Q1

① $i=j$

$$\frac{\partial S_i}{\partial z_j} = \frac{\partial}{\partial z_i}\left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right) = \frac{1}{\left(\sum_k e^{z_k}\right)^2}\cdot\left(e^{z_i}\cdot\sum_k e^{z_k} - e^{z_i}\cdot e^{z_i}\right)$$

$$= \frac{e^{z_i}\left(\sum_k e^{z_k} - e^{z_i}\right)}{\left(\sum_k e^{z_k}\right)^2}$$

$$= \frac{e^{z_i}}{\sum_k e^{z_k}}\cdot\left(1 - \frac{e^{z_i}}{\sum_k e^{z_k}}\right)$$

$$= S_i(1-S_i)$$

② $i \neq j$

$$\frac{\partial S_i}{\partial z_j} = \frac{\partial}{\partial z_j}\left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right) = \frac{1}{\left(\sum_k e^{z_k}\right)^2}\cdot\left(0 - e^{z_j}\cdot e^{z_i}\right)$$

$$= -\frac{e^{z_j}}{\sum_k e^{z_k}}\cdot\frac{e^{z_i}}{\sum_k e^{z_k}}$$

$$= -S_iS_j$$

$$\therefore \frac{\partial \vec{S}}{\partial \vec{z}} = \begin{bmatrix} S_1(1-S_1) & -S_1S_2 & -S_1S_3 & \cdots & -S_1S_K \\ -S_2S_1 & S_2(1-S_2) & -S_2S_3 & \cdots & -S_2S_K \\ \vdots & & \ddots & & \vdots \\ -S_KS_1 & -S_KS_2 & \cdots & & S_K(1-S_K) \end{bmatrix}$$

Q2.

Assume $W = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$

① For 'AND',

$$\begin{cases} w_1 \cdot 0 + w_2 \cdot 0 + b < 0 \\ w_1 \cdot 0 + w_2 \cdot 1 + b < 0 \\ w_2 \cdot 0 + w_1 \cdot 1 + b < 0 \\ w_1 \cdot 1 + w_2 \cdot 1 + b \geq 0 \end{cases} \Rightarrow \begin{cases} b < 0 \\ w_1 + b < 0 \\ w_2 + b < 0 \\ w_1 + w_2 + b \geq 0 \end{cases} \Rightarrow \begin{cases} b < 0 \\ w_1 < -b \\ w_2 < -b \\ w_1 + w_2 \geq -b \end{cases}$$

Assume $w_1 = w_2$, Let $b = -2$.

We can get $w_1 < -2$. $w_1 \geq -1$.

∴ One solution is ~~$W = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$~~ $W_{AND} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $b_{AND} = -2$.

② For 'OR', similarly

$$\begin{cases} b < 0 \\ w_2 + b \geq 0 \\ w_1 + b \geq 0 \\ w_1 + w_2 + b \geq 0. \end{cases} \Rightarrow \begin{cases} b < 0 \\ w_2 \geq -b \\ w_1 \geq -b \\ w_1 + w_2 \geq -b. \end{cases}$$

Assume $w_1 = w_2$. Let $b = -1$.

One solution is $W_{OR} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $b_{OR} = -1$

Q3.

Assume XOR can be represented using a linear model with form provided. Assume $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$, $b = b$

Then, $\begin{cases} w_1 \cdot 0 + w_2 \cdot 0 + b < 0 \\ w_1 \cdot 0 + w_2 \cdot 1 + b \geq 0 \\ w_2 \cdot 0 + w_1 \cdot 1 + b \geq 0 \\ w_1 \cdot 1 + w_2 \cdot 1 + b < 0 \end{cases} \Rightarrow \begin{cases} b < 0 \\ w_2 \geq -b \\ w_1 \geq -b. \\ w_1 + w_2 < -b. \end{cases}$

$\because \begin{cases} w_2 \geq -b \\ w_1 \geq -b \end{cases} \therefore w_1 + w_2 \geq -2b \quad \therefore w_1 + w_2 \geq -2b > 0. \Rightarrow$

Also, $\because w_1 + w_2 + b < 0 \quad \therefore w_1 + w_2 < -b.$

$-b > w_1 + w_2 \geq -2b. \Rightarrow -b > -2b. \Rightarrow 0 > -b \Rightarrow b > 0,$ which is conflicted with $b < 0$.

Therefore, the assumption is wrong. XOR cannot be represented using a linear model with the same form provided in Q2.

Assignment 1 Theory Problem Set

DO NOT TAG

Name: Haoming Zhang
GT Email: hzhang961@gatech.edu

Theory PS Q1. Feel free to add extra slides if needed.

Theory PS Q2. Feel free to add extra slides if needed.

Theory PS Q3. Feel free to add extra slides if needed.

Assignment 1 Paper Review

DO NOT TAG

Name:
GT Email:

Provide a short preview of the paper of your choice.

In summary, the paper presents an innovative approach to neural network design - Weight Agnostic Neural Networks(WANN). WANN are neural network architectures that can perform certain tasks with randomly assigned weights parameters. The researcher populates the connection of a network with a single shared weight parameter and measures its expected performance. They found these minimal architectures can successfully perform reinforcement learning tasks and achieve higher than chance accuracy on MNIST using random weights.
The paper proposes a search method for these neural network architectures that deemphasizes the importance of weights. The method aims to find networks that can be described with a minimal description length, preferring simple networks with similar performance levels, taking into account the size of the network as well as its performance when ranking the architectures.
WANNs are evaluated on continuous control tasks and the MNIST dataset. The results show that WANNs can perform effectively with random shared weights on a wide range and their architectures encode a strong bias toward solutions. The paper also demonstrates that the performance of WANNs can be further improved by tuning the shared weight or individual weights.
In addition, the paper discusses the potential applications of WANNs in few-shot learning, continual learning, and the exploration of new neural network building blocks beyond gradient-based methods.
This paper suggests that further research should focus on discovering new architectures and building blocks with strong inductive biases and can be trained with more diverse algorithms.
From my point of view, the method proposed by this paper also has some limitations. One of them is it may be difficult to find complicated architectures like CNNs by using this topology search approach.

Paper specific Q1. Feel free to add extra slides if needed.

From the perspective of search, the paper also implements a search method. However, instead of searching the space of weights, they are searching the space of architectures. The researcher used three operators to search for neural network topologies, namely insert node, add connection, or change activation.
In this paper, Networks are evaluated over several rollouts. At each rollout a value for the single shared weight is assigned and the cumulative reward over the trial is recorded. The population of networks is then ranked according to their performance and complexity. The highest ranking networks are then chosen probabilistically and varied randomly, modified by one of these three operators, to form a new population, and the process repeats.
As we know, Convolutional Neural Networks (CNNs) are designed to simulate and capture hierarchical structures in visual data, going from individual pixels to patterns, blocks, and finally the entire picture, reflecting the hierarchical feature learning capability of CNNs, making them particularly effective for image-related tasks.
Similarly, by implementing topology search, the researchers is searching for neural network architectures with strong inductive biases for certain tasks, such as continuous control tasks and the MNIST experiment.

Paper specific Q2. Feel free to add extra slides if needed.

According to the experimental results section, in contrast to the conventional fixed topology networks used as baselines, which only produce useful behaviors after extensive tuning, WANNs perform better with random shared weights. That being said, representational power of architectures is robust to magnitude of the weights and the method for determining the weights. Thus, the representational power of architectures given a fixed method for determining weights is very satisfying for certain tasks. Additionally, when tuning WANNs, it becomes trivial to tune the single shared weight, without requiring the use of gradient-based methods.

In some cases, it is possible for WANNs to discover architectures with similar representational power as traditional neural networks, but it is not guaranteed. WANNs could find simple architectures that can perform certain tasks well. But in other cases, e.g. compared with CNNs on image recognition tasks, it may not have equal representational power. I think the reason is that the it is hard to find the optimal complicated architectures by using topology search. However, for certain tasks which can be completed with neural networks of simple structures, WANN may have equal or better representational power comparing to traditional neural networks because it has a better baseline before tuning the parameters.

Assignment 1 Writeup

DO NOT TAG

Name:
GT Email:

Two-Layer Neural Network

DO NOT TAG

1. Learning Rates

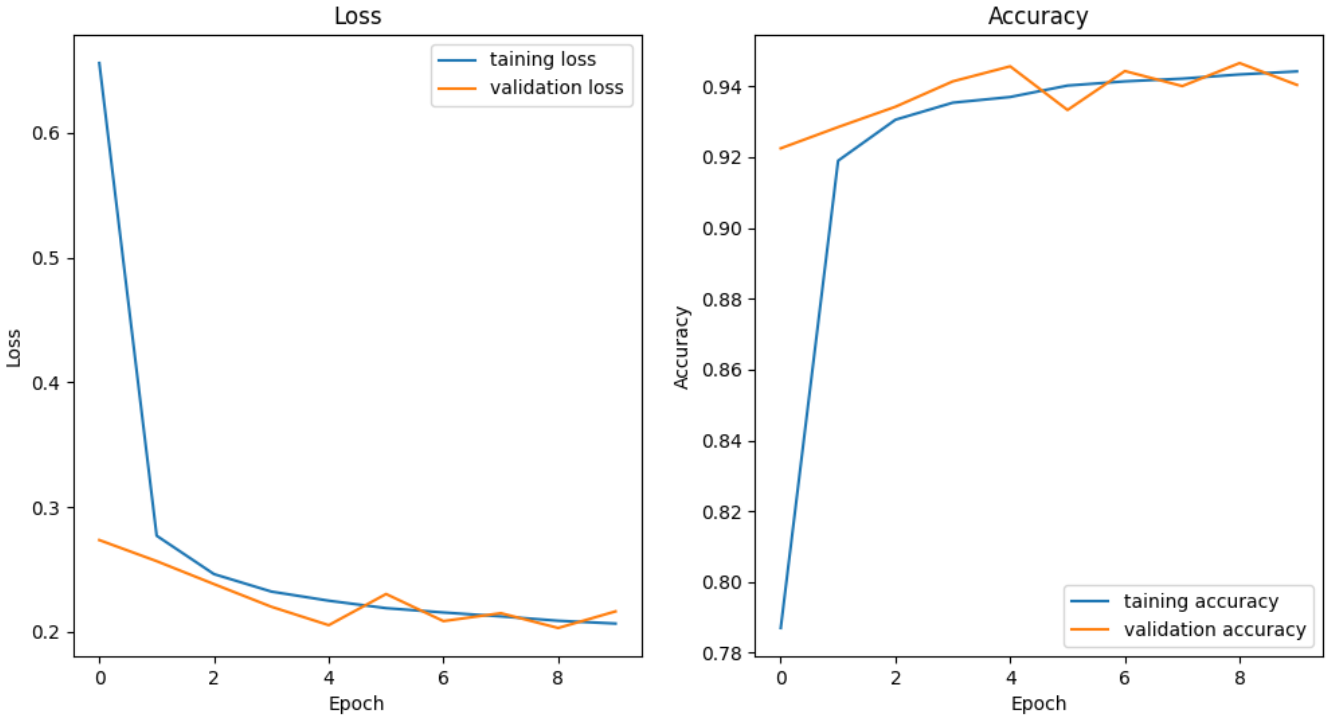
Tune the learning rate of the model with all other default hyper-parameters fixed. Fill in the table below:

	lr=1	lr=1e-1	lr=5e-2	lr=1e-2
Training Accuracy	0.9434	0.9214	0.9084	0.7305
Test Accuracy	0.9467	0.9244	0.9136	0.7588

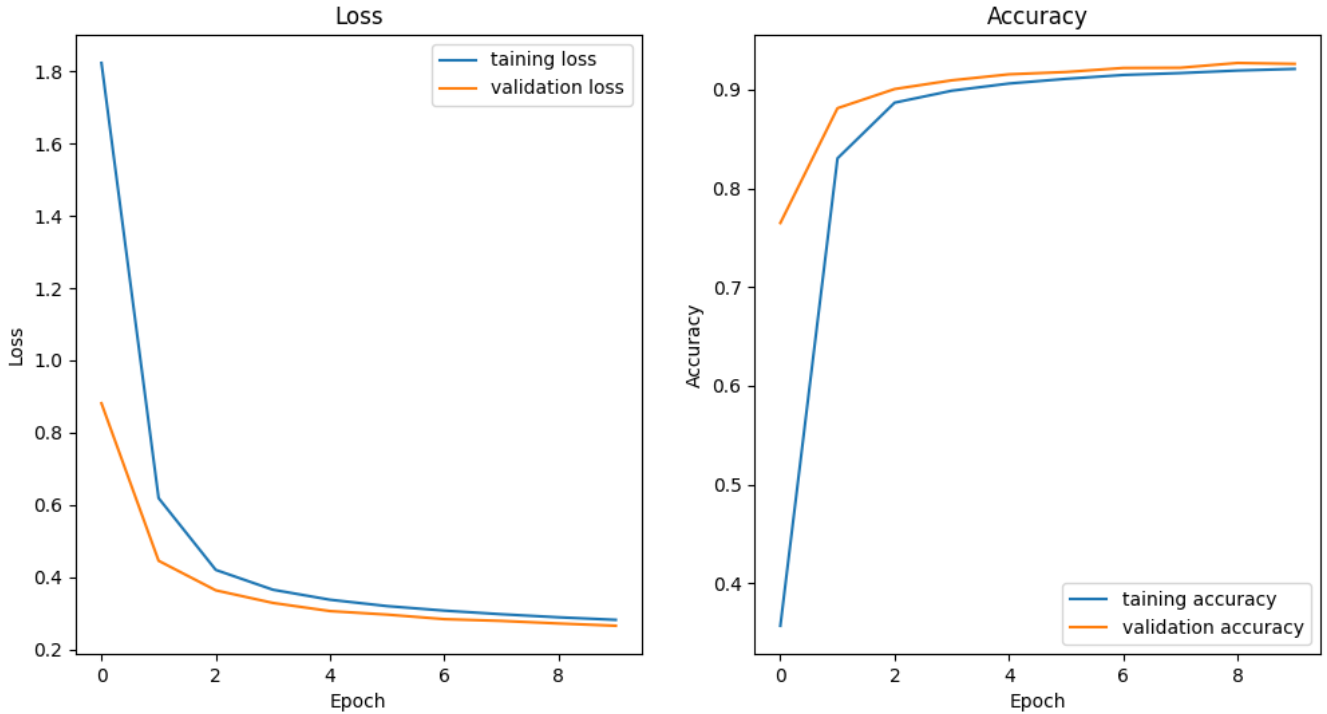
1. Learning Curve

Plot the learning curves using the learning rates from the previous slide and put them below (you may add additional slides if needed).

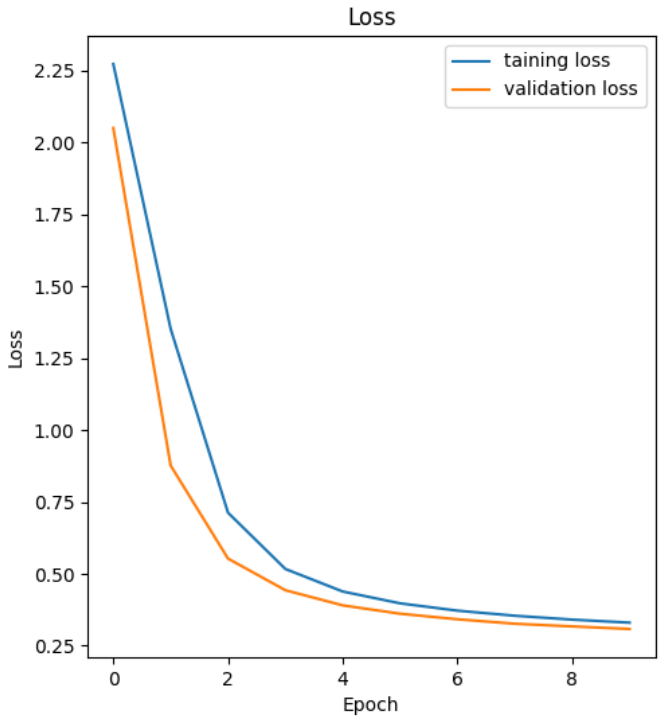
Learning Rate = 1

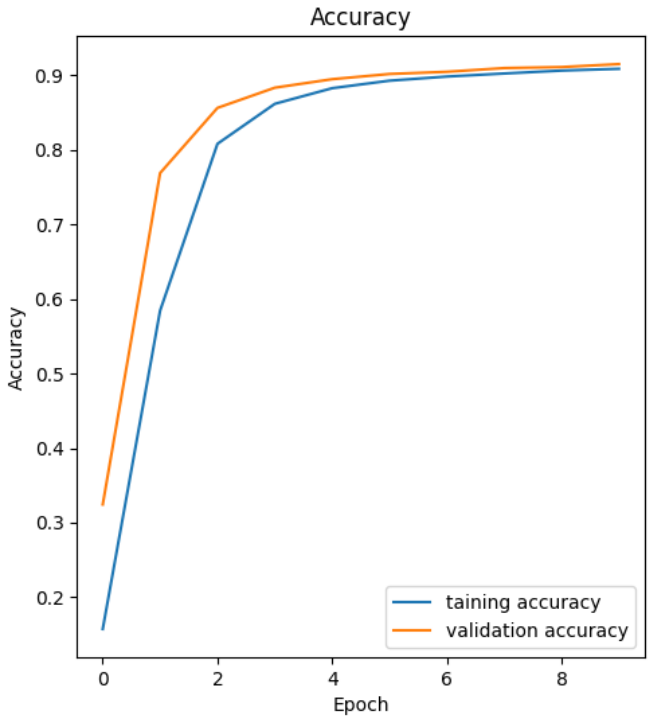


Learning Rate = 1e-1

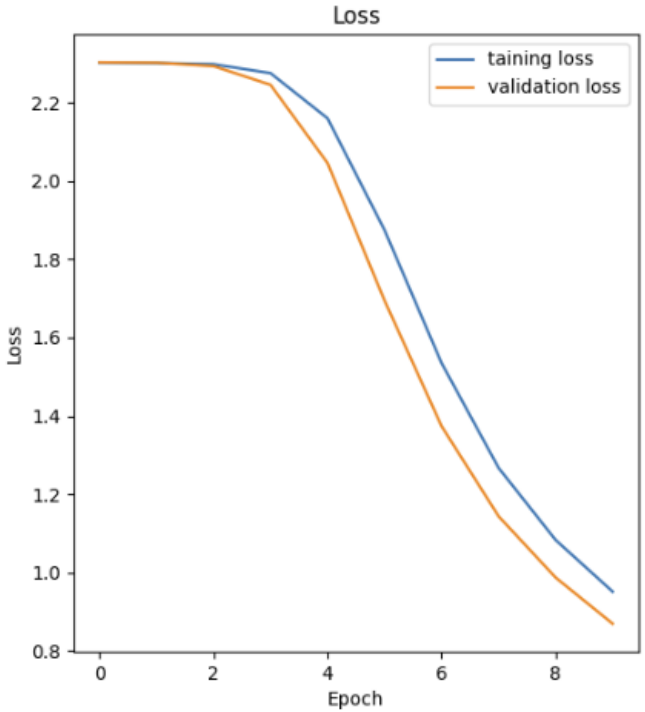


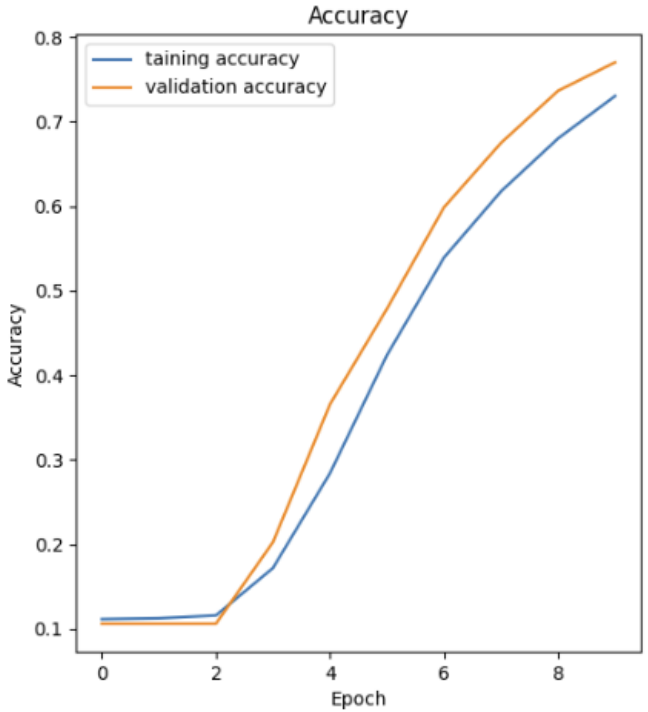
Learning Rate = 5e-2





Learning Rate = 1e-2





1. Learning Rates

Describe and Explain your findings: *Explanation should go into **WHY** things work the way they do in the context of Machine Learning theory/intuition, along with justification for your experimentation methodology. **DO NOT** just describe the results, for example, you should explain why the learning rate has the observed effect. Also, be cognizant of the best way to organize and show the results that best emphasizes your key observations. If you need more than one slide to answer the question, you are free to create new slides.*

Firstly, in the accuracy plot, the validation accuracy is higher than the training accuracy most of the time, because the training accuracy is the average accuracy of each epoch while the validation accuracy is computed at the end of that epoch after the whole training data has been fed. When learning rate = 1, keeping all other default hyper-parameters fixed, the accuracy plot fluctuates a lot, indicating that the learning rate is too high and it might cause the model to converge too quickly and oscillate around the optimal weights, leading to fluctuations in accuracy. When learning rate = 0.1, 0.05, 0.01, Both the training accuracy and testing accuracy are lower than those with a learning rate of 1, which might indicate that the learning rate is too low and we have achieve the optimal weights. Thus, we should increase the learning rate or the number of epochs. When learning rate = 0.01, as there is a wide gap between the training accuracy curve and the validation accuracy curve, the model doesn't perform well with the validation data. It might indicate the model is still at an early stage in the learning process.

2. Regularization

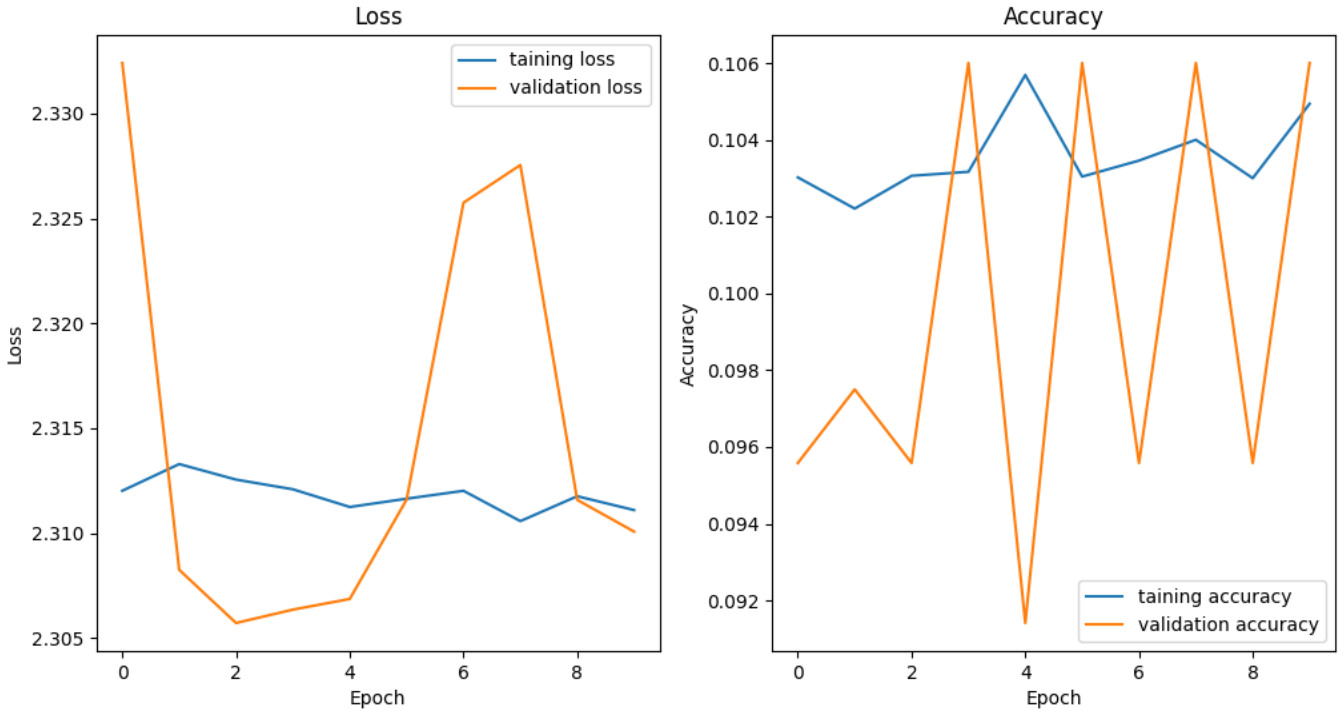
Tune the regularization coefficient of the model with all other default hyper-parameters fixed. Fill in the table below:

	alpha=1	alpha=1e-1	alpha=1e-2	alpha=1e-3	alpha=1e-4
Training Accuracy	0.1032	0.3318	0.8821	0.9219	0.9291
Validation Accuracy	0.1060	0.3997	0.8961	0.9275	0.9347
Test Accuracy	0.1135	0.3954	0.8939	0.9254	0.9338

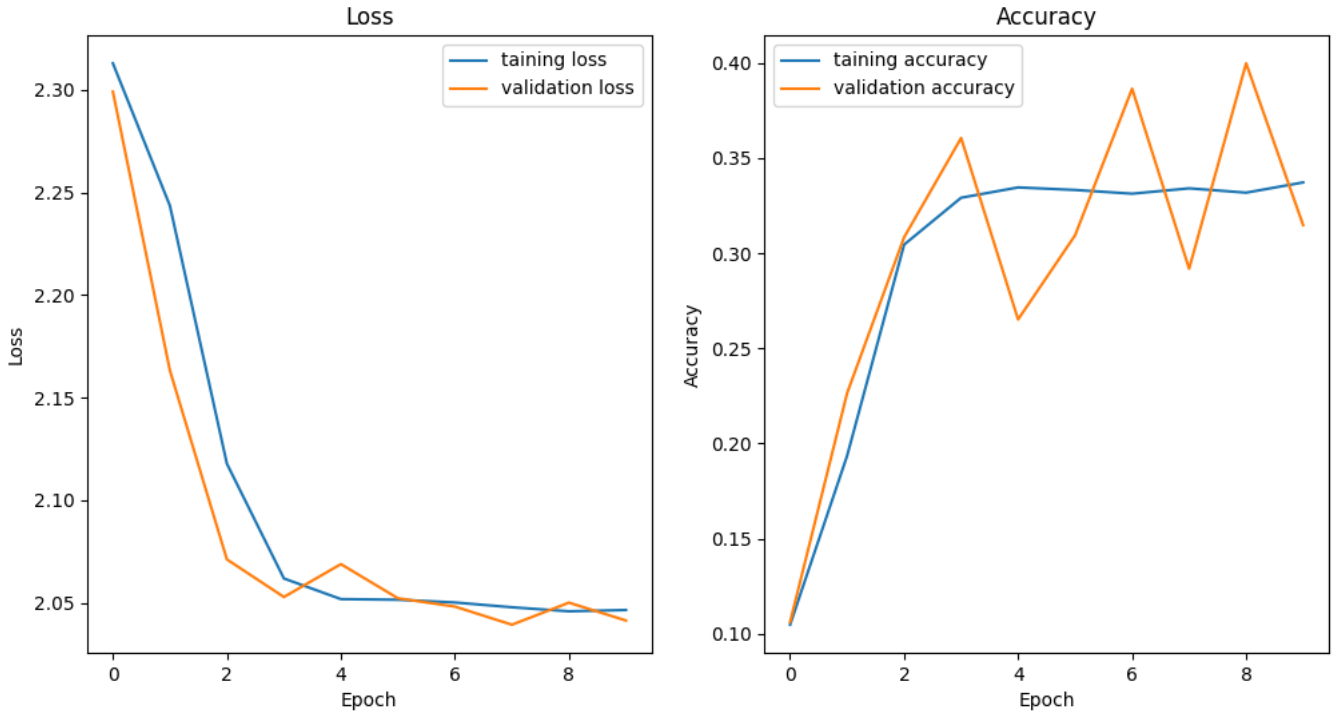
2. Regularization

Plot the learning curves using the regularization coefficients from the previous slide and put them below (you may add additional slides if needed).

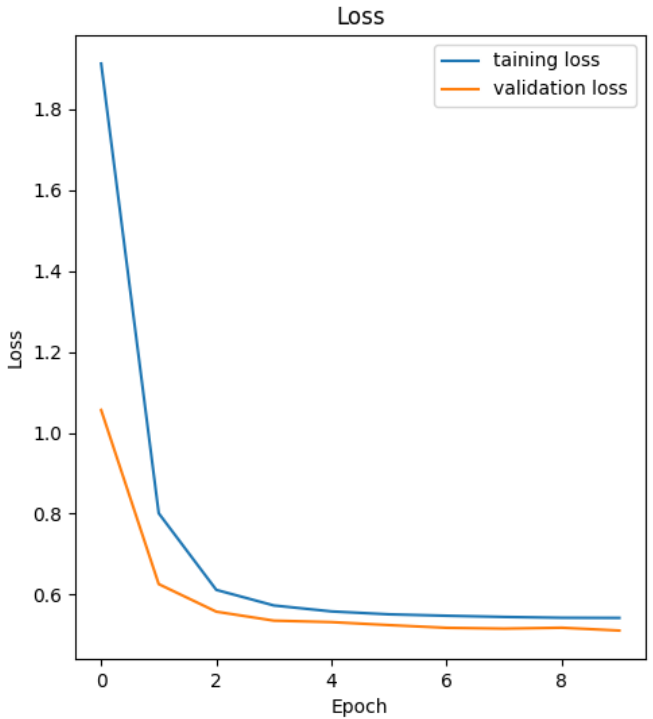
alpha=1

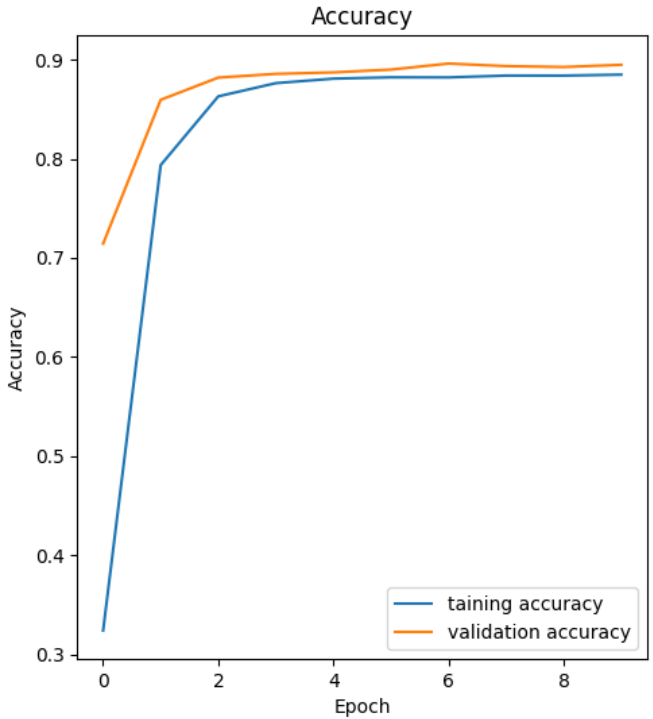

Loss
taining loss
validation loss
2.330
2.325
2.320
2.315
2.310
2.305
Loss
0
2
4
6
8
Epoch
Accuracy
0.106
0.104
0.102
0.100
0.098
0.096
0.094
0.092
Accuracy
taining accuracy
validation accuracy
0
2
4
6
8
Epoch

alpha=1e-1

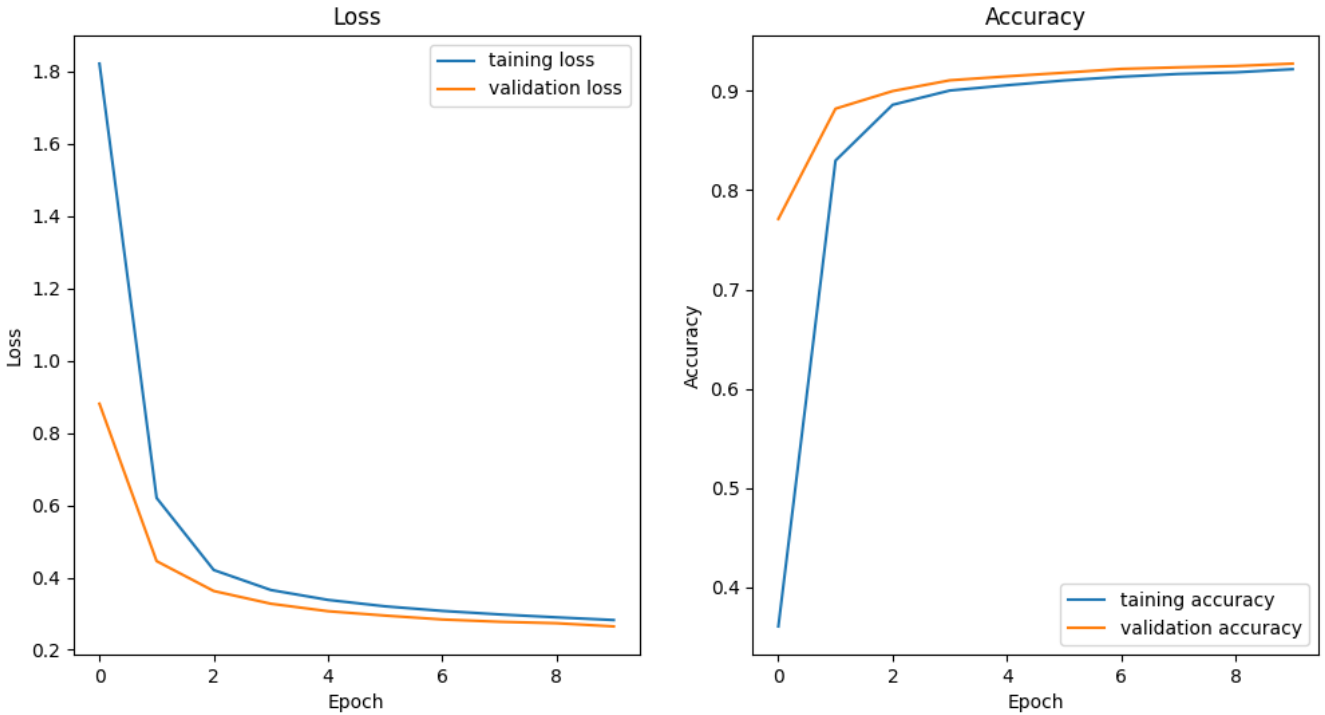

Loss
taining loss
validation loss
Loss
Epoch
2.30
2.25
2.20
2.15
2.10
2.05
0
2
4
6
8
Accuracy
taining accuracy
validation accuracy
Accuracy
Epoch
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0
2
4
6
8

alpha=1e-2

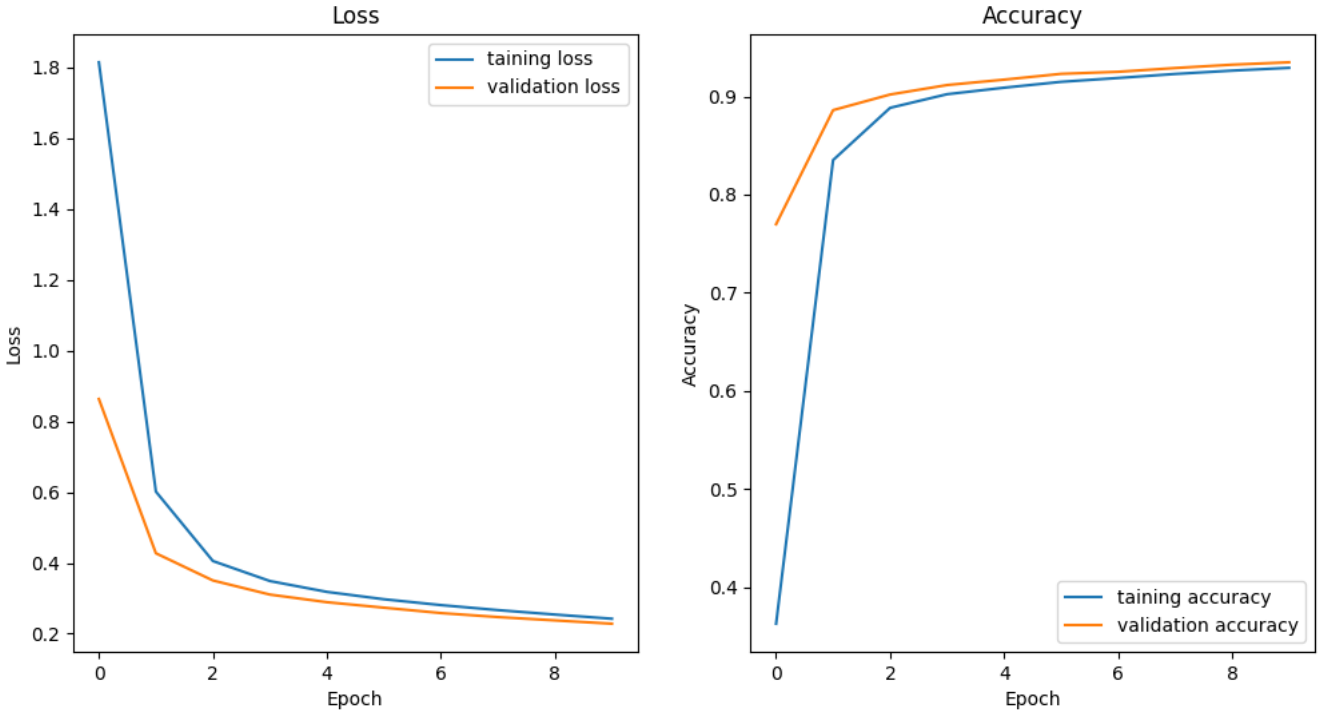

Loss
taining loss
validation loss
Loss
Epoch
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0
2
4
6
8


Accuracy
taining accuracy
validation accuracy
Accuracy
Epoch
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
2
4
6
8

alpha=1e-3


Loss
taining loss
validation loss
Loss
Epoch
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
2
4
6
8
Accuracy
taining accuracy
validation accuracy
Accuracy
Epoch
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
8

alpha=1e-4


Loss
taining loss
validation loss
Loss
Epoch
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
2
4
6
8
Accuracy
taining accuracy
validation accuracy
Accuracy
Epoch
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
8

2. Regularization

Describe and Explain your findings: *Explanation should go into **WHY** things work the way they do in the context of Machine Learning theory/intuition, along with justification for your experimentation methodology. **DO NOT** just describe the results, for example, you should explain why the regularization value affects performance as well as model weights. Also, be mindful of the best way to organize and show the results that best emphasizes your key observations. If you need more than one slide to answer the question, you are free to create new slides.*

When alpha = 1 or 1e-1, the validation accuracy fluctuates a lot, indicating that it may be hard to capture the optimal weights. This is because we penalizes large weights too much and the model may become too simple, underfitting the training data. A low L2 regularization penalty imposes a weak penalty on large weights, which may lead to overfitting, with a high training accuracy and low validation and testing accuracy. As the L2 regularization coefficient decreases (alpha = 1e-2, 1e-3, 1e-4) , the training accuracy, validation accuracy, testing accuracy all increases. Thus, keeping other parameters fixed, the best alpha may be smaller than 1e-3. The optimal alpha could fall into [1e-4, 1e-3) or smaller than 1e-4.

3. Hyper-parameter Tuning

You are now free to tune any hyper-parameters for better accuracy. Create a table below and put the configuration of your best model and accuracy into the table:

Batch Size	Learning Rate	Regularization Coefficient	Epochs	Training Accuracy	Validation Accuracy	Testing Accuracy
32	0.5	0.0002	50	0.9833	0.9769	0.9766

Explain why your choice works: *Explanation should go into **WHY** things work the way they do in the context of Machine Learning theory/intuition, along with justification for your experimentation methodology. **DO NOT** just describe the results, you should explain the reasoning behind your choices and what behavior you expected. Also, be cognizant of the best way to mindful and show the results that best emphasizes your key observations. If you need more than one slide to answer the question, you are free to create new slides.*

Based on the knowledge obtained when tuning the learning rate and the regularization coefficient, we know that the optimal learning rate should fall into the interval [0.1, 1] and the regularization coefficient should be less than 0.001 . Also, Larger batches can lead to better computational efficiency but may generalize less well. Smaller batches may introduce more noise into the optimization process, acting as a form of regularization. Therefore, I set the initial parameters as: learning_rate = 0.1, batch_size =64, reg = 0.0001. When the model overfits, I increases the reg or decreases the batch size. When the validation accuracy fluctuates, I reduces the reg or increases the batch size or decrease the learning rate. Lastly, I select the parameters with the highest testing accuracy, keeping that the model doesn’t overfit.