Comp 551 Mini-Project 2 Report

Ningping Wang (260969794)
Dept. of Computer Science
McGill University
Montreal, Canada
ningping.wang@mail.mcgill.ca

Simon Li (260984998)
Dept. of Electrical Engineering
McGill University
Montreal, Canada
xi.yang.li@mail.mcgill.ca

Xue Qi Ao (260891387)
Dept. of Computer Science
McGill University
Montreal, Canada
xue.qi.ao@mail.mcgill.ca

CONTENTS

I Abstract 1

II Introduction 1

III Datasets 1

IV Results 2
- IV-A Experiment 3.1 2
- IV-B Experiment 3.2 2
- IV-C Experiment 3.3 2
- IV-D Experiment 3.4 3
- IV-E Experiment 3.5 3
- IV-F Experiment 3.6 3
- IV-G Experiment 3.7 - Additional Analysis . 4

V Discussion and Conclusion 4

VI Statement of Contributions 4

I. ABSTRACT

The main aim of this project was to develop a multilayer perceptron (MLP) from scratch for image classification and gain a deeper understanding of neural networks and their training algorithms. To achieve this, we created a basic feed-forward neural network and tested various configurations, including model architecture, data augmentation, activation functions, learning rate, and other hyperparameters that have an impact on MLP performance.

Furthermore, we explored the utilization of pre-trained models and convolutional neural networks (CNNs) for image classification by training these models on image datasets and using their accuracies and confusion matrices to evaluate their performance. Our research discovered that the accuracy score of the model increased as we added more layers, and the use of different activation functions led to slight variations in an accuracy improvement. Overall, the results of this project indicate that CNNs outperform MLPs in image classification tasks.

II. INTRODUCTION

This project aims to develop a multilayer perceptron (MLP) from scratch for image classification and present the results of several experiments in which we explored the impact of various factors on the accuracy of MLP models for image classification. Specifically, we investigated the effects of different architectures, hyperparameters, activation functions, and regularization techniques on the performance of MLP models. Our experiments were conducted on the CIFAR10 dataset, a widely used benchmark for image classification. Regarding optimization algorithms for the MLPs, our findings suggested that adding more layers (i.e., increasing the depth) to the network can improve accuracy. Moreover, results reveal that L1 regularization was more effective than L2 and no regularization in preventing overfitting and improving accuracy and that using unnormalized images significantly negatively impacted MLP model accuracy due to weight initialization and numerical instability.

Furthermore, the mini-project investigates the effectiveness of convolutional neural networks (CNNs) and pre-trained models in image classification compared to MLPs. Results show that CNNs and pre-trained models significantly outperform MLPs for image classification. Finally, the accuracy for each class is examined after training the CNN, indicating a comparatively lower performance in recognizing animals compared to other objects.

III. DATASETS

The dataset contains 60,000 32x32 RGB images in 10 classes, with 6,000 images per class. The classes are equally distributed, meaning each class has an equal number of images. This ensures that the model is not biased toward one particular class.

As a means of normalization, the pixel values of each image in the dataset were divided by 255, which rescales the pixel values to a range of 0 to 1. This step is necessary to ensure that all features are on the same scale, which helps the model learn more effectively.

Additionally, the dataset was augmented by applying various transformations, such as horizontal flip, Gaussian blurring, rotation, perspective transform, and adding noise. These transformations create new training examples that are slightly different from the original images, which helps the model learn to recognize objects from different perspectives, angles, and lighting conditions. This technique is beneficial when there is limited training data available, as it helps to create more diverse and representative training data.

The augmented and original images were visualized by plotting one instance from four randomly chosen classes. This helped to understand the differences between the original and augmented images.

Finally, the accuracy for each class was examined after training a CNN, with dogs, cats, and deer having the lowest accuracy, and cars, planes, and trucks having the highest accuracy. This finding suggests that the model has a harder time recognizing animals compared to other objects in the dataset.

Overall, this whole mini-project will provide a good understanding of how neural networks learn feature extractors and classifiers simultaneously and how data augmentation can be used to improve the model's performance. By using these techniques mentioned previously, we can build more accurate and robust machine learning models that can be applied to a wide range of computer vision tasks.

IV. Results

A. Experiment 3.1

In this experiment, we created and trained three different MLP models for image classification, each with different architectures. The hyperparameters that are used are the following: learning rate = 0.001, and batch size equals 64.

The first model was a simple linear classifier with no hidden layers. This model mapped the inputs directly to the outputs and did not incorporate non-linearity or complex interactions between features. As a result, it was not very effective at classifying images and had a relatively low accuracy score.

The second model was implemented with one hidden layer with 256 units and ReLU activation by introducing non-linearity into the network and allowing the network to capture more complex relationships and patterns between features. Adding a hidden layer made the model deeper and could better capture more abstract and nuanced image features. As a result, the accuracy of this model was significantly higher than the first model.

The third model had two hidden layers with 256 units and ReLU activation. Increasing the number of layers could capture even more detailed features, potentially improving accuracy. However, despite having a similar accuracy score to the second model with only one hidden layer, a neural network with two hidden layers can potentially learn more complex and intricate features. The limited number of epochs may have hindered the benefits of additional hidden layers, given that the training set size was adequate.

This experiment demonstrated the importance of non-linearity and network depth in improving the accuracy of the MLP models for image classification. By carefully tuning the architecture and hyperparameters of the network, we can more effectively capture the relevant features in the images and classify them accurately.

B. Experiment 3.2

In this experiment, we compared the test accuracies of the three models, all of which were copies of the same MLP architecture with two hidden layers but with different activation functions: Leaky ReLU, Tanh and Relu. The results showed that the model with ReLU activation performed slightly better than Leaky ReLU, which in turn performed better than the Tanh model. This suggested that the choice of activation function in our experiment can slightly have an impact on the model performance.

Tanh, similar to sigmoid, can model both positive and negative numbers. However, unlike ReLU, Tanh is almost linear and can suffer from the vanishing gradient problem, making it challenging to train deep neural networks. In contrast, ReLU eliminates everything below zero and is more computationally efficient. Leaky Relu, on the other hand, is similar to ReLU but allows for a small gradient for negative inputs.

In conclusion, we have discovered that the ReLU activation function is better for our task. There could be several plausible reasons why the ReLU model outperformed the Leaky ReLU model in our experiment. One possibility is that the training set was large enough, so perhaps the number of epochs was insufficient, and more training would reveal the benefits of Leaky ReLU. Another hypothesis is that the dying neuron problem, encountered in ReLU where a large gradient updates weights such that its neuron ceases activation forever, was not encountered in the training of the CIFAR10 dataset.

C. Experiment 3.3

In this experiment, we investigated the effects of L1 and L2 regularization on the accuracy of an MLP with a ReLU activation function and two hidden layers, each having 256 units. Our experiment revealed that L1 regularization outperformed no regularization and L2 regularization.

Our initial hypothesis was that L1 regularization would be better suited for image classification for feature selection. We confirmed this hypothesis based on an experiment on our dataset, which showed that many irrelevant pixel variables through correlation matrix analysis, such as background or noise, did not contribute to the classification task.

L1 regularization encourages the model to set some weights to precisely zero, effectively removing corresponding features from consideration. This leads to a simple and more interpretable model, which improves performance by reducing the impact of irrelevant pixels and increasing focus on the most important features of the images. Therefore, reducing the dimensionality of the model by removing useless and irrelevant features through feature selection techniques can be beneficial in image classification tasks.

Although L2 regularization reduces the coefficients toward zero without reaching it, it performs less well than L1 regularization for classifying images. However, L2 regularization tends to spread the weight values more evenly across all features, which may not be as effective in image classification as in other machine learning tasks.

Moreover, L2 regularization is similar to PCA in that it reduces the dimensionality of the model by giving less importance to redundant features. Therefore, the performance of L2 regularization depends on the correlation of the features.

CIFAR10 dataset may have a low correlation, therefore, needs more of a feature selection, which may explain why L2 regularization did not perform as well as L1 regularization in this experiment: L1 regularization: 36% vs L2 regularization: 33.97%.

D. Experiment 3.4

In this experiment, we have trained the MLP with unnormalized images and discovered that unnormalized images would significantly affect the accuracy of the MLP due to several reasons. One possible issue is that the weights initialization expects normalized data. If the input data is not normalized, it may cause overflow with larger inputs, leading to numerical instability and no convergence and causing error in the calculation of the softmax function, which affect the final accuracy score.

Moreover, unnormalized data may have saturated some ReLU units in the negative region, which can result in not updating weights during training. Its derivative becomes very small, making it difficult for the network to learn, thus leading to the very low accuracy of the model.

To sum up, the model's accuracy was significantly reduced when trained on unnormalized images, possibly due to overflow from large input numbers and the saturation of some ReLU units in the negative region. Therefore, normalizing input data before training neural networks is generally advisable. Normalization aids in scaling the data to a similar range, simplifying the learning process for the network to identify appropriate features.

E. Experiment 3.5

In this experiment, we created a convolutional neural network (CNN) with two convolutional and two fully connected layers, using ReLU activations and 256 units in the fully connected layers, and trained it on the CIFAR-10 dataset.

Compared to using MLPs, the CNN model performed significantly better. This improvement can be attributed to four key ideas behind convolutional layers, which take advantage of the properties of natural signals: local connections, shared weights, pooling and the use of numerous layers.

As an additional analysis, we found that kernel size is a crucial factor affecting CNN performance. Smaller kernel sizes increase receptive field, resulting in slightly better accuracy. Our results show that 3x3 and 4x4 kernels achieve 63% accuracy, while 5x5 kernels achieve 61% accuracy. This finding aligns with previous studies, such as the improved performance of AlexNet with 3x3 kernels [1]. Smaller kernel sizes allow each neuron to analyze a larger portion of the input image, leading to a more comprehensive understanding, fewer parameters, and better localization of features, capturing more fine-grained details in the input data. This also reduces the risk of overfitting the model to the training data.

Furthermore, we found that using more kernels per layer can enhance the model's ability to learn diverse and complex features, which is critical for achieving high accuracy, which is 67% in accuracy. By using more kernels, the network can extract more information from the input image and learn more layers of abstraction, and also help the network to learn hierarchical representations of the input, thus achieving slightly higher accuracy.

We also found that including padding in the CNN model can improve accuracy, resulting in a 66% accuracy score. To maintain and expand the spatial dimensions of the output feature maps, padding is particularly important when the input image, in our case, is relatively small, as it can prevent the loss of crucial information throughout the network.

In conclusion, our findings demonstrate that CNNs are highly effective in image classification tasks. Their superior performance can be attributed to convolutional and pooling layers, which enable the network to identify relevant features in the image data more effectively than MLPs. In addition to the results, our findings also emphasize the importance of carefully selecting kernel size, number of kernels per layer, and padding in designing CNN architectures for different tasks.

F. Experiment 3.6

In this experiment, we compared the performance of a pre-trained ResNet model to an MLP with two hidden layers and a CNN with two convolutional and two fully connected layers. The pre-trained ResNet model outperformed both models, with an accuracy score approximately 60% higher than the MLP and a 20% higher accuracy than the CNN.

For the MLP, the difference in performance can be attributed to its limited capacity to capture complex features compared to the ResNet's deeper architecture with skip connections. The MLP with two hidden layers may not be capable of capturing complex features, especially if the images are high-dimensional. This may limit its ability to learn useful representations for the given task. Additionally, pre-training the ResNet model on a large dataset allowed for better weight initialization and regularization, and have a more complex architecture which may have contributed to its superior performance, also requires more training time as proved in our study.

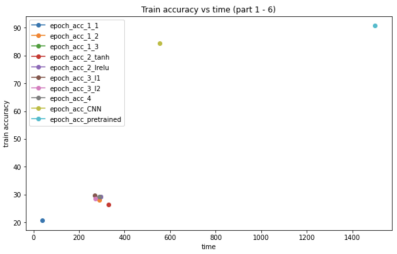

Fig. 1. required training time for different models

Similarly, CNN's performance was hindered by its need to learn features from scratch during training. In contrast, the pre-trained ResNet model had already learned relevant features from its training on ImageNet. The ResNet's depth and regularized training may have also contributed to its higher accuracy, which is 90.35% in accuracy score.

When comparing two different pre-trained models, RestNet18 and RestNet34, with the similar architecture, we did

not observe a significant difference in accuracy. This suggests that the task may not be complex enough to require a more sophisticated model.

By adding an extra hidden layer to ResNet, we have noticed an increase in model accuracy. This improvement can be linked to multiple factors such as enhanced parameter capacity, the ability to combine features in nonlinear ways to form complex feature representations, and better generalization due to the learning of intricate decision boundaries that lead to improved class separation.

G. Experiment 3.7 - Additional Analysis

Based on an analysis of previous Multilayer Perceptron (MLP) models, it has been observed that increasing the model's width results in a steady increase in accuracy scores. This is because it enhances the model's ability to learn complex patterns in the data and reduces the risk of underfitting. However, as the width is increased, accuracy scores may decrease due to reduced generalization and increased risk of overfitting. While we did not observe this in our experiment since we did not train on a larger number of widths, we anticipate a decrease in accuracy as the width increases.

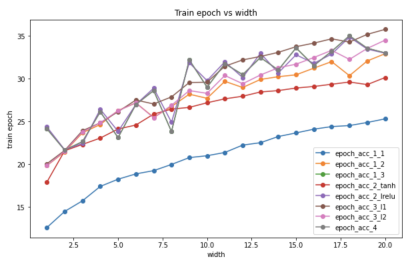

Fig. 2. Effect of the increase in width on MLP models

Data Augmentation: after analyzing the impact of data augmentation on an MLP with two hidden layers, each containing 256 units with ReLU activations and L1 regularization, we observed a lower accuracy score compared to the MLP with L1 regularization but no data augmentation. There could be various reasons for this result. For instance, we only used the horizontal flip technique, which might not have generated a diverse enough set of augmented data points, thus the model may not gain any advantage from the augmented data.

Analysis of different learning rates: we have experimented on a multilayer perceptron (MLP) with a single layer containing 256 units and ReLU activations. We noticed a consistent increase in the model's accuracy. However, we also observed a decline in accuracy as the learning rate grew larger. This pattern is in line with our expectations, as an initial increase in the learning rate can result in faster convergence and improved accuracy. Nevertheless, as the learning rate surpasses a certain threshold, the optimization algorithm may overshoot the optimal solution, causing it to fail to converge or even diverge.

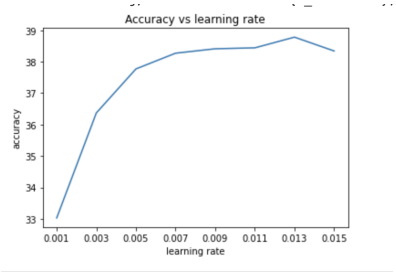

Fig. 3. Effect of the learning rate on MLP model with single layer

V. Discussion and Conclusion

To summarize, our work involved developing MLP, CNN, and pre-trained models for image classification tasks with various modifications, including changes to hidden layers, activation functions, learning rates, regularization techniques, and batch sizes. We evaluated these models on training and test datasets and made several interesting findings. Firstly, increased hidden layers resulted in higher accuracy scores, highlighting the importance of non-linearity and network depth in improving MLP performance. Secondly, while ReLU performed best, we expect Leaky ReLU to outperform it, as we did not encounter the dying neuron problem in the CIFAR10 dataset. Thirdly, L1 regularization outperformed L2 regularization, especially when dealing with irrelevant pixel variables and noise. Fourthly, unnormalized images can significantly impact accuracy by causing numerical instability when calculating the softmax function. Our work also demonstrated the effectiveness of CNNs in image classification, with smaller kernel sizes and more kernels per layer leading to better performance. Additionally, we compared the performance of our models with a pre-trained ResNet model, which outperformed all other models due to its depth and regularization capabilities. Finally, we analyzed the effects of increasing the width of MLP models, using data augmentation and learning rate on the models, finding that our expectations were confirmed. For further experiments, we would consider experimenting with additional pre-trained models to compare their accuracy with the top-performing MLP in part 1 and the standard CNN in part 5. This would enable us to select the most suitable model for comparison. Furthermore, we would train the models with a broader and varied range of data augmentation techniques to investigate potential improvements in their accuracy scores.

VI. Statement of Contributions

Simon Li contributed to writing the code's main bugs, as well as conceptualizing and drafting absurd explanations for the bugs. Catherine Ao was responsible for fixing all the bugs and executing all the experiments in a proper manner. Ningping Wang produced the report and ensured that Simon was kept away from the code to keep it comprehensive, complete and, most importantly, bug-free.

REFERENCES

1) K. Alex, S. Ilya and E. Hinton Geoffrey, "ImageNet Classification with Deep Convolutional Neural Networks," University of Toronto, June 2017, pp84-90.