Classification of Image Data with Multilayer Perceptron and Convolutional Neural Networks

Kara Best, Niki Mahmoodzadeh, Parastoo Gol Mohammadi

ABSTRACT

This report details and compares the performance of the multilayer perceptron model (MLP) and convolutional neural networks (CNN) for the classification of image data. The effects of the number of hidden layers, different activation functions, and different learning rates on the accuracy of MLP models were explored. It was generally observed that the MLP models performed worse than the CNN model on image data.

1. INTRODUCTION

In this project, we implemented three MLP models with backpropagation and mini-batch stochastic gradient descent to classify the CIFAR-10 image dataset. We investigated the accuracy of the MLP models with normalized and unnormalized data, different numbers of hidden layers (zero, one, and two layers), various activation functions (ReLu, tanh, and leaky-ReLu), and other hyperparameters. Furthermore, we compared the performance of the MLP models with a CNN model with two convolutional and two fully connected layers created with the TensorFlow library. We have made several observations. Firstly, the model with two hidden layers has the highest test accuracy among the MLP models. Secondly, for models with the same number of hidden layers, those using the Relu activation function resulted in the best test accuracy. Finally, the CNN model has better training and testing accuracy than the MLP models.

2. DATASET – CIFAR-10

The data set used for this project is CIFAR-10. It is a subset of a more extensive set of 80 million tiny images. For this project, we are using a 10-classifier dataset, which is split into 50,000 training images and 10,000 test images with five training batches and one test batch. The training images are random within each batch but are of the same class, and the test set contains images from all classes. This dataset contains images with a total of 3072 features, and each image is a 32 × 32-pixel patch with three color channels: red, green, and blue (RGB). Each feature, represented as x(i), takes a value between 0 and 255. In order to normalize the data, each of these features was divided by 255 to bring it to a 0 to 1 interval. Additionally, for the MLP model, each image was flattened, resulting in a 3072-dimensional vector instead of a 32 × 32 × 3 matrix. These pre-processing techniques were applied to both the train and test datasets.

3. RESULTS

3.1 Multilayer Perceptron

3.1.1 Network Depth and Nonlinearity

In this experiment, we aim to investigate the effect of the number of hidden layers on the accuracy of an MLP model. To do so, we first created an MLP with no hidden layers (MLP1), a second model (MLP2) with a single hidden layer with 256 hidden units implementing Relu activation, and a third model (MLP3) with two hidden layers, both with 256 units and Relu activation. To evaluate the effect of model depth on the accuracy, the parameters used in all three models were a mini-batch size of 500, a number of epochs equal to 50 and 20, and a learning rate of 0.01. First, they were each trained using the training set of 50,000 samples, then the accuracies were found for the test set predictions. The results obtained are shown in the following table:

		MLP1	MLP2	MLP3
Test Accuracy	50 epochs	39.62 %	50.34%	65.862 %
	20 epochs	38.25 %	50.83 %	51.39 %

Table 1: MLP models' test accuracy

The accuracy of MLP3 is significantly higher than that of MLP1 and MLP2 at 50 epochs; since the hidden layers increase the models' capacity to capture complex patterns in the data due to its increased number of parameters. At 20 epochs, the accuracy of MLP3 is similar to the accuracy of MLP2; however, the accuracies of MLP2 and MLP1 seem to reach their

maximum at 20 epochs. The asymptotic behavior of the accuracies of MLP1 and MLP2 lead to the conclusion that MLP3 is the best-fitting model of the three for the CIFAR-10 dataset.
Below is the plot of the train performance of the MLP3 model as a function of training epochs, which shows the consistent reduction of the SoftMax Cross-Entropy loss as the model goes through more training. No overfitting is observed for this amount of training epochs, but training the model for more epochs can cause overfitting to the training data and consequently compromise the test accuracy.

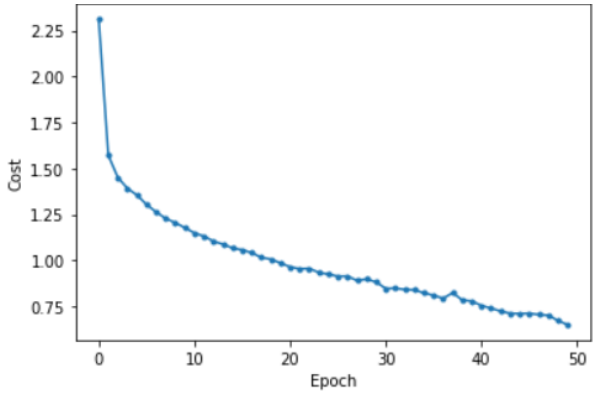

Fig. 1: Training Curve of MLP3 model, cost vs epoch of training

3.1.2 Tanh and Leaky Relu Activation

Continuing from the previous experiment, we created two other MLP models with two hidden layers and 256 units each. The first implements tanh activation on both layers, while the second implements leaky Relu activation. We aim to analyze how the different activation functions affect the accuracy of the model.

Before we collected test accuracy results, we optimized several parameters on each of the models using a validation set. The training set was divided into a training set of 40,000 samples and a validation set of 10,000 samples. First, we evaluated the leaky Relu validation accuracy as a function of the leaky Relu parameter, gamma, for a mini-batch size of 500, epoch number of 50, and learning rate of 0.01, which yielded the curve below. The validation accuracy is highest for gamma = 0.001.

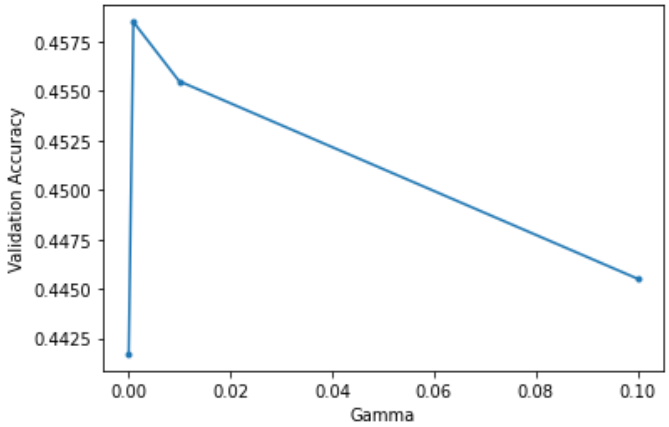

Fig. 2: Validation accuracy vs. gamma – Leaky Relu activation

We then evaluated the validation accuracy as a function of batch size, and then the learning rate for all three models using the same training and validation sets and an epoch number of 10. For the batch size analysis, a learning rate of 0.01 was used. By observing the results of varying batch sizes, we select 40 for learning rate evaluation to minimize running time but also maximize accuracy. The best learning rate for all three activation functions is 0.01, which we will use for all the following experiments.

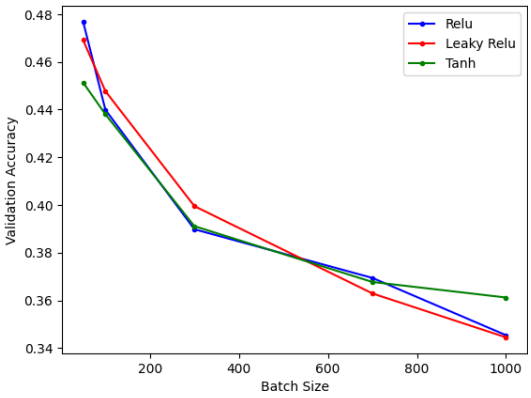

Fig. 3: Validation accuracy vs. batch size

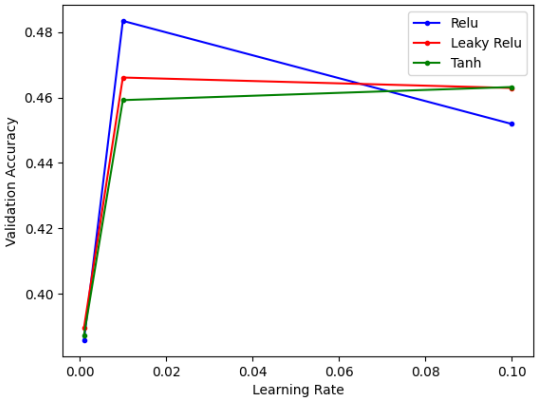

Fig. 4: Validation accuracy vs. Learning rate

We trained each model on the CIFAR-10 dataset and evaluated the test accuracies for 50 epochs. Comparing the results of the two shown below, and that of the previous model with two hidden layers and Relu activation, we can conclude that the Relu activation function results in the best fit.

	Tanh	**Leaky Relu**
Test Accuracy (50 epochs)	51.7%	51.81 %

Table 2: Activation functions' effect on test accuracy

3.1.3 L1 and L2 Regularization

Following the previous experiments, we further optimized the model using regularization. To do so, we took the 2-layer MLP model with Relu activation and applied varying L2 and L1 regularization strengths to them. The model had a minibatch size of 40, and an epoch number of 20 and was trained using the training set with 50,000 samples. The test accuracy was then plotted as a function of the L1 and L2 regularization strength, shown in figures 5 and 6, respectively. Observing the results, we note that for all L1 regularization strengths, the accuracies are higher than the accuracy of the model with no regularization. L2 regularization does not follow the same trend however and seems to decrease the test accuracy of the model. It is possible that the optimal regularization strength was not among the values we tried, or that L2 regularization is just not a good fit for this model.

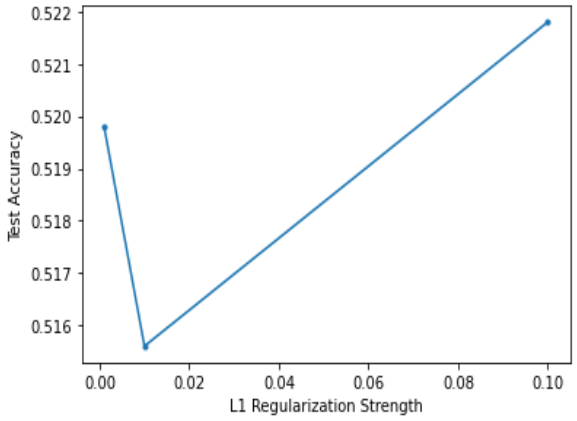

Fig 5: Test accuracy vs. L1 reg. strength

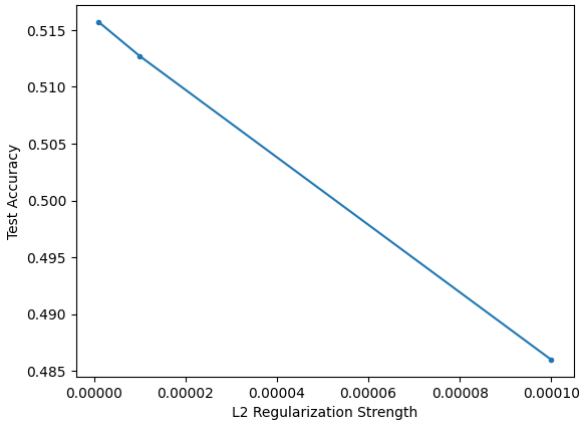

Fig 6: Test accuracy vs. L2 reg. strength

3.1.4 Unnormalized Training Data

The accuracy of MLP3 reduces to 10%; when using unnormalized data. When large unnormalized pixel values are fed to the model, the SoftMax activation function can easily produce large values as well, which can cause numerical instability and result in inaccurate predictions. This is because the exponential function in the SoftMax function grows very quickly for large values, leading to overflow errors.

3.2 Convolutional Neural Networks

3.2.1 Implementing CNN with TensorFlow

For this experiment, a CNN model with 2 convolutional and 2 fully connected layers, using ReLu activation in all layers, to classify images from the CIFAR-10 dataset was implemented. The model achieves a test set accuracy of 88.21% while using 50 epochs. The use of CNNs resulted in better performance than MLPs on image classification tasks due to their ability to capture spatial features in the data. The evolution of various metrics, including accuracy, during training and validation is plotted in this section.

A sudden drop in accuracy between epoch 10 and 20, followed by a recovery, could be due to a variety of reasons, such as:

- Overfitting: The model may be overfitting to the training data. The sudden drop in accuracy may indicate that the model has learned to fit the training data too closely, leading to poor performance on new data. However, the model may then recover as it learns to generalize better to new data.
- Learning rate schedule: The learning rate schedule may have been set in such a way that the learning rate decreases suddenly after epoch 10. This sudden decrease in learning rate may cause the model to take longer to converge, leading to a temporary decrease in accuracy. However, as the model adapts to the new learning rate, it may recover and continue to improve.
- Data quality: The CIFAR-10 dataset may have some inconsistencies or quality issues that make it difficult for the model to learn the features correctly. The sudden drop in accuracy may indicate that the model has encountered a difficult part of the dataset, but then recovered as it learned to adapt to the challenges.

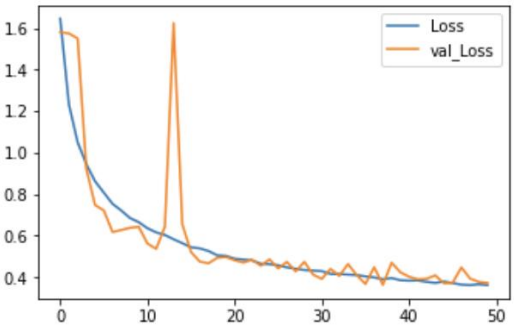

Fig. 7: Loss vs. number of epochs for CNN

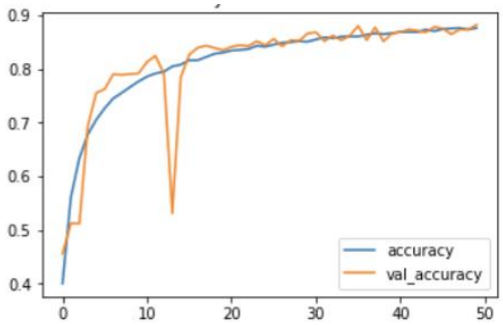

Fig. 8: Accuracy vs. number of epochs for CNN

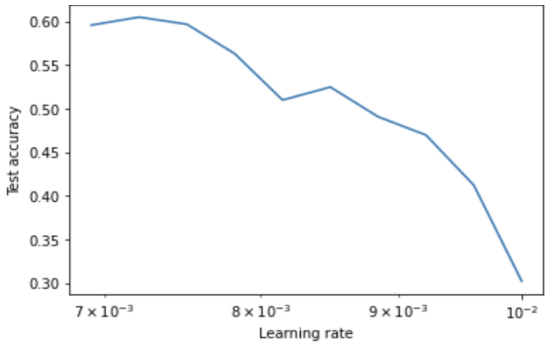

Fig. 9: learning rate vs test accuracy for CNN

As anticipated, CNNs are typically more effective when working with large image datasets, owing to their ability to better capture pixel correlation through the use of convolutional layers.

3.2.2 Loading a pre-trained model

In the last experiment, a pre-trained DenseNet model was loaded with frozen convolutional layers and no fully connected ones. Two fully connected layers were added after the convolutional layers, and only these newly added layers were trained on the CIFAR-10 dataset. The model was trained for 10 epochs on the CIFAR-10 dataset.

During the training, the model achieved an accuracy of 80.27% on the training set. However, on the test set, the model achieved an accuracy of only 63.27%, indicating that the model may be overfitting to the training set. Despite the lower accuracy compared to the CNN model, the pre-trained DenseNet model with frozen convolutional layers and added fully

connected layers can still be a useful tool for image classification tasks. It can also be fine-tuned on specific image datasets to achieve higher accuracy.

The plot of test accuracy vs. learning rate shows that the model achieved the highest accuracy of 63.27% when the learning rate was set to 0.0001. When the learning rate was too high (0.1), the model converged too slowly, resulting in lower accuracy.
The results suggest that the choice of learning rate is an important hyperparameter that can affect the performance of the model. Fine-tuning the learning rate to find an optimal value can improve the accuracy of the model on the test set.

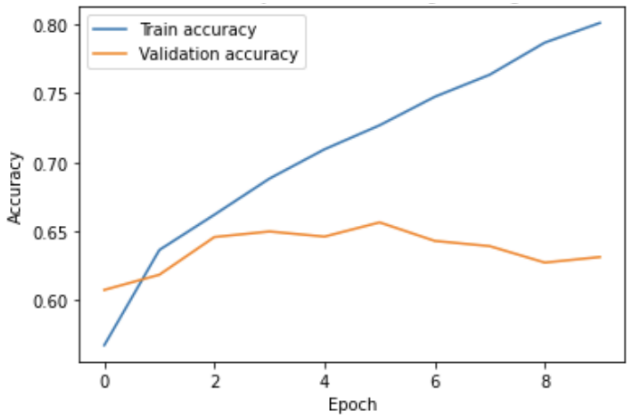

Fig 10: Accuracy vs. number of epochs for DenseNet

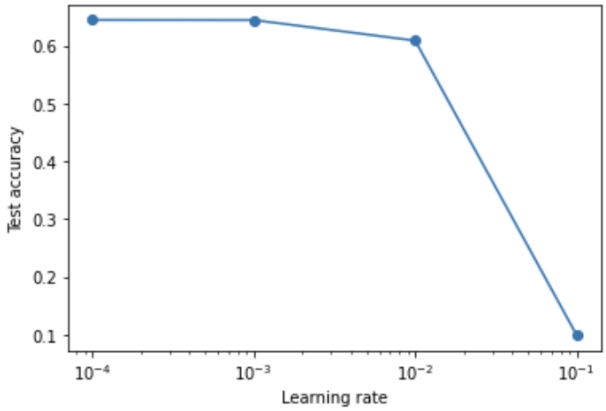

Fig. 11: learning rate vs test accuracy for DenseNet

4. DISCUSSION & CONCLUSION

The following conclusions were drawn from the results of our experiments on image classification. First, we found that training an MLP model with unnormalized data reduces the accuracy of the model significantly; since large pixel values introduce numerical instability to the model, and the SoftMax activation function is especially susceptible to this instability. To address this issue, regularization techniques such as L1 and L2 can be used to encourage the model to produce smaller weights, which can lead to more stable predictions. Additionally, using less training data or reducing the number of hidden units in the model can also reduce this instability. However, these solutions can come at the cost of reduced model capacity and decreased accuracy. Therefore, a balance between stability and model performance should be sought.
Second, we found that adding nonlinearity to the MLP models increases its accuracy because it allows the model to capture more complex relationships between the input features and the output labels. This nonlinearity is introduced through activation functions such as ReLu, which allow the model to learn more intricate representations of the input data.
Third, we learned that the hyperparameters of both MLPs and CNNs are critical for achieving high accuracy. For example, the choice of activation functions, learning rates, and regularization techniques can all have a significant impact on the performance of the models. Therefore, it is crucial to perform practical hyperparameter tuning to obtain optimal results.
Finally, we found that CNNs are generally more suitable for image classification tasks involving large datasets than MLPs. This is because CNNs are designed to capture correlations between pixels and take into account the structural information of the image. In contrast, MLP models were not designed for this purpose and do not do particularly well with image data.

References

1. “CIFAR-10 and CIFAR-100 Datasets.” Www.cs.toronto.edu, www.cs.toronto.edu/~kriz/cifar.html.

2. “CS231n Convolutional Neural Networks for Visual Recognition.” Cs231n.github.io, cs231n.github.io/neural-networks-2/#datapre.

3. Verma, Suraj. “Softmax Regression in Python: Multi-Class Classification.” Medium, 2 June 2021, towardsdatascience.com/softmax-regression-in-python-multi-class-classification-3cb560d90cb2. Accessed 5 Mar. 2023.