Deep learning - Homework 1

Stefanela Stevanovic (63220492)

1 Introduction

The goal of this homework is to implement and train a basic neural network on the CIFAR-10 dataset. We will experiment with different network configurations and training strategies to investigate their impacts on classification accuracy. Specifically, we will compare results when training with:

- different number of neurons and hidden layers,
- different optimizers (SGD vs. Adam) and learning rates,
- with and without L2 regularization, and
- with and without applying a learning rate schedule.

2 Results

2.1 Choosing Network Architecture and Epochs

In this section, we conducted a series of experiments to explore how varying network architectures influence the classification accuracy of a basic neural network trained on the CIFAR-10 dataset. The input layer always consists of 3072 neurons, corresponding to the 32x32x3 pixel values of the CIFAR-10 images and the output layer always consists of 10 neurons, each representing one of the CIFAR-10 classes. The architectures specified in the Table 1 denote the arrangement and size of the hidden layers. For example, ”100-100” signifies a network with two hidden layers, each containing 100 neurons. Across these experiments, we maintained a constant learning rate (LR) of 0.01, did not apply L2 regularization, and used the Stochastic Gradient Descent (SGD) optimizer for all runs. Each network was trained for 20 epochs.

Experiment	Architecture	L2 reg.	Optimizer	LR	Epoch	Cls. Acc.
Exp. 1	100-100	No	SGD	0.01	20	44.82%
Exp. 2	200-100	No	SGD	0.01	20	47.71%
Exp. 3	500-100	No	SGD	0.01	20	48.59%
Exp. 4	1000-100	No	SGD	0.01	20	48.80%
Exp. 5	**1200-100**	**No**	**SGD**	**0.01**	**20**	**50.57%**
Exp. 6	300-100-20	No	SGD	0.01	20	46.80%
Exp. 7	300-100-100	No	SGD	0.01	20	45.45%
Exp. 8	1000-100-10	No	SGD	0.01	20	49.69%

Table 1: Experiments with different network architectures.

From Table 1, we can see that starting with simple architectures (e.g., 100-100) and moving towards more complex ones, there’s a general trend of increasing classification accuracy. This suggests that more complex networks, with more neurons and layers, have a higher capacity to learn from the CIFAR-10 dataset. The highest classification accuracy achieved was 50.57% with the 1200-100 architecture. We will chose the 1200-100 network architecture as our best option for balancing high classification accuracy with reasonable training times, despite the potential benefits of trying even more complex networks.

Furthermore, we conducted a series of experiments to investigate the impact of varying the number of training epochs on the classification accuracy of our neural network, while keeping the rest of parameters constant. The results are presented in Table 2.

Experiment	Architecture	L2 reg.	Optimizer	LR	Epoch	Cls. Acc.
Exp. 1	1200-100	No	SGD	0.01	10	48.54%
Exp. 2	1200-100	No	SGD	0.01	15	49.29%
Exp. 3	1200-100	No	SGD	0.01	20	50.79%
Exp. 4	1200-100	No	SGD	0.01	30	50.60%
Exp. 5	1200-100	No	SGD	0.01	40	47.67%

Table 2: Experiments with different number of training epochs.

The experiments with 10, 15 and 20 epochs in table 1 show a trend where increasing the number of epochs leads to improved classification accuracy, while with more than 20 epochs, the classification accuracy begins to decline.

2.2 Choosing Optimizer and Learning rate

In this section, we experimented with different optimizers (SGD and Adam) alongside varying learning rates to understand their impact on model training for the CIFAR-10 dataset. The outcomes of these experiments are presented in Tables 2 and 3, reflecting how each configuration influences classification accuracy.

Experiment	Architecture	L2 reg.	Optimizer	LR	Epoch	Cls. Acc.
Exp. 1	1200-100	No	SGD	0.1	20	10.00%
Exp. 2	1200-100	No	SGD	0.01	20	50.59%
Exp. 3	1200-100	No	SGD	0.001	20	46.87%
Exp. 4	**1200-100**	**No**	**SGD**	**0.001**	**50**	**52.22%**
Exp. 5	1200-100	No	SGD	0.0001	20	35.86%
Exp. 6	1200-100	No	SGD	0.0001	50	40.10%

Table 3: Experiments with SGD optimizer and different learning rates.

With the SGD optimizer, a high learning rate of 0.1 led to a notably low classification accuracy of 10%, suggesting that such a rate is too aggressive, likely causing the optimizer to overshoot the minima. Adjusting the learning rate to 0.01 significantly improved performance, yielding a classification accuracy of 50.59%, indicating this as a more suitable rate for SGD in this context. When decreasing the learning rate further to 0.001 and 0.0001, while maintaining a consistent epoch count of 20, we notice decreasing classification accuracy, showing that out model is underffiting. For these low learning rates of 0.001 and 0.0001 we decided to extend the training to 50 epochs, where we saw significant improvement with a learning rate 0.001 reaching a 52.22% classification accuracy. However, with the for lowest learning rate of 0.0001 even 50 epochs did not adequately compensate for the slow convergence.

Experiment	Architecture	L2 reg.	Optimizer	LR	Epoch	Cls. Acc.
Exp. 1	1200-100	No	Adam	0.1	20	10.00%
Exp. 2	1200-100	No	Adam	0.01	20	10.00%
Exp. 3	1200-100	No	Adam	0.001	20	45.36%
Exp. 4	1200-100	No	Adam	0.001	50	47.42%
Exp. 5	1200-100	No	Adam	0.0001	20	49.86%
Exp. 6	**1200-100**	**No**	**Adam**	**0.0001**	**50**	**53.61%**

Table 4: Experiments with Adam optimizer and different learning rates.

From table 2 we can see that starting with high rates of 0.1 and 0.01, Adam struggled, producing a low accuracy of 10%. With the learning rate of 0.001 Adam still underperformed compared to SGD, achieving lower accuracy for both 20 and 50 training epochs. A learning rate of 0.0001 with Adam and a training duration of 20 epochs achieved a classification accuracy of 49.86%, significantly outperforming SGD under the same conditions. Further extending the training to 50 epochs with this learning rate of 0.0001 allowed Adam to excel, achieving the classification accuracy of 53.61%. These results show Adam's capability to better navigate the optimization landscape at lower learning rates, benefiting from its adaptive learning rate mechanism.

2.3 Incorporating L2 Regularization

The experiments presented in Table 5 incorporate L2 regularization into a neural network under different regularization strengths (Lambda values) and optimizers (SGD and Adam). The inclusion of a lambda value of 0 serves as a baseline, indicating the model's performance without regularization.

Experiment	Architecture	Lambda	Optimizer	LR	Epoch	Cls. Acc.
Exp. 1	1200-100	0.0	SGD	0.001	50	52.22%
Exp. 2	1200-100	0.001	SGD	0.001	50	51.86%
Exp. 3	1200-100	0.01	SGD	0.001	50	51.93%
Exp. 4	1200-100	0.0	Adam	0.0001	50	53.61%
Exp. 5	1200-100	0.001	Adam	0.0001	50	53.32%
Exp. 6	1200-100	0.01	Adam	0.0001	50	53.67%

Table 5: Experiments with different L2 regularization parameter (Lambda) values.

When incorporating regularization with both the SGD and Adam optimizers, no improvement in classification accuracy was observed. In fact, with the SGD optimizer, there was a slight decrease in accuracy. This behavior can possibly be explained by the absence of significant overfitting in the model prior to the introduction of regularization. L2 regularization is most beneficial when a model is overfitting, if overfitting is not present, the introduction of regularization may not lead to improvements and could even slightly degrade performance due to the unnecessary constraint on the model's capacity. However, this decrease in accuracy is minor and could possibly also be attributed to the model's initialization.

2.4 Incorporating Learning Rate Schedule

Table 6 presents the results of experiments designed to evaluate the impact of incorporating an exponential learning rate decay schedule into our neural network architecture, using two different decay rates: 0.01 and 0.1.

Architecture	L2 reg.	Optimizer	LR	Decay Rate	Epoch	Cls. Acc.
1200-100	No	SGD	0.01	No	20	50.59%
1200-100	No	SGD	0.01	0.01	20	50.93%
1200-100	**No**	**SGD**	**0.01**	**0.1**	**20**	**53.16%**

Table 6: Experiments with exponential learning rate decay.

While introducing a decay rate of 0.01 resulted in a slight improvement, a significant increase in accuracy to 53.16% was observed with the introduction of a higher decay rate of 0.1, compared to the baseline accuracy of 50.59%. The experiment with the higher decay rate of 0.1 demonstrates the benefit of allowing the learning rate to decrease at a faster rate, as it seems to provide a better balance between exploration (at the start of training with a higher learning rate) and exploitation (towards the end of training with a lower learning rate).

2.5 Putting it all together and increasing accuracy

After observing significant improvement in classification accuracy with the introduction of a learning rate schedule over just 20 epochs of training, we decided to see if accuracy could be improved by extending the training duration and incorporating L2 regularization to counteract potential overfitting. The results of this experiment, presented in Table 7, show a significant improvement in classification accuracy to 55.14%.

Architecture	Lambda	Optimizer	LR	DR	Epoch	Cls. Acc.
1200-100	0.001	SGD	0.01	0.1	40	55.14%
1200-100	0.001	Adam	0.001	0.1	40	%

Table 7: Combining L2 regularization and learning decay rate.

3 Conclusion

In this homework, we investigated various techniques to optimize a neural network on the CIFAR-10 dataset, focusing on architectural complexity, optimizer choice, learning rate adjustments, and regularization strategies. We discovered that employing a more complex network architecture (specifically, the 1200-100 configuration) and fine-tuning the learning rate provided solid results with both SGD and Adam optimizers. Notably, Adam demonstrated an ability to converge to satisfactory results in less learning epochs than SGD at lower learning rates. Initially, incorporating L2 regularization did not yield significant improvements, suggesting that our neural network was not experiencing overfitting with the given configuration and dataset. Another key finding from our experiments was the significant performance enhancement achieved through the integration of an exponential decay schedule for the learning rate when using the SGD optimizer. This strategy enabled the model to start training with higher learning rates, promoting faster convergence initially, and gradually decrease the learning rate to fine-tune the weights towards the end of training. This allowed us to reach higher classification accuracy in less training epochs compared to just using lower learning rates from the beginning.

In the culmination of our experiments, we combined both L2 regularization and a learning rate schedule with the SGD optimizer, achieving a classification accuracy of 55.14%.