CSE 574 - Programming Assignment 2
Handwritten Digits Classification

Submitted by

1. Sahithya Arveti Nagaraju - 50559752
2. Sushmitha Manjunatha - 50560530

1. Finding optimal hyper parameters for Neural network

For default values mentioned in the code, we got below accuracies and time for training:

```
lambda:0
hidden layers:50

Training set Accuracy:95.724%

Validation set Accuracy:95.32000000000001%


Test set Accuracy:95.1%

Time taken:108.21511125564575
```

The process of selecting optimal hyperparameters for a neural network, specifically the number of hidden units and the regularization parameter (λ). Hyperparameter tuning is crucial for balancing model complexity, training performance, and generalization to unseen data.

Hyperparameters Under Consideration

1. **Number of Hidden Units**:
    - Determines the capacity of the model to learn complex patterns.
    - Values tested: **4, 8, 12, 16, 20**.
2. **Regularization Parameter (λ)**:
    - Controls the magnitude of weights to prevent overfitting.
    - Values tested: **0, 10, 20, 30, 40, 50, 60**.

Methodology

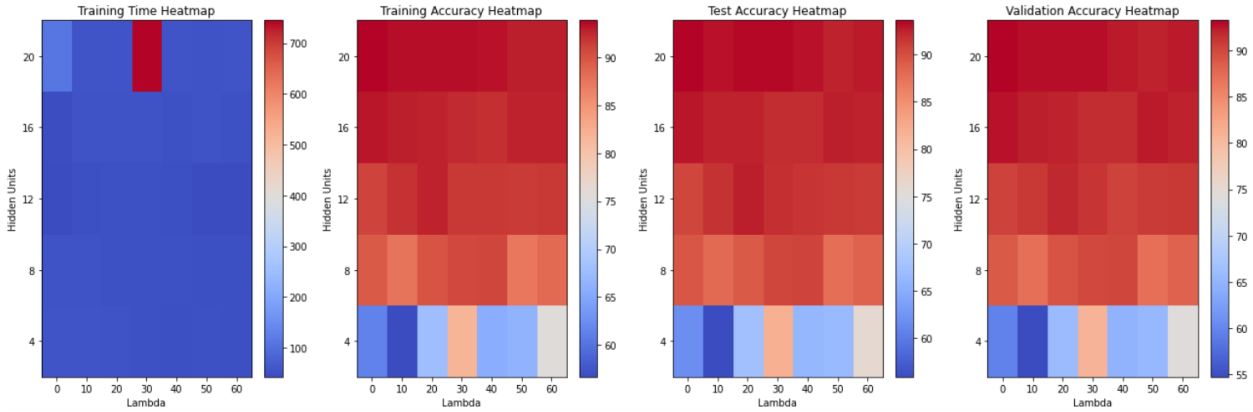
The experiment involves training neural networks with various combinations of hidden units and λ, recording metrics like:

- **Training Time**
- **Training Accuracy**

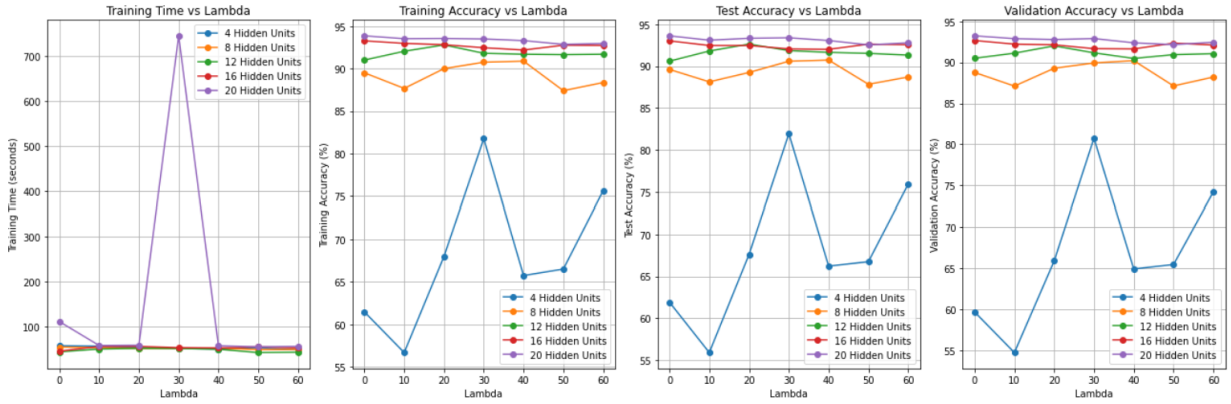
- **Validation Accuracy**
- **Test Accuracy**

Observations from Heatmaps and Line Plots

Heatmaps: Visualize the combined effects of hidden units and λ on training time and accuracy metrics.



Line Plots: Provide detailed trends of accuracy and training time as λ varies for each hidden unit configuration.



1. Training Time:

- **Heatmap**: Shows training time increases as the number of hidden units grows due to increased model complexity.
- **Plot**: As λ increases, regularization adds computational overhead, slightly impacting training time.

2. Training Accuracy:

- **Heatmap**: Highlights higher accuracy with more hidden units and smaller λ, indicating the model's ability to fit the training data.
- **Plot**: High accuracy at λ=0, but a slight drop as λ increases due to regularization constraints.

3. Validation and Test Accuracy:

- **Heatmaps and Plots**:
    - Peak validation accuracy is observed with **16-20 hidden units** and λ values in the range of **10-30**.
    - Test accuracy trends mirror validation accuracy, confirming good generalization.

Observations:

1. **Optimal Hidden Units**:
    - **16 or 20 hidden units** consistently yield the best validation and test accuracies, balancing model complexity and performance.
2. **Optimal Regularization (λ)**:
    - **λ=10** achieves the best trade-off between overfitting and underfitting.
    - Higher λ values (e.g., 40-60) reduce overfitting but slightly degrade performance due to underfitting.

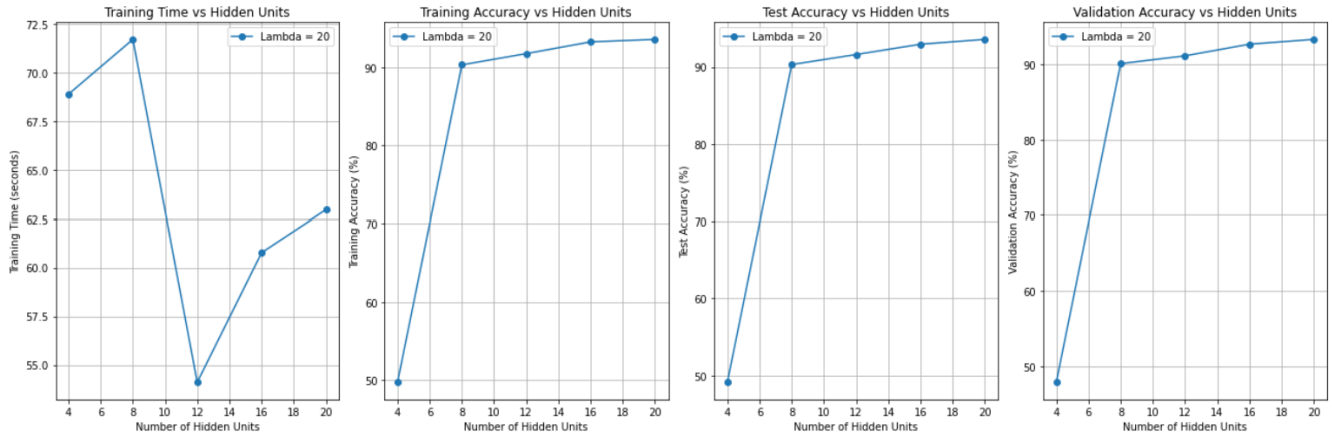
Based on the analysis, the optimal hyperparameter combination is:

- **Hidden Units**: 20
- **λ**: 10

With 20 hidden units and λ=10 parameter values we got below accuracies and time:

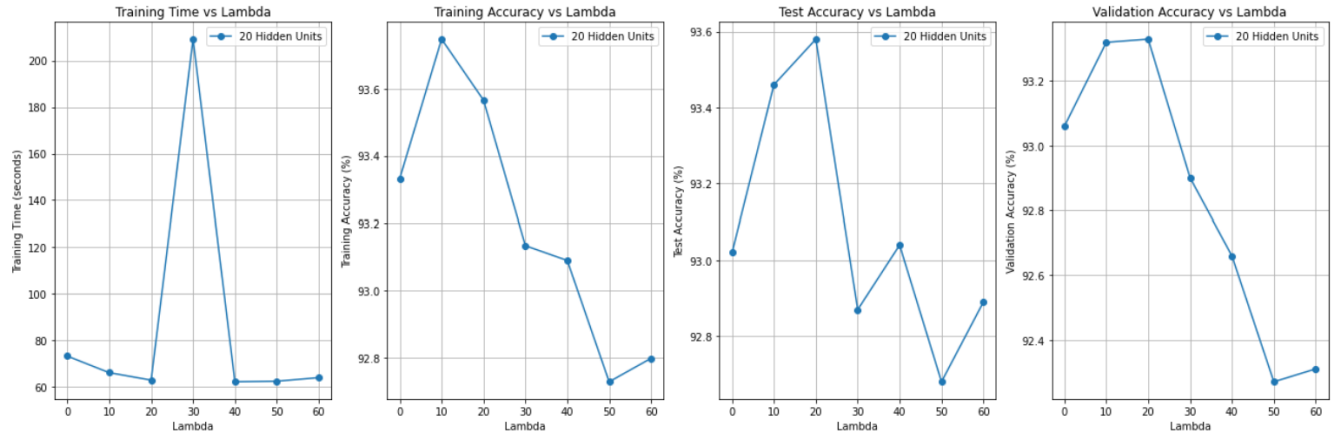
```
Experimenting with 20 hidden units and lambda=10...
Training accuracy:  93.748 %
Testing accuracy:  93.46 %
Validation accuracy:  93.32000000000001 %
Time taken: 66.14591336250305
```

By setting λ = 10, we observe the accuracies and training times for different hidden unit values, as shown below,



As hidden units increase, validation and test accuracies improve and stabilize at 20, indicating strong generalization without overfitting. Training accuracy also peaks at this point, showing the model effectively learns complex patterns. While training time slightly increases, it remains efficient, making 20 hidden units an ideal choice.

By setting hidden units = 20, we observe the accuracies and training times for different λ values, as shown below,



Lambda (λ) = 10 was chosen since it was adequate in balancing accuracy versus generalization. A look at the graphs shows that the value of λ guarantees high training, test, and validation accuracy without encountering overfitting. This makes λ = 10 the best choice for consistent and reliable performances.

In conclusion, the choice of 20 hidden units and λ=10 strikes a good balance between model complexity and generalization. For the case with 20 hidden units, the model approximates the pattern really well without overfitting and without an increased computation cost, while λ=10 prevents overfitting by penalizing large weights, ensuring that the validation and test accuracies stay high and fairly close to the training accuracy. This configuration turns out to be best for efficiency and accuracy among all configurations and provides a robust and generalizable model.

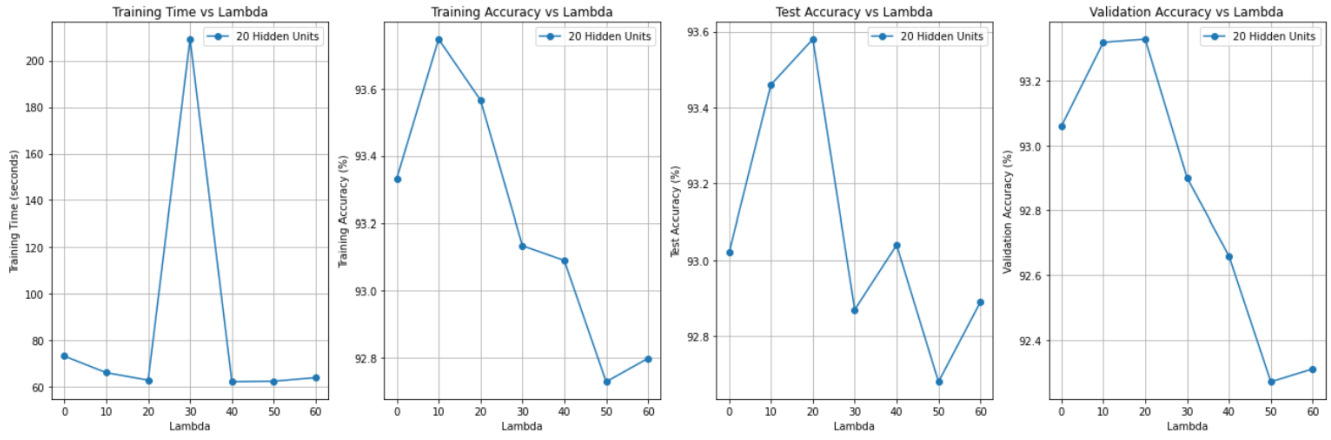
2.**Accuracy of classification method on handwritten digits test data**

For default values mentioned in the code, we got below accuracies and time for training:

```
lambda:0
hidden layers:50

Training set Accuracy:95.724%

Validation set Accuracy:95.32000000000001%


Test set Accuracy:95.1%

Time taken:108.21511125564575
```

For optimal values, with 20 hidden units and λ=10 parameter values we got below accuracies and time:

```
Experimenting with 20 hidden units and lambda=10...
Training accuracy:  93.748 %
Testing accuracy:  93.46 %
Validation accuracy:  93.32000000000001 %
Time taken: 66.14591336250305
```



At λ =20, the model strikes the best balance between generalization and computational efficiency. It avoids underfitting, which happens at higher λ), yet still has high accuracy on both the test and validation datasets. While there is a spike at λ=30, the training time for λ=20 remains relatively efficient compared to other points. Hence, λ=20 ensures robust performance while controlling overfitting and minimizing unnecessary complexity.
Thus, highest accuracy observed on handwritten digits data is 93.32

3. Accuracy of classification method on CelebA data set

Now, the model is tested on the CelebA dataset with a lambda value of 10 and 256 hidden units. When this configuration was used, the testing accuracy on the CelebA data set was 85.42% with a training time of 91.07 seconds. This outcome suggests that the model is adapted to other data sets, but the accuracy is slightly lower as compared to handwritten digits data due to the extra complexities involved in CelebA images. This setup provides a good compromise between classifying accuracy and computational efficiency.

Testing Data Accuracy is 85.42% and Training time is 91.07 sec.

Time to train: 90.47542572021484

Training set Accuracy:85.64928909952607%

Validation set Accuracy:84.80300187617262%

Test set Accuracy:85.42770628311885%

Time on testing data: 91.07971501350403

4. Comparison of Simple Neural Network with Deep Neural Network

We now compare the results of a Simple Neural Network with those of a Deep Neural Network using varying numbers of hidden layers.

Deep Neural Network with 2 layers:	
Accuracy	0.80961394
Time Taken (in seconds)	61.5912
Number of Epochs	50

Deep Neural Network with 2 layers:	
Accuracy	0.81112796
Time Taken (in seconds)	134.726
Number of Epochs	100

Deep Neural Network with 3 layers:	
Accuracy	0.7880394
Time Taken (in seconds)	82.03244
Number of Epochs	50

Deep Neural Network with 3 layers:	
Accuracy	0.7982589
Time Taken (in seconds)	148.933
Number of Epochs	100

Deep Neural Network with 5 layers:	
Accuracy	0.75359577
Time Taken (in seconds)	93.396
Number of Epochs	50

Deep Neural Network with 5 layers:	
Accuracy	0.75662374
Time Taken (in seconds)	176.436
Number of Epochs	100

Deep Neural Network with 7 layers:	
Accuracy	0.7422407
Time Taken (in seconds)	113.3072

Number of Epochs	50

Deep Neural Network with 7 layers:	
Accuracy	0.7513248
Time Taken (in seconds)	202.115
Number of Epochs	100

The highest accuracy achieved with the Deep Neural Network was approximately 81.11%, which is lower than the 85.42% accuracy obtained with the single-layer network.

Comparison table:

Features	**Single layer Neural Network**	**Deep Neural Network**
Number of layers	1	2
Accuracy	85.42%	81.1127%
Time taken (seconds)	91.07	134.726

Observations:
1. Increasing the number of hidden layers leads to longer training times due to the rise in computational requirements and network complexity.
2. Adding more hidden layers can cause overfitting, which results in reduced accuracy.
3. Simpler neural networks achieve higher accuracy and shorter training times compared to deep neural networks with multiple layers.

Conclusion: As shown in the table, the accuracy decreases while the training time increases in the neural network., which suggests that a more complex network does not necessarily yield better performance.

5. Convolutional Neural Network Results

Optimization Iteration	**Training Accuracy**	**Accuracy on Test-Set**	**Time taken for**

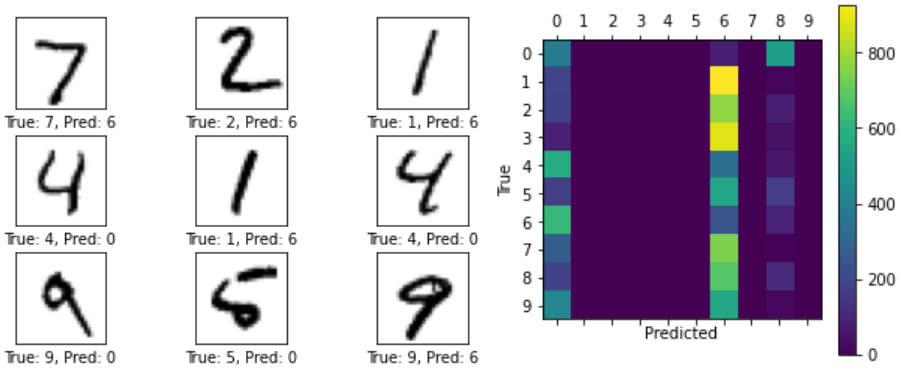
			training (in seconds)
0	-	**7.4%**	**00**
1	**15.6%**	**10.5%**	**00**
100	**57.8%**	**55.4%**	**03**
1000	**87.5%**	**92.4%**	**24**
9901	**100%**	**98.8%**	**226**

The CNN showed major improvements in accuracy and performance with an increase in iterations. It was seen that when the number of iterations was increased to 9901, the model achieved **98.8%** accuracy on the test set; hence, showing that it generalized well to new or unseen data. The training process took a total of **3 minutes and 46 seconds**, which is a reasonable time considering the complexity of training a CNN.

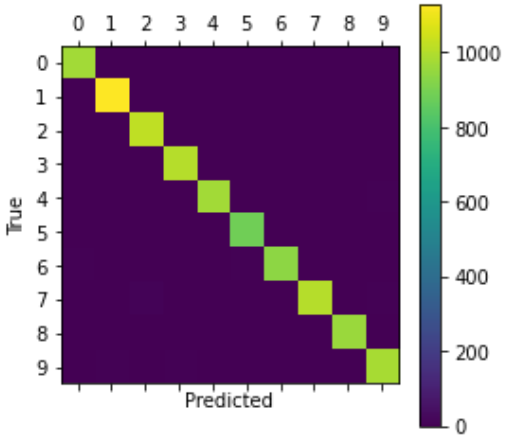
Observations:

1. **Training Time:** The training time for every optimization step varied with each iteration, where the consumed time was increasing gradually as the training went on. The last recorded time for all training is 0:03:46 (3 minutes and 46 seconds).
2. **Training Accuracy:**
    - At the start of the training, the **training accuracy** was relatively low at **15.6%** after the first iteration.
    - As the number of iterations increased, the accuracy improved significantly and reached 100% at 9901 iteration.
3. **Test Accuracy:**
    - **Initial Test Accuracy**: Right at the beginning of training, **7.4%** was the test accuracy with a low error rate on the test set.
    - As the training progressed:
        i. After **1 iteration**, the test accuracy improved to **10.5%**.
        ii. After **1001 iterations**, the test accuracy had risen to **92.4%**.
        iii. Finally, after **9901 iterations**, the test accuracy peaked at **98.8%** (9877 out of 10000 correctly classified digits).
        iv. The **test accuracy** continued to increase as the training progressed, confirming the model's ability to generalize to unseen data.

4. **Confusion Matrix:** The confusion matrix gives more detail about how well the model performed in classifying each digit from 0 to 9. The final confusion matrices at different stages of training are shown below:
    - **Initial Confusion Matrix (Before Training)**: Before training, there is much misclassification among different digits, meaning that the model has not been able to learn the distinction between the digits very well. Example errors and Confusion matrix associated with this are below,



    - **Confusion Matrix after 9901 Iterations**: The model's predictions were much better, as can be seen by the much higher density of correctly classified digits in each category.



The last matrix suggests that the CNN model was able to learn a way of correctly classifying most of the digits, since in this matrix, most of the entries fall into positions corresponding to a correctly predicted digit.

The model correctly classified most of the MNIST digits, with only a few misclassifications throughout the digits, as can be seen from the confusion matrix of the last iteration. In all, the experiment proves that the CNN works extremely well in doing image classification tasks on this dataset, MNIST; it has outstanding accuracy while still maintaining a relatively fast training time.