MINI-PROJECT 2 REPORT

Tianyi Xu
261082272

Ali Kadum
260808977

October 31, 2023

ABSTRACT

This project delves into the foundational aspects of neural networks. We construct a multilayer perceptron from scratch and apply it to image classification on two benchmark datasets: Fashion MNIST and CIFAR-10. Our project covers a wide range of topics in neural network training, including weight initialization, model architecture, activation functions, regularization techniques, and data preprocessing. We investigate how these factors impact the accuracy of both multilayer perceptron (MLP) and Convolutional Neural Network (CNN) models for image classification.

Through our experiments, we make several key observations:

Proper weight initialization significantly influences model performance, with Xavier and Kaiming initialization methods outperforming others. Deeper network architectures and non-linear activation functions, like ReLU, lead to improved accuracy. The choice of activation functions can significantly impact the model's performance, with ReLU performing best among the tested options. Regularization techniques, especially L1 regularization, need to be carefully tuned to prevent over-regularization. Data normalization is crucial for efficient model training, improving both MLP and CNN performance. Custom CNN models, when compared to MLP, may require deeper architectures to outperform on image classification tasks.

Our results provide valuable insights into the design and training of neural networks for image classification.

1 Introduction

In this report, we evaluate the performance of multilayer perceptron (MLP) and Convolutional Neural Network (CNN) models on two image datasets: *Fashion MNIST* and *CIFAR-10.*

Our project encompasses various aspects of model training, including weight initialization, model architecture, activation functions, regularization techniques, and the impact of data preprocessing. Through our experiments, we aim to gain insights into how these factors influence the accuracy of both MLP and CNN models in classifying images.

The results and observations presented in this report provide valuable insights into the design and training of neural networks, offering a comparative analysis between MLP and CNN models for image classification tasks.

The two datasets used are cited in the references. Both datasets have been thoroughly exploited, and numerous works that use the datasets already exist, and can be found directly from the link in the references 1 2.

2 Datasets

We have tested our multilayer perceptron model on two image datasets: *Fashion MNIST* and *CIFAR-10.*

2.1 Dataset 1: FMNIST

The Fashion-MNIST dataset consists of 70,000 greyscale images, where each image is a 28x28 pixel representation of fashion items across 10 categories, namely: T-shirts/tops, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and ankle boots. These categories are evenly distributed across the dataset, with each class presenting 7,000 images. We would like to predict the class of these images, which is a multi-class classification task with 10 distinct classes.

2.2 Dataset 2: CIFAR-10

The CIFAR-10 dataset consists of 60,000 color images, where each image is 32x32 pixels. The dataset is evenly distributed among 10 different classes, which are: airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

2.3 Data Processing

- **Data Preparation and Image Visualization**
  First, data were loaded and basic data exploration on the dataset dimensions was performed. Then, for both datasets, we visualized a subset of images from each class to get a qualitative sense of the variety within categories.

- **Class Distribution Exploration**
  To confirm class balance, we plotted histograms of the class distribution. This is a critical step in making sure we avoid training bias.

- **Data Splitting and Normalization**
  We first split both datasets into training and testing sets before normalizing them as to avoid contaminating the dataset. Afterwards, images are transformed into tensors, which are then normalized so as to speed up training.

3 Results

We evaluated our multilayer perceptron models through accuracy. A higher accuracy score corresponds to a better model performance

We acknowledge that during this mini-project, we referenced Yoshuo Bengio's paper titled "Deep Learning". 3

The set of hyper parameters that were used consistently for the corresponding models across the experiments 1, 2, 3, and 5 are defined in Appendix A. These default parameters were chosen through experimental measures as well as referencing information from 4.

3.1 Experiment 1

In experiment 1, all MLP consists of a single-hidden-layer with 128 hidden units and each having the ReLU activation function.

Table 1: Accuracy Scores for Training and Test Sets with Different Weight Initialization

Weight Initialization Technique	Training Set Accuracy	Test Set Accuracy
Zero Initialization	0.1	0.1
Uniform [-1,1] Initialization	0.817	0.817
Gaussian Initialization	0.778	0.776
Xavier Initialization	0.857	0.842
Kaiming Initialization	0.859	0.844

From the results shown in Table 1, we see that varying initial weights had a significant impact on the performance of single-hidden-layer MLPs. Performing zero initialization on weights proved to be ineffective as it yielded 10% accuracy on both training and test sets. Initialization with random weights using Uniform [-1,1] and Gaussian distributions improved test accuracy to 81.7% and 77.6%, respectively. The highest accuracy were achieved using Xavier and Kaiming weight initialization methods, with 84.2% and 84.4% test accuracy respectively, showing their suitability for the task.

3.2 Experiment 2

Table 2: Accuracy Scores for Training and Test Sets with Different MLP Layer Numbers

Activation Function Type	Number of Layer(s)	Number of Units	Training Set Accuracy	Test Set Accuracy
None	0	0	0.837	0.824
Relu	1	128	0.858	0.841
Relu	2	128	0.871	0.855

From the results shown in Table 2, we observe that there is a clear trend correlating the depth of the model to their classification accuracy. The simplest model, with no hidden layers, served as a baseline with test accuracy of 82.4%. Then, MLP with 1 hidden layer of 128 units and ReLU activations gave a test accuracy of 84.1%. Finally, by further increasing the model complexity, the MLP with 2 hidden layers both with 128 units and ReLU activations resulted in a test accuracy of 85.5%. The positively correlated increase in test set accuracy with model depth and non-linearity shows a consistent improvement in the model's ability to capture the underlying patterns within the data.

3.3 Experiment 3

In experiment 3, all MLPs are composed of 2 hidden layers each with 128 hidden units.

Table 3: Accuracy Scores for Training and Test Sets with Different MLP Activation Function Choice

Activation Function Type	Training Set Accuracy	Test Set Accuracy
Relu	0.871	0.855
Sigmoid	0.76	0.753
TanH	0.867	0.851

From the results shown in Table 3, we see that the model using ReLU activation for both layers achieved a training accuracy of 87.1% and a test accuracy of 85.5%. On the other hand, the model using sigmoid activation for both layers yielded the model achieving only 76% training accuracy and 75.3% test accuracy, which is a significant decrease from the original ReLU model. Lastly, the model using TanH for both layers resulted in comparable performance to the original ReLU model, with a training accuracy of 86.7% and a test accuracy of 85.1%.

3.4 Experiment 4

In experiment 3, all MLPs are composed of 2 hidden layers each with 128 hidden units using the ReLU activation function.

Table 4: Accuracy Scores for Training and Test Sets with Different L1 and L2 Regularization Constants

L1	L2	Training Set Accuracy	Test Set Accuracy
0.00001	0	0.082	0.079
0.000001	0	0.838	0.827
0.0000001	0	0.872	0.852
0	0.00001	0.832	0.819
0	0.000001	0.869	0.85
0	0.0000001	0.872	0.854
0.0000001	0.0000001	0.871	0.857

From table 4, we observe that both training and test set accuracy dropped dramatically to around 8% when given a strong L1 regularization (L1=0.00001). As we decrease L1 regularization strength, however, we notice that the accuracy rebounded with L1=0.0000001 yielding a 82.7% accuracy on the test set. L2 regularization, on the other hand, proved to be less strict. Even the strongest L2 regularization (L2=0.00001) maintained a test accuracy above 81%. As we weaken the L2 regularization strength, we notice an increase in both training and test set accuracy. Interestingly, after combining both regularization in with L1 and L2 both initialized as 0.0000001, the most optimal result of 85.7% in test set accuracy is achieved.

3.5 Experiment 5

All models below implement a double-layer perceptron model with ReLU activation on both and 128 hidden units for each layer.

(a) **Table 5** Accuracy Scores for Training and Test Sets with Normalized Datasets

Training Set Accuracy	Test Set Accuracy
0.871	0.855

(b) **Table 6** Accuracy Scores for Training and Test Sets with Unnormalized Datasets

Training Set Accuracy	Test Set Accuracy
0.847	0.837

From the results shown in Tables 5 and 6, we notice that when trained on normalized images, the MLP achieved a training accuracy of 87.1% and a test accuracy of 85.5%. On the other hand, when trained on unnormalized images, there is a decline in performance, with a training accuracy of 8 84.7% and the test accuracy to 83.7%.

3.6 Experiment 6

In this experiment, we observed that a Convolutional Neural Network (CNN) configuration featuring two convolutional layers and two fully connected layers led to a modest improvement in both training and test set accuracy. After 25 epochs, the training set accuracy reached 89.33%, while the test set accuracy was 87.32%.

Table 6: Model Accuracy on Training and Test Sets

Training Set Accuracy	Test Set Accuracy
0.8933	0.8732

3.7 Experiment 7, 8, and 9

These experiments are combined as they are highly correlated and were conducted in parallel. We employed the same CNN architecture. We initially explored the choice of optimizers between Adam and SGD with momentum. Additionally, we pretrained ResNet-50 while freezing only the convolutional layers and adding two fully connected layers. We tested SGD with various momentums, and the results after 20 epochs are as follows:

Table 7: Model Performance Comparison: Training and Test Set Accuracy

Model	Training Set Accuracy (%)	Test Set Accuracy (%)
SGD Momentum 0.0	47.21	45.67
SGD Momentum 0.1	51.34	48.96
SGD Momentum 0.2	52.36	48.93
SGD Momentum 0.3	54.47	50.18
SGD Momentum 0.4	59.88	54.53
SGD Momentum 0.5	62.21	54.52
SGD Momentum 0.6	67.14	56.95
SGD Momentum 0.7	71.25	58.15
SGD Momentum 0.8	70.33	57.34
SGD Momentum 0.9	69.83	60.14
SGD Momentum 1.0	10.00	10.00
Adam	67.56	61.76
MLP	86.90	85.10
ResNet-50	99.16	85.06

4 Discussion and Conclusion

- **Experiment 1**
  A key takeaway from this experiment is the critical role proper weight initialization plays in the training process of neural networks. The significant difference in the final model accuracy between zero-initialization with the rest of the weight initialization methods shows that having a diverse weight at the beginning is crucial for model learning. The better results achieved by Uniform and Gaussian initialization show the potential of random weight distribution to kickstart learning. Lastly, the fact that Xavier, tailored for Sigmoid activation functions, and Kaiming, tailored to ReLU activation functions by solving the "dying ReLU" problem, resulted in the best test accuracy align with theoretical expectations as these methods account for the size of the previous layer of neurons.
- **Experiment 2**
  This experiment highlights two fundamental aspects of a neural network's design: the importance of deeper network architecture and the importance of non-linearity within a network. By introducing the ReLU activation function, we introduce non-linearity, which enables the model to learn more complex functions beyond simple linear patterns. By increasing the layer numbers, we allow the network to construct a more abstract representation at each level and capture hierarchical features that are crucial for classification tasks.
- **Experiment 3**
  From this experiment, we observed the effect of the choice of activation functions on a neural network's performance. The Sigmoid model's poor performance is likely the result of Sigmoid's susceptibility to the vanishing gradient problem. The TanH model, although performs slightly worse than the ReLU model, did not suffer as much as the Sigmoid model. This is most likely due to the fact that TanH centers the data as it is scaled between -1 and 1. This then helps with the gradient flow during back-propagation.
- **Experiment 4**
  From this experiment, we noticed that L1 regularization has a critical effect on our model performance, more so than L2 regularization. Having an overly aggressive L1 regularization can drastically impair model learning. This is likely due to an excessive penalty imposed on the weights, which then leads to our model under-fitting the data. As the L1 penalty decreases, however, we notice that our model performance recovers. This then hints towards the fact that there exists a sweet spot where regularization can improve generalization. On the other hand, L2 regularization is much less aggressive and performs only slightly worse as the strength of

lambda2 increases. This could be due to its nature of penalizing weights less severely, thereby preserving the ability to learn.

The improved performance with combined L1 and L2 regularization at the lowest regularization strengths implies a complementary effect, where each counterbalances potential over-fitting in different aspects of the weight distribution.

- **Experiment 5**
  With this experiment, we found that normalization of inputs is a critical data prepossessing step as it leads to improved model training and generalization. This result is most likely due to the fact that neural networks are sensitive to the scale and distribution of input data. Without normalizing the dataset, gradients can become unstable and this makes it harder for our MLP to learn efficiently as it is dependent on gradient descent methods to optimize. Moreover, as our model implements the ReLU activation function, which is unbounded, large input values can potentially lead to exploding gradients which then makes training even more difficult.
- **Experiment 6**
  We anticipated that this experiment would yield slightly better results, as Convolution Neural Networks (CNNs) take into account the neighboring elements of image pixels. We hypothesize that further increasing the number of convolution layers could lead to even higher accuracy.
- **Experiments 7, 8, and 9**
  It is surprising to note that after 20 epochs, the Multi-Layer Perceptron (MLP) outperformed the Custom CNN models. This result is unexpected since CNNs are generally expected to perform better on image data. ResNet-50 achieved an impressive accuracy of 99.16%, which is notable. Further investigation revealed that having only two convolution layers might not be sufficient for training on the CIFAR-10 dataset. ResNet's success can be attributed to its deeper architecture with 16 convolution layers. Additionally, our CNN model may not have performed as well on FMNIST compared to CIFAR due to the color differences (3 channels vs. 1 channel), making training on higher channels potentially more challenging.

Future research could investigate the threshold for which adding more depth to the MLP model would stop yielding improvements and beginning to give back lower accuracy scores. Another direction for future research could be to explore a wider array of activation functions, such as Leaky ReLU or parametric ReLU, to determine if they can overcome the minor shortcomings of TanH and potentially outperform ReLU. Additionally, exploring the combination of these different activation functions and their potential synergies with different MLP structures can provide deeper insights into different network architectures.

5 Statement of Contributions

Each member contributed equally to this project. Each of us completed this lab individually and wrote the report by merging our responses and code.

References

[1] Krizhevsky, A. (2009) Learning Multiple Layers of Features from Tiny Images. Technical Report TR-2009, University of Toronto, Toronto.

[2] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. ArXiv, abs/1708.07747.

[3] Bengio, Y. (2016). Deep Learning. MIT Press.

[4] Silipo, R. (2020, February 11). Machine learning algorithms and the art of hyperparameter selection. Medium. https://towardsdatascience.com/machine-learning-algorithms-and-the-art-of-hyperparameter-selection-279d3b04c281

Appendix A Default Model Parameters

Model	L1 Regularization Term	L2 Regularization Term	Max iteration	Batch Size	Learning Rate
Multilayer Perceptron	1e-07	1e-07	25	16	0.001