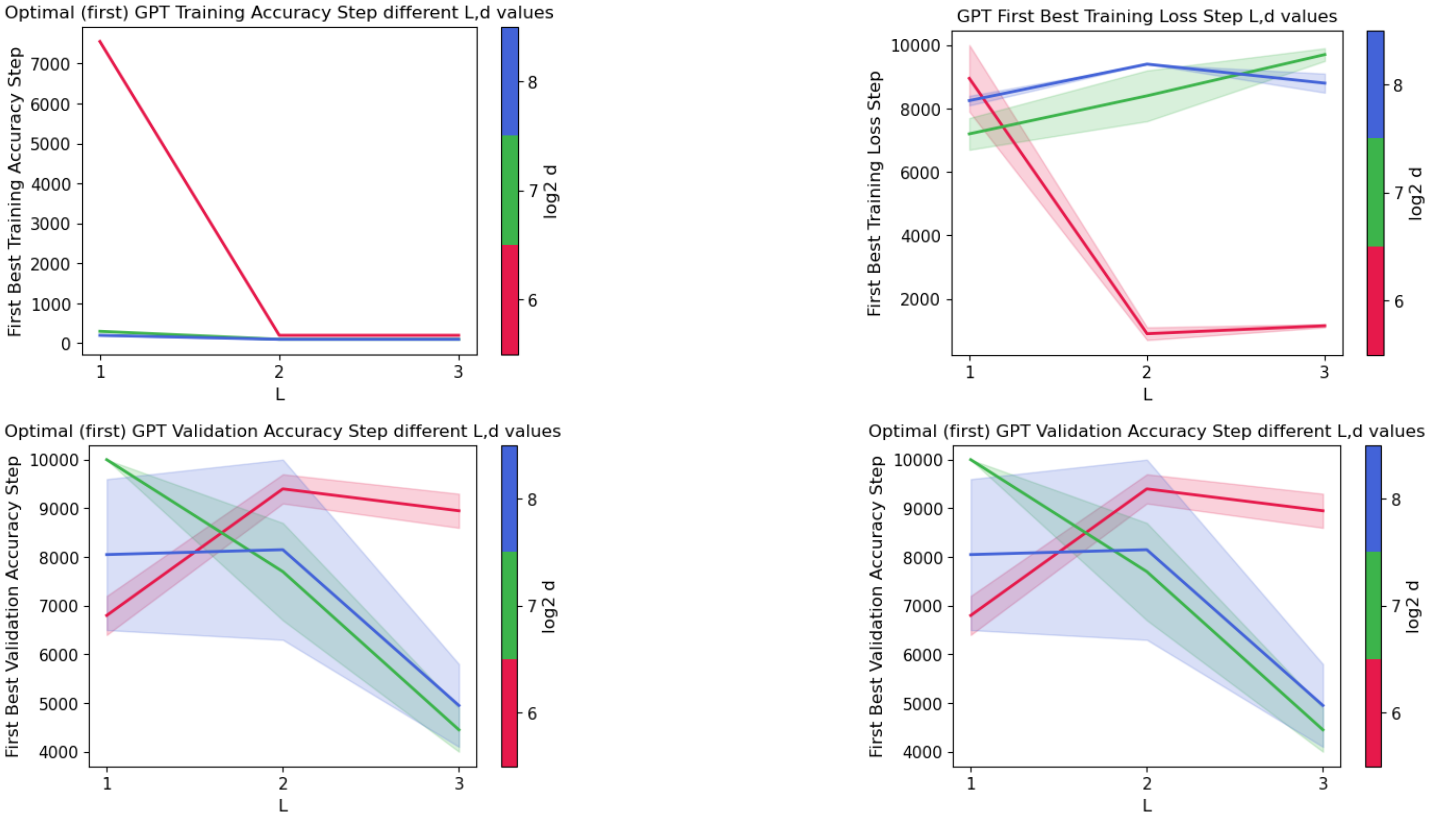


Figure 10: Question 5 part b.

Best Acc/Loss metrics achieved for LSTM as functions of number of parameters

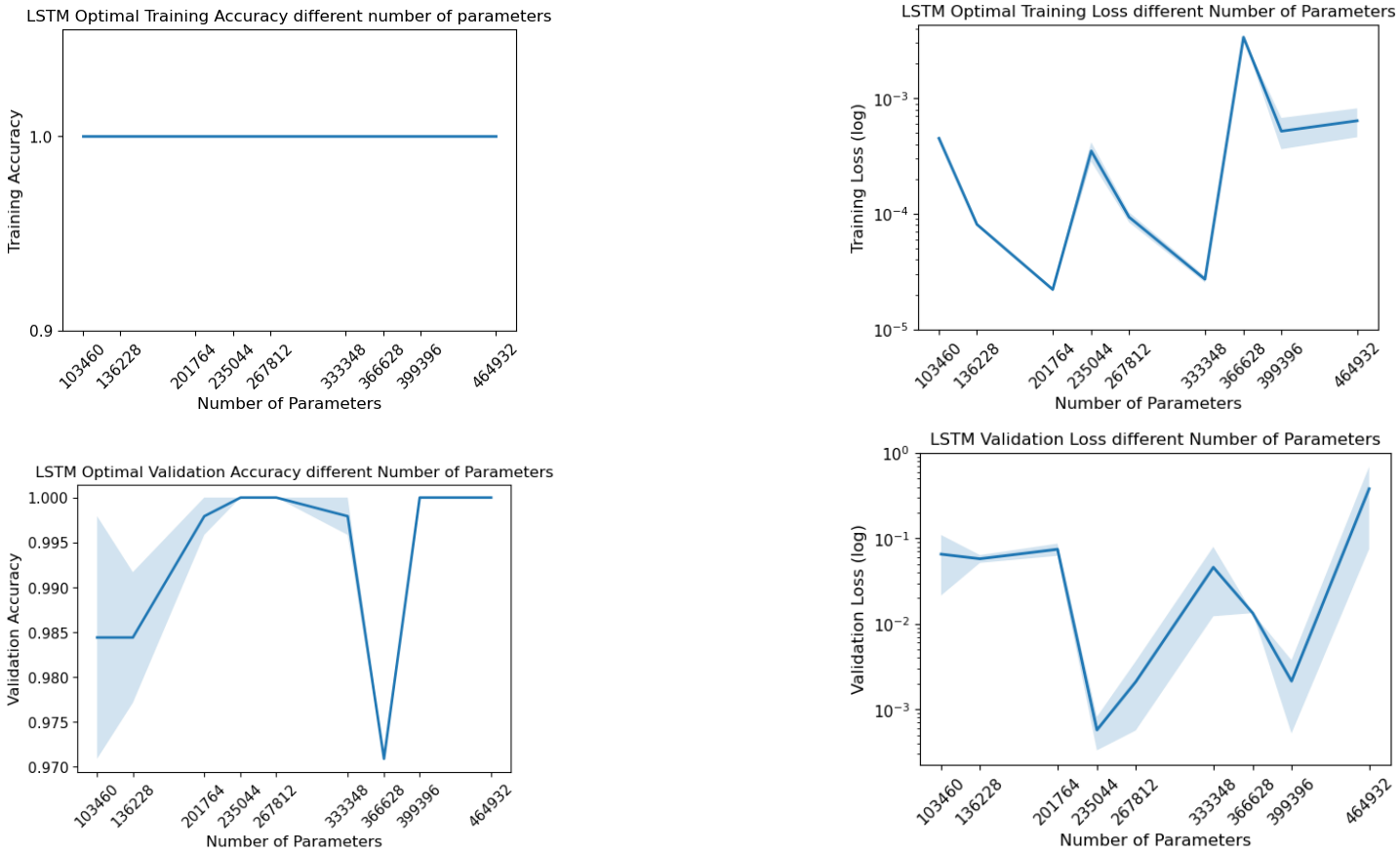


Figure 11: Question 5 part b.

Training Step Best Acc/Loss metrics achieved for LSTM as functions of number of parameters

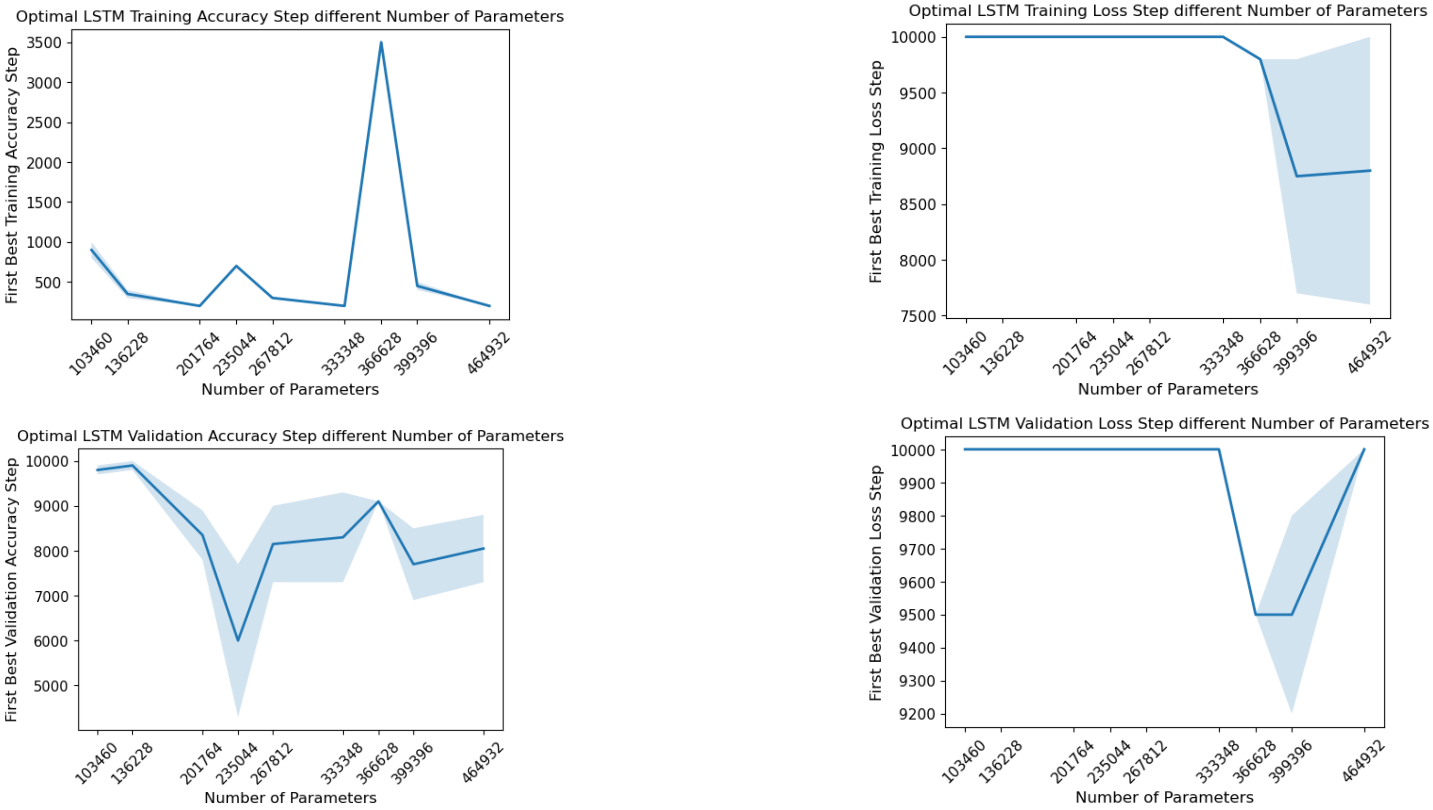


Figure 12: Question 5 part b.

Best Acc/Loss metrics achieved for GPT as functions of number of parameters

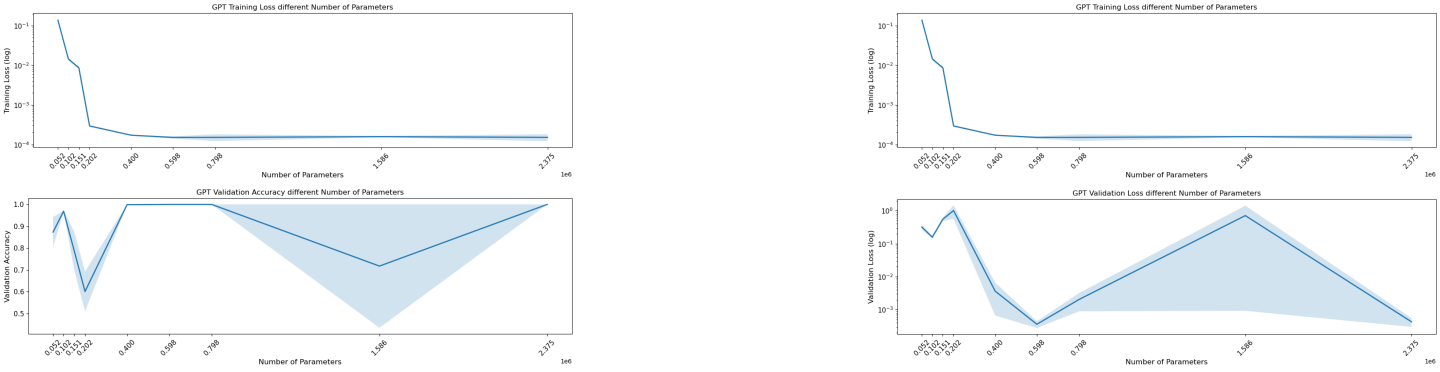


Figure 13: Question 5 part b.

Training Step Best Acc/Loss metrics achieved for GPT as functions of number of parameters

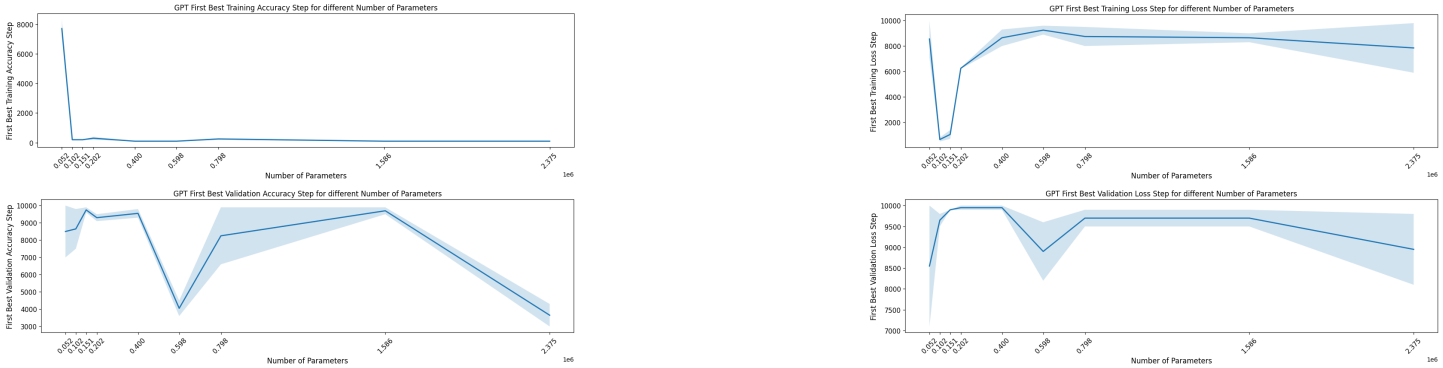


Figure 14: Question 5 part b.

Question 5 part c

For L fixed, which model scales better with d? For d fixed, which model scales better with L?Which model scales better with the number of parameters (LSTM or GPT)? You need to answer separately in terms of Lval and Aval

For fixed L, and for Validation Accuracy, it does appear that the LSTM model scales better with d, as the GPT metrics are more volatile in d for same L values.

Similarly, for fixed L and for Validation Loss, it does appears again that the LSTM model scales better with d, as it consistently has $2^8 > 2^7 > 2^6$ ranking in d (GPT volatile results)

For fixed d and for Validation Accuracy, it seem that for d=2^8, LSTM scales better in L and for 2^6 GPT scales better in L. Both scale poorly when d=2^7

For fixed d and for Validation Loss, it seem that for d=2^8, again LSTM scales better in L and GPT scales better in L for 2^6. Both scale poorly when d=2^7

When looking at which model scales better with number of parameters for Validation Accuracy, it appears LSTM is superior as its metrics are much more stable than GPT's, whose validation accuracy drops to around 0.75 at one point, when LSTM's is always is 0.97 at the lowest.

When looking at which model scales better with number of parameters for Validation Loss though, it looks like the GPT model scales better, as the LSTM validation loss is very volatile compared to GPT.

Question 4.6

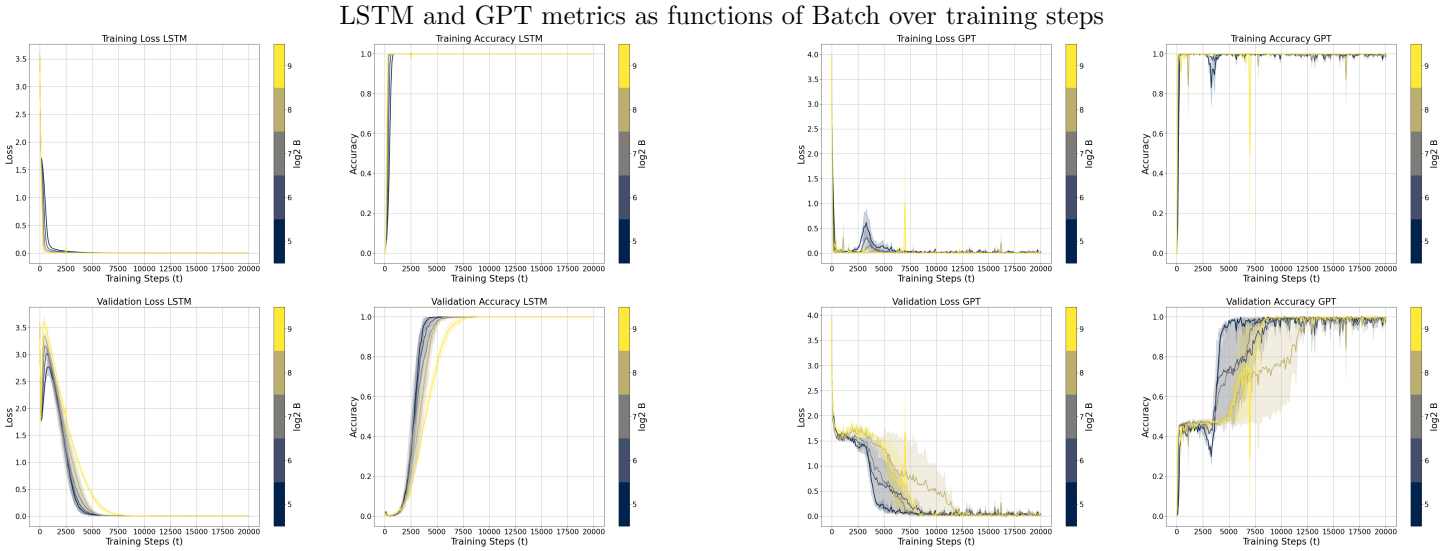


Figure 15: Question 6 part a.

Best Acc/Loss metrics achieved for LSTM vs GPT

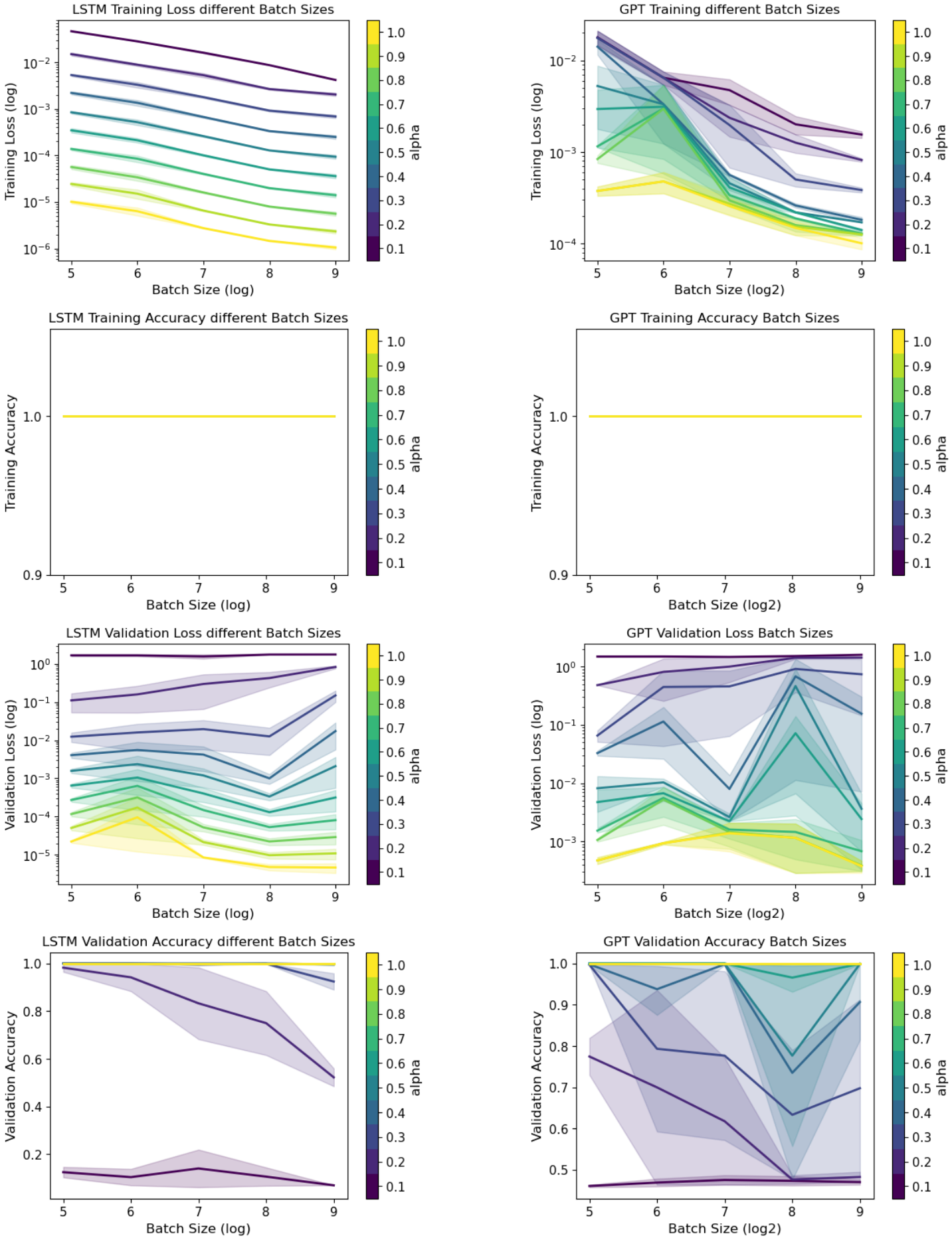


Figure 16: Question 6 part b.

Earliest Step Optimal Acc/Loss metrics were achieved for LSTM vs GPT

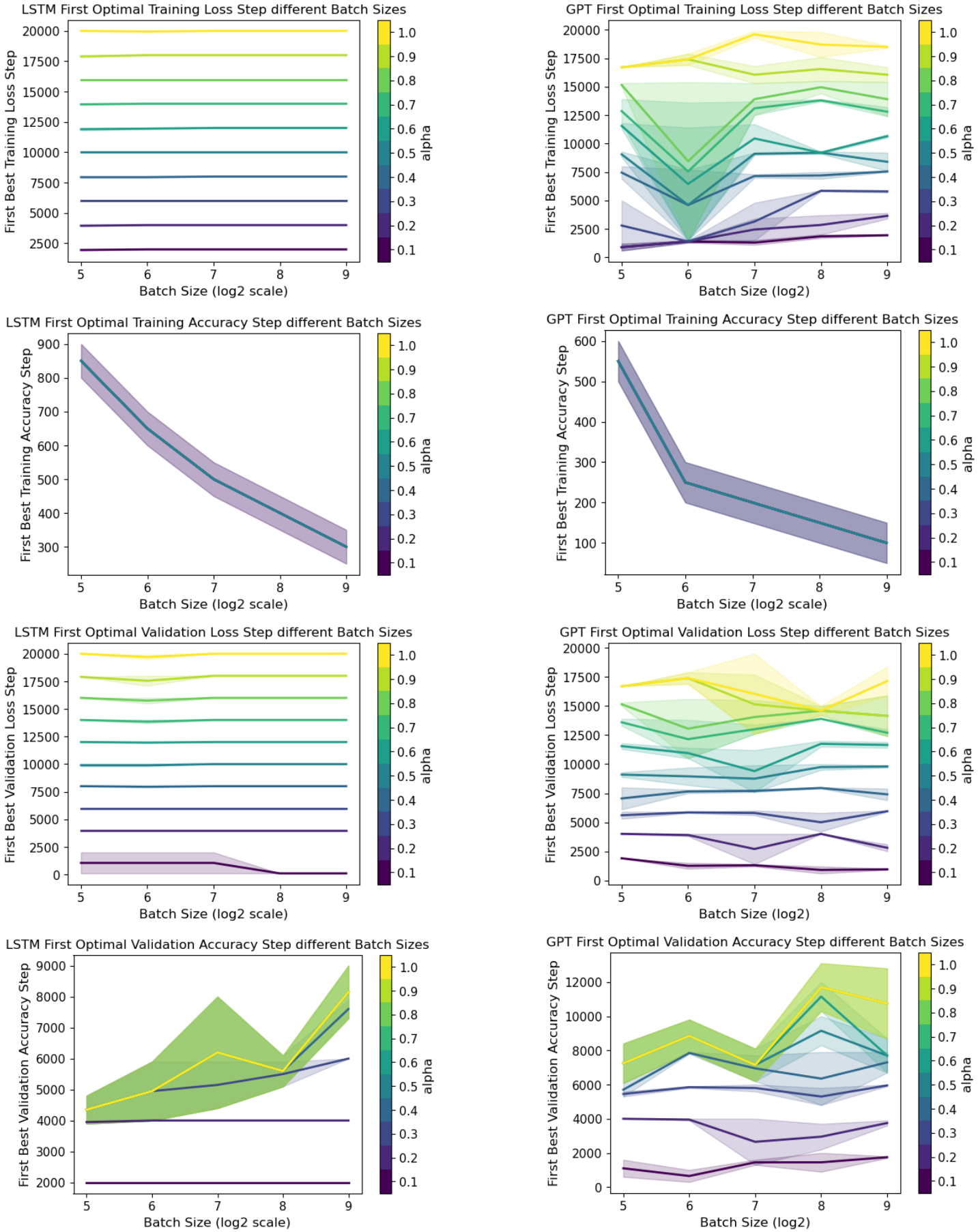


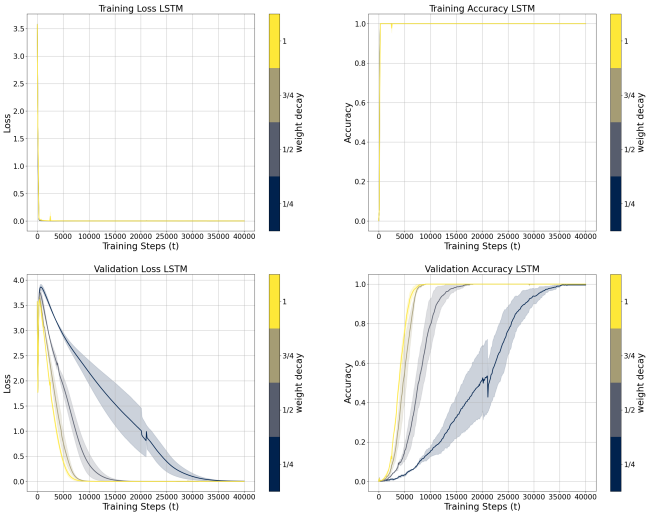
Figure 17: Question 6 part b.

Question 6 part c

There is a pretty clear trend for both Validation Loss and Accuracy across both models. When α is increased, we achieve better loss and accuracies, but when B is increased, we see worse loss and accuracies.

Question 4.7

LSTM metrics as functions weight decay over training steps and L2 Norm over training steps for different weight decays



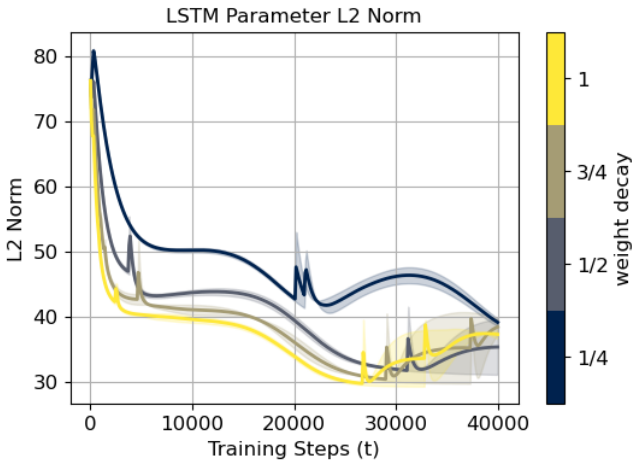


Figure 18: Question 7 part a.

LSTM Optimal Metrics as functions of Weight Decay

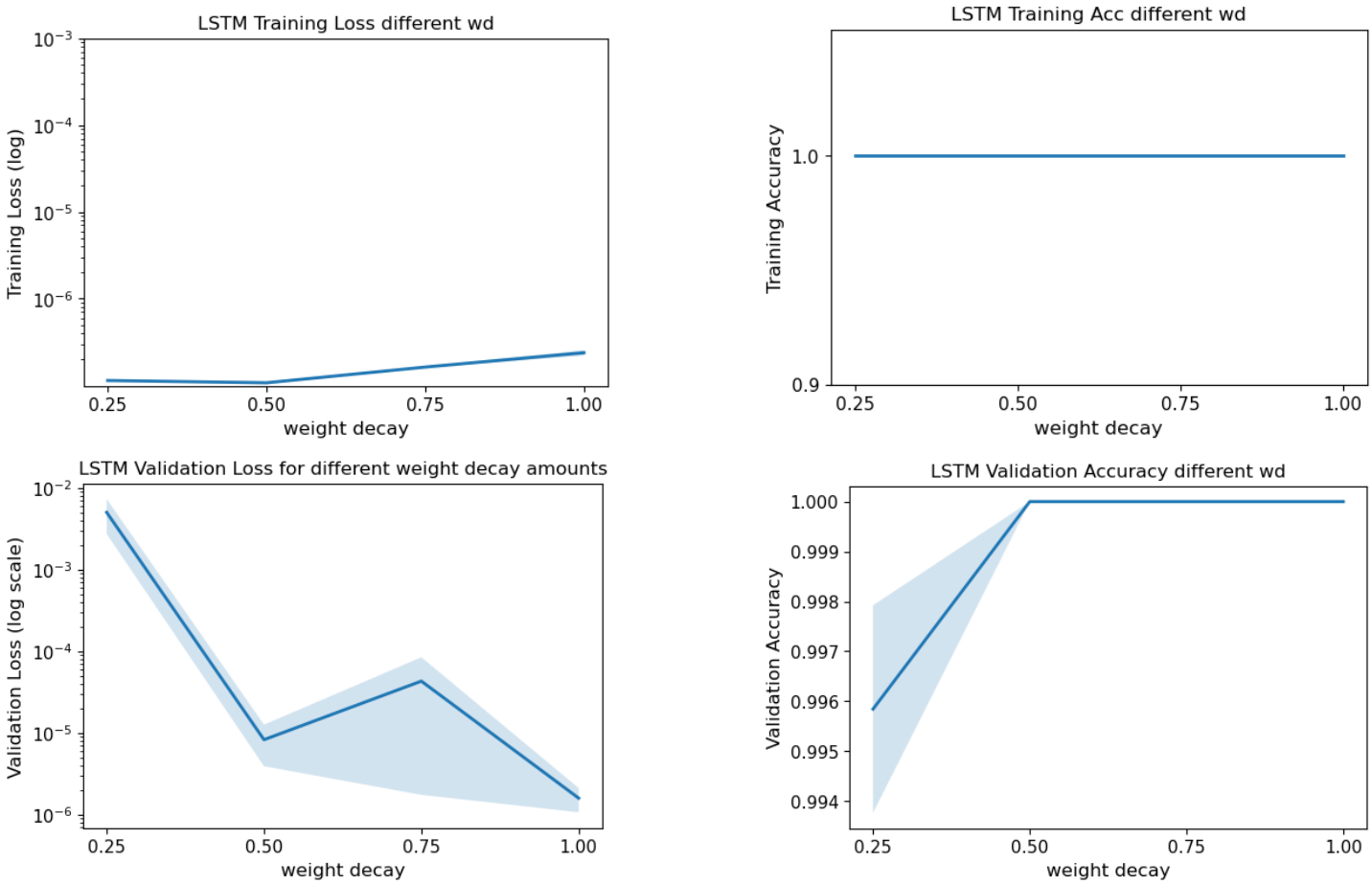


Figure 19: Question 7 part b.

LSTM Optimal Metrics as functions of Weight Decay

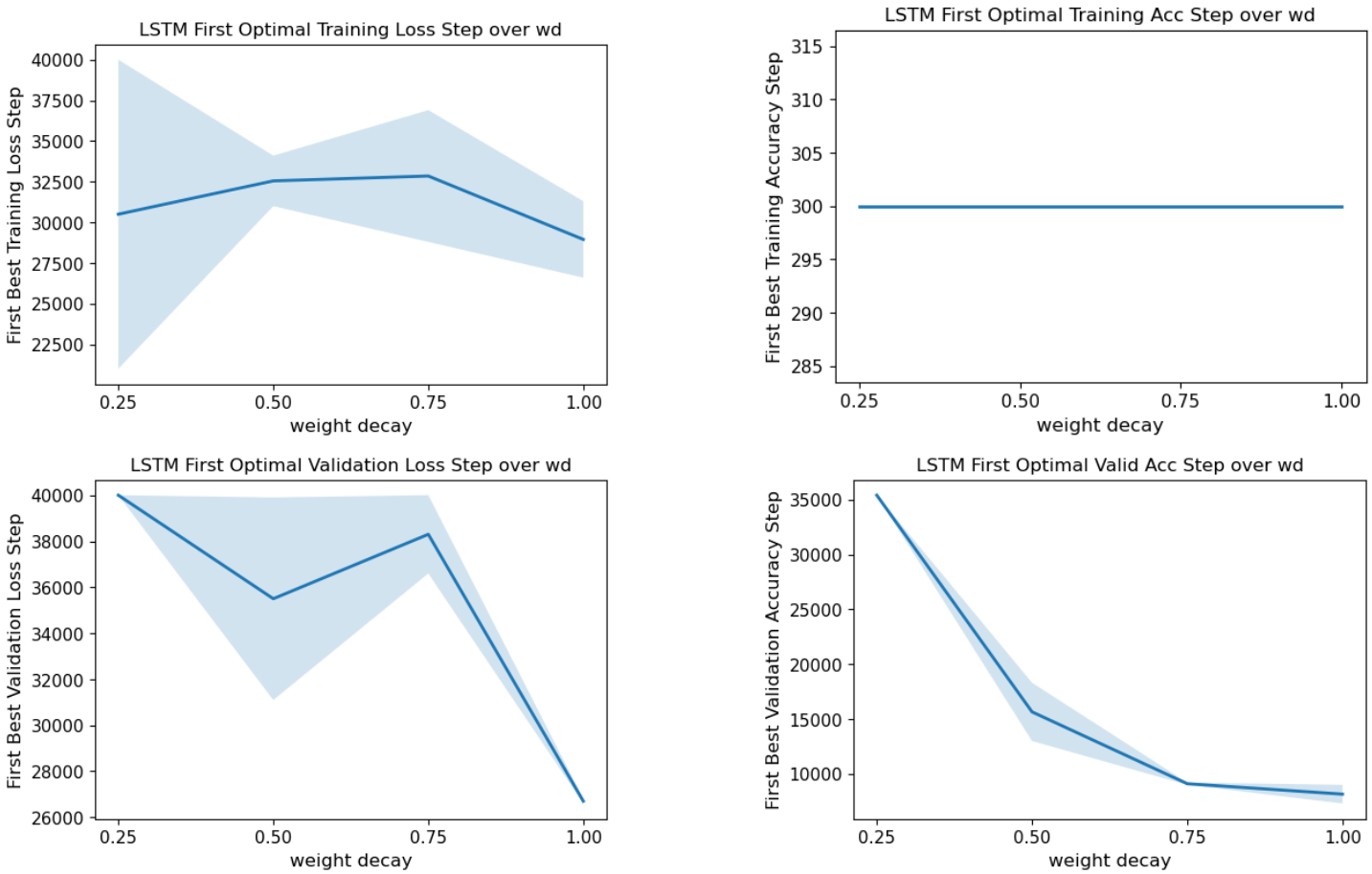


Figure 20: Question 7 part b.

Question 7 part c

For both Validation Loss and Validation Accuracy, the trend is that your performance gets better as weight decay is increased.

For the earliest training step you achieve best performance, we again see that for both Validation Loss and Validation Accuracy more weight decay means you achieve best performance sooner in training.

Question 7 part d

There is a trend that once the parameter norms have stabilized, the models achieve good generalizability. For example, the norms for weight decay 1,0.75 and 0.5 are flat at around 10000 training steps and this is the first point where all 3 models achieve Validation Accuracy of at least 0.80. The only exception is with weight decay of 0.25 that takes longer to generalize well. But its still true that it achieves good generalizability when it's L2 norm is flat (25 000 steps approximately for 0.25 weight decay).

Question 4.8

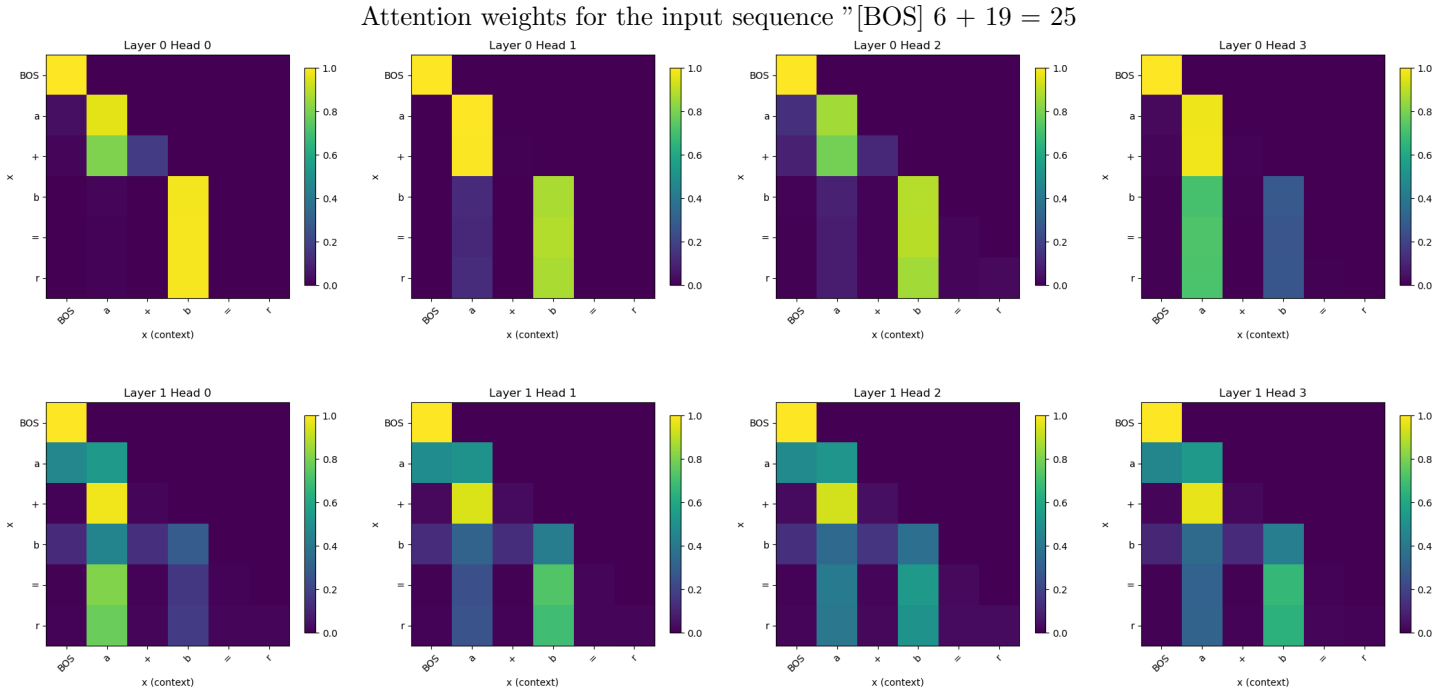


Figure 21: Question 8 a

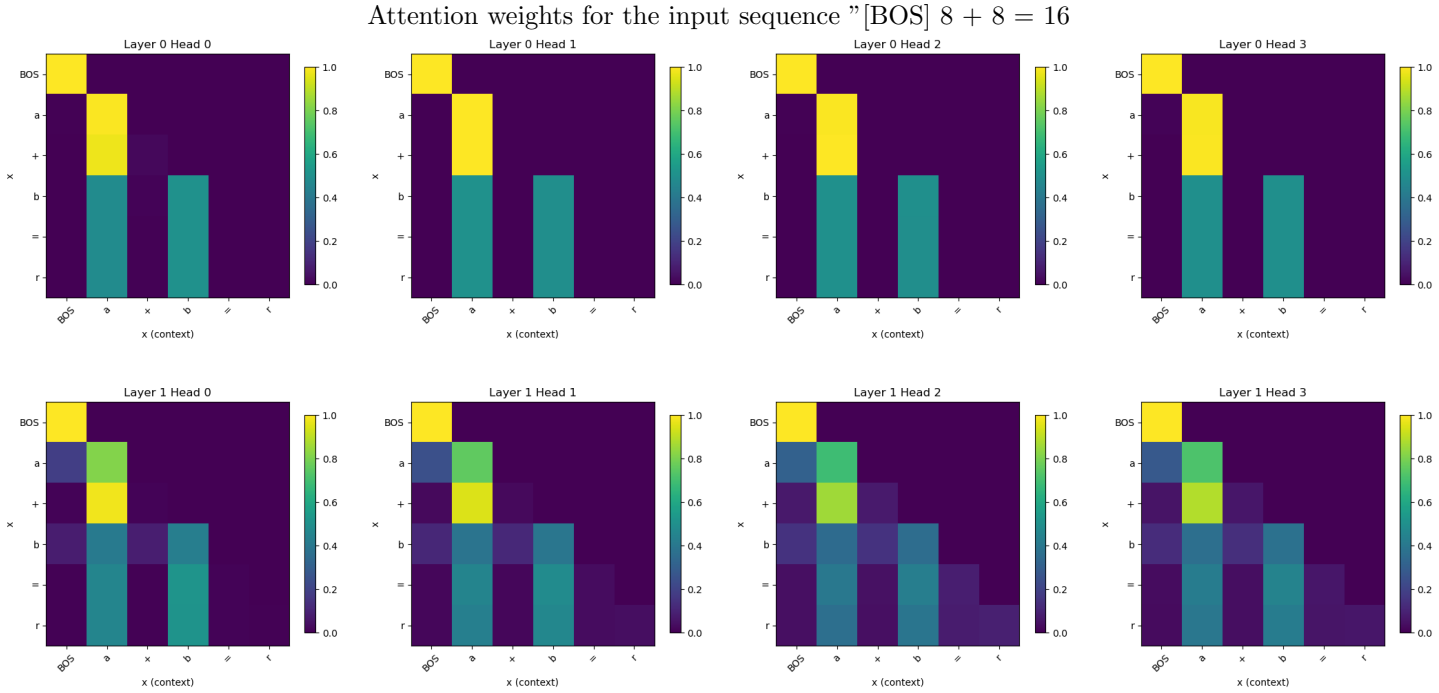


Figure 22: Question 8 a

Question 8 b

There is a pretty clear trend that the tokens in the positions of the integers being summed (a and b) are the tokens that are being attended to the most and the tokens for [BOS], + and = do not get attended too much. This is a trend that persists across all the heads and layers. Likely the reason for this is because we are only learning a single mathematical equation. Every input has a + and an = so they are not useful for predicting the solution of the equation. The real pieces of information that are useful for the predictive task are the integers at the "a" and "b" positions.