COMP 551
Assignment 3

Jessie Kurtz, Marc-Antoine Nadeau, Baicheng Peng

2024-11-18

Abstract

In this assignment, three neural network architectures for classifying the OrganAMNIST dataset were implemented; Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNNs), and the pre-trained MobileNetV2 model using TensorFlow. Tuning hyperparameters such as learning rate, batch size, and regularization strength showed that normalized inputs paired with Tanh activation yielded the best MLP performance, achieving an accuracy of 0.7037. Modified CNNs incorporating MaxPooling and larger fully connected layers outperformed basic CNNs, achieving a maximum accuracy of 0.8325. MobileNetV2, optimized with two fully connected layers, achieved the highest accuracy of 0.9249, balancing precision and computational efficiency. These findings demonstrate the effectiveness of MobileNetV2 for complex image classification tasks and highlight the importance of architecture design and hyperparameter optimization in improving performance.

1 Introduction

Derived from abdominal CT scans, OrganAMNIST provides a structured dataset for multi-class organ classification. This paper investigates the effectiveness of different architectures, including Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNNs), and the pre-trained MobileNetV2, in addressing this classification problem.

MLPs, while foundational in machine learning, often struggle to capture the spatial dependencies present in image data. CNNs address this limitation by leveraging convolutional layers to extract hierarchical spatial features. However, selecting optimal configurations, such as the number of layers, kernel sizes, and pooling strategies, is crucial for achieving high accuracy. Pre-trained models, like MobileNetV2, offer an alternative by transferring knowledge from large-scale datasets, providing improved accuracy with reduced training time.

This dataset is particularly relevant for assessing advanced architectures like Bayesian Deep Neural Networks (BDNNs), which not only make predictions but also quantify uncertainty—an essential capability in healthcare decision-making. A study published by the "Frontiers in Medical Technoology" highlights the robustness of BDNNs compared to deterministic models, particularly their ability to maintain better accuracy and provide reliable confidence measures under challenging conditions like noise and adversarial attacks [1]. This paper demonstrates that while achieving high accuracy is an important benchmark, it is insufficient in isolation, meaning robustness, confidence calibration, and failure detection should be included in our models to achieve better results.

2 Dataset

The Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN) were trained on the OrganAMNIST dataset, which contains medical images classified into 11 organ types. We used and analyzed the performance of the models using the 28 by 28 pixel dataset and the 128 by 128 pixel dataset.

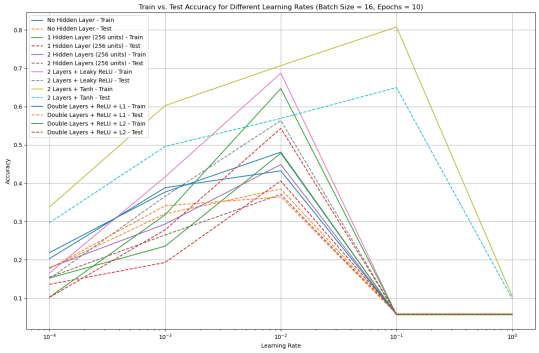
The normalization process adjusted every input pixel values using the Z-score technique [2].The images were then flattened from their original 2D form into 1D vectors, making them suitable input features for the models. The labels for each image were also one-hot encoded, transforming them into binary vectors where each class was represented by a 1 in the corresponding position and 0 in all others.

'Xavier' Initialization was used for layers with tanh activations to keep activations stable within the function's range (e.g., [-1, 1]). It scales weights based on both input and output neurons, avoiding saturation and ensuring gradients flow effectively during training [3].

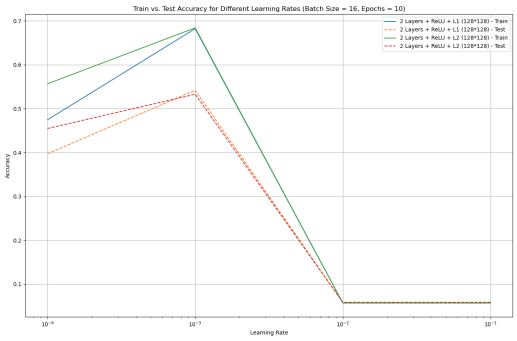
'He' Initialization was applied for layers with ReLU and sigmoid activations. For ReLU, it compensates for inactive neurons (those outputting zero) by scaling weights based only on input neurons, helping maintain gradient stability [3]. For sigmoid, an **additional activation tested on the 28 by 28 image**, He was chosen to initializing weights to avoid small gradients [3].

3 Result

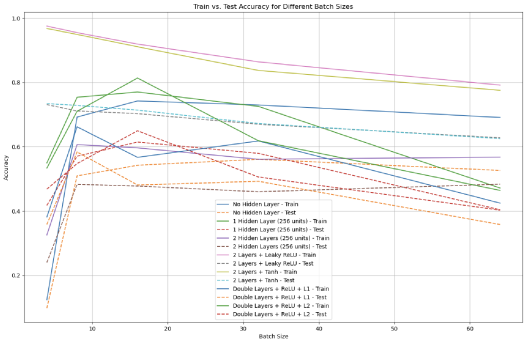
3.1 Multilayer Perceptron



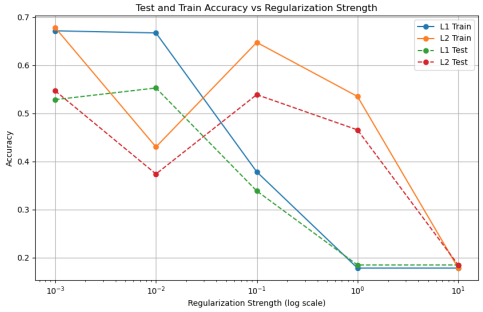
(a) Learning Rate Optimization - 28×28



(b) Learning Rate Optimization - 128×128



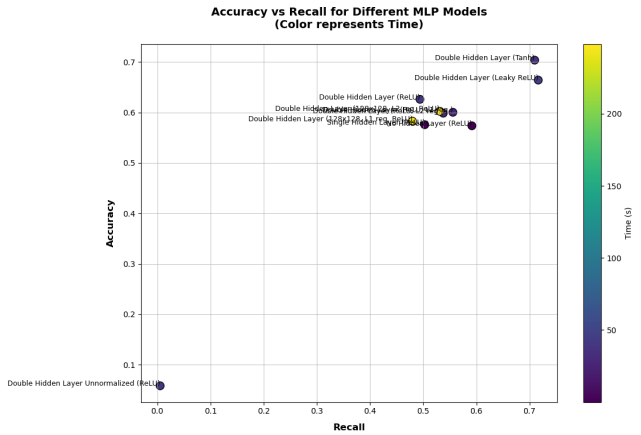
(c) Batch Size Optimization - 28×28 and 128×128



(d) Regularization Strength Optimization - 28×28

Figure 1: Hyper-Parameter Tuning Across Different Activation Functions and Resolution Sizes

Figure 1a demonstrates an ideal[1] learning rate of 10^{-2} for all 28×28 resolution models, except tanh, which favors 10^{-1}. For the 128×128 models shown in Figure 1b, 10^{-1} was chosen. A subset of models, shown in 1c, indicates that accuracy peaks for batch sizes between 8 and 16. Batch size 16 was selected over 8 to minimize computation time. Regularization optimization, shown in 1b for the 28×28 double-layered ReLU activation models, shows an ideal Lasso strength (L1) of 10^{-1}, and Ridge strength (L2) of 10^{-2}. 10 epochs were used for generating the graphs in Figure 1. Figures 1d and 1c used the learning rates found in Figures 1a and 1b.



(a) Graph: Accuracy, Recall, Computation Time

Size	Configuration	Accuracy	Recall	Time (s)
28×28	Double Hidden Layers (Tanh)	0.7037	0.7097	29.69
	Double H. Layers (Leaky ReLU)	0.6642	0.7164	22.37
	Double Hidden Layers (ReLU)	0.5761	0.5940	21.91
	Single Hidden Layer (ReLU)	0.5758	0.5029	15.08
	No Hidden Layer (ReLU)	0.5736	0.5916	2.10
	Double Layers (L2 Reg., ReLU)	0.5987	0.5378	27.94
	Double Layers (L1 Reg., ReLU)	0.6004	0.5559	25.27
	Unnormalized Images	0.0583	0.0053	39.25
128×128	Double Layers (L1 Reg., ReLU)	0.5643	0.4781	245.78
	Double Layers (L2 Reg., ReLU)	0.6025	0.5311	248.51

(b) Table: Accuracy, Recall, and Computation Time

Figure 2: Accuracy, Recall, and Computation Time for Various MLP Configurations

3.1.1 Comparison of Hidden Layer Configurations for ReLU (None, Single, Double)

The results gathered in Figure 2 shows that increasing the network depth and the non-linearity of the data has a minimal impact on accuracy for the OrganAMNIST dataset.

[1] 'Ideal' is defined as having the highest test accuracy.

The model with two hidden layers achieved the highest accuracy (0.5761), slightly outperforming the single hidden layer model (0.5758) and the no-hidden layer model (0.5736). However, the improvement is marginal, indicating that the dataset may not require deeper models for optimal performance. Interestingly, the no hidden layer (0.5916) and double hidden layer (0.5940) models outperform the single hidden layer model in recall (0.5029).

Additionally, training time increases with model complexity, with the double hidden layer configuration taking the longest (21.91 seconds) and the smallest taking (2.10 seconds). These results suggest a trade-off between model complexity and training efficiency, with minimal gains in both accuracy and recall from additional layers. This is not the behavior expected, as typically, deeper networks should show more significant improvements in both accuracy and recall.

3.1.2 Comparison of Activation Function (ReLU, Leaky-ReLU, Tanh)

The results gathered in Figure 2 shows that the model with Tanh activation performs the best, achieving the highest accuracy (0.7037) and recall (0.7097). One can hypothesize that Tanh pairs well with normalized inputs because the activation function's range aligns with zero-centered data. This can result in smoother optimization and better convergence compared to ReLU, which isn't inherently designed for zero-centered inputs.

Leaky ReLU outperforms ReLU in recall (0.7164 vs. 0.5940) and accuracy (0.6642 vs. 0.5761). This improvement can be attributed to Leaky ReLU's ability to fix the zero-gradient problem by allowing small negative values. Namely, it ensures that the network learns weights that would otherwise be zero with ReLU.

While ReLU is simple, it shows the weakest performance. Overall, these results suggest that Tanh is the most effective activation function for this particular task, and Leaky-ReLU strikes a balance between training time and performance.

3.1.3 Comparison of Regularization Technique (L1 reg., L2 reg.)

For the 28×28 resolution models, L1 regularization slightly outperformed L2 in accuracy (0.6004 vs. 0.5987) and recall (0.5559 vs. 0.5378) while training faster (25.27s vs. 27.94s). L1's sparsity-inducing approach proved more effective for this task, offering a better balance of performance and efficiency.

3.1.4 Effect of Normalization

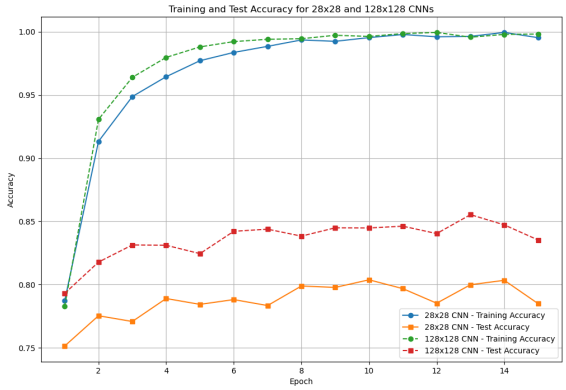
Training the MLP with unnormalized images results in a significant drop in performance. With Z-score normalized images, the model achieves an accuracy of 0.5761 and recall of 0.5940, with a training time of 21.91 seconds. However, when using unnormalized images, accuracy falls to 0.0583, recall drops to 0.0053, and training time increases to 39.25 seconds.

This clearly shows that normalization helps the network converge more efficiently, improving accuracy and reducing training time. Unnormalized images lead to poor learning, higher computational costs, and worse overall performance.

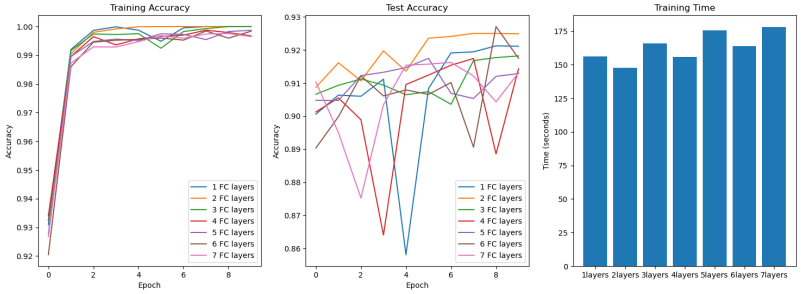
3.1.5 Impact of Input Size with Regularization (L1 reg., L2 reg.)

The 128×128 models outperform the 28×28 models under both L1 and L2 regularization, demonstrating the benefits of higher-resolution inputs. However, this comes at the cost of significantly longer training times, making the choice between resolutions application-dependent, with 128×128 better suited for tasks prioritizing precision. That is to say, the model achieves minimal improvement when compared to the training time.

3.2 Convolutional Neural Network and MobileNetV2



(a) Training and test accuracy trends for CNNs.



(b) Comparing MobileNet with various fully connect layers

Figure 3: Performance comparision of MLP, CNN, and MobileNetV2

Model	Accuracy	Training Time(s)
MLP (Tanh 2 layers)	0.7037	29.69
CNN-1	0.7918	16.03
CNN-2 (Modified)	0.8325	27.40
MobileNet (2 Dense layers)	0.9249	147.72

Table 1: Accuracy and training time comparison for MLP, CNN, and MobileNetV2

3.2.1 CNN and the modification

Using the TensorFlow library, we implemented a simple convolutional neural network (CNN) with two Conv2D layers, one 256-unit fully connected layer (ReLU activation), and an 11-unit output layer (Softmax activation) for classification. We also created a modified CNN by adding a MaxPooling layer between the Conv2D layers to reduce spatial dimensions while retaining information. Additionally, a larger fully connected layer was added to enable learning of more complex patterns from the larger input.

As shown in Table 1, after training both CNNs separately on the normalized OrganAMNIST dataset and its 128×128 versions, the simple CNN achieved an accuracy of 0.7918 with a training time of 16.03 seconds, while the modified CNN achieved an improved accuracy of 0.8325 with a training time of 27.40 seconds. This represents a significant improvement over the multilayer perceptron, which achieved a best accuracy of 0.7037 with a training time of approximately 30 seconds.

3.2.2 Pre-trained MobileNetV2

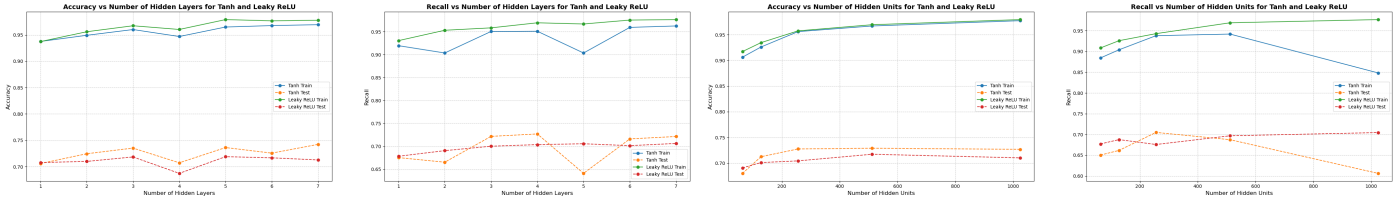
We selected MobileNetV2 for its lightweight architecture and faster training compared to larger models like ResNet50. After freezing the convolutional layers, We stacked trainable fully connected layers and the final output layer. To optimize its performance, we tested configurations with varying numbers of fully connected (FC) layers, as shown in Figure 3b.

The 2-FC layer model provided the best balance, with stable test accuracy across epochs and short training time, avoiding the fluctuations observed with more layers (e.g., 5–7) due to increased parameter complexity.

As detailed in Table 1, the 2-FC layer setup achieved a test accuracy of 0.9249 in 147.72 seconds, significantly outperforming previous MLP and CNN models with minimal extra computational cost, making it the optimal choice for our application.

3.3 Extra Experimentation

3.3.1 MLP Architecture: Width, Depth, and Activation Functions



(a) Accuracy vs. Network Depth (b) Recall vs. Network Depth (c) Accuracy vs. Network Width (d) Recall vs. Network Width

Figure 4: Impact of Multilayer Perceptron Depth and Width on Model Performance

Furthermore, we also experimented with network depth and width and observed their influence on the model's performance (i.e., recall and accuracy). In Figures 4a and 4b, we observe that increasing the depth of the network produces a slight improvement in both accuracy and recall.

However, in Figures 4c and 4d, we note that increasing the width of the network does not yield significant gains in performance. In fact, recall appears to exhibit a negative trend for the tanh model.

We also experimented with different activation functions, such as Sigmoid and Softmax. We observed that Sigmoid slightly underperformed compared to ReLU, whereas Softmax exhibited very poor performance.

3.4 Impact of CNN Hyperparameters

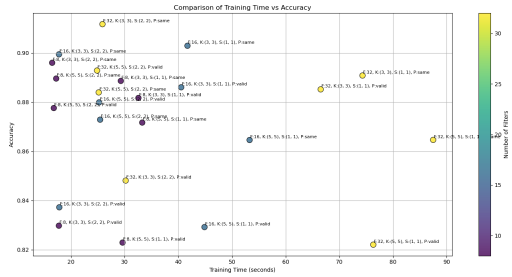


Figure 5: Test Accuracy and Training Time on various hyperparameter settings

We evaluated the modified CNN for 128×128 input size by varying the number of filters, kernel size, stride, and padding settings. As illustrated in Figure 5, a total of 24 variations were analyzed in terms of test accuracy and training time. The configuration with filter sizes [32, 64, 128] across three Conv2D layers, a 3×3 kernel size, a stride of 2, and padding set to maintain constant input and output dimensions between Conv2D layers was found to provide the best balance between accuracy and training efficiency.

4 Discussion & Conclusion

This study assessed MLPs, CNNs, and MobileNetV2 for classifying the OrganAMNIST dataset, emphasizing accuracy, recall, and computational efficiency. MLPs showed moderate performance (accuracy: 0.7037), with Tanh activation and normalized inputs yielding the best results, while L1 regularization was more efficient than L2. Modified CNNs incorporating MaxPooling and larger fully connected layers improved accuracy to 0.8325 compared to the base model. MobileNetV2 with frozen convolutional layers and trainable fully connected layers achieved the best results, with a test accuracy of 0.9249, leveraging a two-layer configuration to balance accuracy and stability. Although higher-resolution 128×128 inputs marginally improved accuracy over 28×28 models, they demanded much longer training times, limiting practicality for efficiency-focused tasks.

These results emphasize the advantages of pre-trained architectures for medical image classification. Future research could investigate methods combining MLP, CNN, and MobileNetV2 features, as well as other pre-trained models, such as EfficientNet, ResNet, or DenseNet.

5 Statement of Contributions

All team members contributed equally by independently implementing all parts of the report. Once all models were completed, we analyzed each one and merged them by selecting the best aspects of each. From these merged models, we collaboratively compiled our findings into this report.

References

[1] Sabeen Ahmed et al. "Failure Detection in Deep Neural Networks for Medical Imaging". In: *Frontiers in Medical Technology* 4 (July 2022). DOI: https://doi.org/10.3389/fmedt.2022.919046. URL: https://www.frontiersin.org/articles/10.3389/fmedt.2022.919046/full.

[2] Andrea Apicella et al. "On the effects of data normalization for domain adaptation on EEG data". In: *Engineering Applications of Artificial Intelligence* 123 (2023), p. 106205. DOI: https://doi.org/10.1016/j.engappai.2023.106205. URL: https://www.sciencedirect.com/science/article/pii/S0952197623003895.

[3] Jason Brownlee. "Weight Initialization for Deep Learning Neural Networks". In: *MachineLearningMastery.com* (Feb. 2021). Accessed: Nov. 19, 2024. URL: https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/.