Assignment 3: Classification of Image Data

COMP 551
April 1st, 2025

Androw Abd El Malek (261050237)
Kevin Luo (261038261)
Radu Petrescu (261051351)

Abstract

In this project, we implemented a multilayer perceptron (MLP) from scratch to classify images from the Kuzushiji-MNIST dataset. Our research explored the impact of neural network architectural decisions on classification performance. We systematically investigated the effects of network depth, activation functions, and regularization techniques on model accuracy.

We developed MLPs with varying configurations: a linear model, single-layer, and two-layer networks with different hidden unit counts. Our experiments revealed that increasing network complexity and introducing non-linear activation functions significantly improved classification performance. Specifically, the two-layer ReLU network demonstrated superior accuracy compared to simpler architectures, highlighting the importance of network depth and non-linearity.

Further investigation compared different activation functions (ReLU, sigmoid, and Leaky-ReLU), finding that leaky-ReLU outperformed alternatives. We also examined L2 regularization's impact, observing its potential to mitigate overfitting. Additionally, we implemented a convolutional neural network (ConvNet) and compared its performance against our MLPs, finding that with accuracy 98.17% of the ConvNet demonstrated the potential of more advanced deep learning architectures for image classification. These findings underscore the critical design choices in neural network development and provide insights into optimizing model performance for image classification tasks.

Introduction

Multilayer perceptrons (MLPs) and convolutional neural networks (ConvNets) represent fundamental architectures in machine learning for image classification tasks [1]. In this assignment, we investigate the performance and characteristics of neural network designs through a comprehensive experimental approach, exploring critical design decisions that impact model effectiveness. This will include comparing different depth MLPs amongst one another and then comparing CNNs to determine which model is better at classifying images. Our study aimed to systematically investigate neural network performance through several key experiments: analyzing the impact of network depth and non-linear transformations, comparing different activation functions (ReLU, sigmoid, Leaky-ReLU), evaluating the effects of L2 regularization, and comparing multilayer perceptron and convolutional neural network architectures.

To do this, we implemented MLPs from scratch, with a rigorous experimental design focusing on: varying hidden layer count and unit numbers, implementing different activation functions for the neurons, applying L2 regularization to our weight coefficients to ensure that the model does not overfit during training, and developing a comparative ConvNet implementation [2]. Utilizing the Kuzushiji-MNIST dataset as our benchmark, we explored how architectural choices influence classification performance.

Dataset

The Kuzushiji dataset is created by the National Institute of Japanese Literature (NIJL) and is curated by the Center for Open Data in the Humanities (CODH) [3]. In 2014, NIJL and other institutes began a national project to digitize about 300,000 old Japanese books, transcribing some of them and sharing them as open data to promote international collaboration. The full dataset can be found at https://codh.rois.ac.jp/kmnist/. This dataset is used instead of the MNIST dataset in machine learning as a nuanced alternative. Unlike standard MNIST, which focuses on Arabic numerals, Kuzushiji-MNIST captures historical Japanese script, offering a more complex classification problem. This dataset comprises 70,000 28×28 pixel images across 10 character classes, providing a rich environment for exploring neural network architectures. The dataset's structure means that for each image, the model needs to output a prediction indicating which of these 10 character classes it believes the image represents. Out of the 60,000 training data samples, each one of the 10 classes contains exactly 6000 images, and similarly for the testing set of 10,000 images, each class has 1000 images. Therefore, we do not have to worry about a specific class bias coming from our training data.

For our assignment's purposes, we separated the data into two sets: a training set containing 60,000 images and a testing set containing 10,000 images. Notably, we also standardized the data to ensure that all training and testing data is centered and scaled so that the standard deviation becomes 1. This preprocessing step is essential for neural network training, as it normalizes feature scales and accelerates convergence during optimization. Standardization also mitigates the impact of outliers and ensures that our models respond to the relative intensity patterns within the images rather than absolute pixel values, thereby improving generalization capabilities across the diverse character styles present in the dataset.

Results

Comparing MLP's

The experiment involved training three different MLP architectures on the Kuzushiji-MNIST dataset to analyze the impact of network depth and non-linearity on classification accuracy. The first model had no hidden layers and mapped inputs directly to outputs, essentially functioning as a linear classifier. The second model included a single hidden layer with ReLU activations, allowing for the learning of non-linear features. The third model extended this further by incorporating two hidden layers, also using ReLU activations. To determine the optimal configuration, each hidden layer model was trained with varying numbers of hidden units—32, 64, 128, and 256—and evaluated based on test accuracy. All models ended with a softmax layer to facilitate multi-class classification.

The results showed a clear trend: deeper networks and increased hidden units led to better performance as can be seen in Fig. 1. The MLP with no hidden layers performed the worst, achieving a test accuracy of only 69%, which is expected since a purely linear classifier lacks the ability to model complex feature interactions. Introducing a single hidden layer significantly improved performance, with test accuracy increasing from 81% (32 units) to 84% (256 units). This demonstrates how non-linearity, introduced through ReLU activations, allows the network to capture more intricate relationships within the data.

The two-hidden-layer models further enhanced accuracy, with the best configuration (128 units per layer) achieving 85%. This suggests that additional depth allows the model to extract even more meaningful hierarchical representations, although the improvement over the best single-hidden-layer model was relatively small. The results highlight a common tradeoff in deep learning: while adding layers can improve accuracy, the gains diminish after a certain point, and increasing model complexity beyond necessity can lead to diminishing returns or potential overfitting. For two layer perceptrons, the same trend of increase in units bringing an increase in accuracy was present. One key observation however was that the 32 and 64 units 2-layered MLP produced worse results than single-layer MLP. This indicates that although the ceiling of the second MLP is higher, with a restricted number of hidden units, equal or better results can be obtained using the single-layer MLP. This could be a key way to optimize training time since the backpropagation algorithm will take more time for two layers than a single one.

Overall deeper networks generally perform better, the difference between one and two hidden layers is less pronounced than the difference between zero and one hidden layer, emphasizing the crucial role of non-linearity over sheer depth. The results suggest that for tasks like Kuzushiji-MNIST, a well-configured shallow network can perform nearly as well as a deeper one, reinforcing the idea that model architecture should be chosen carefully based on the complexity of the dataset rather than assuming deeper is always better.

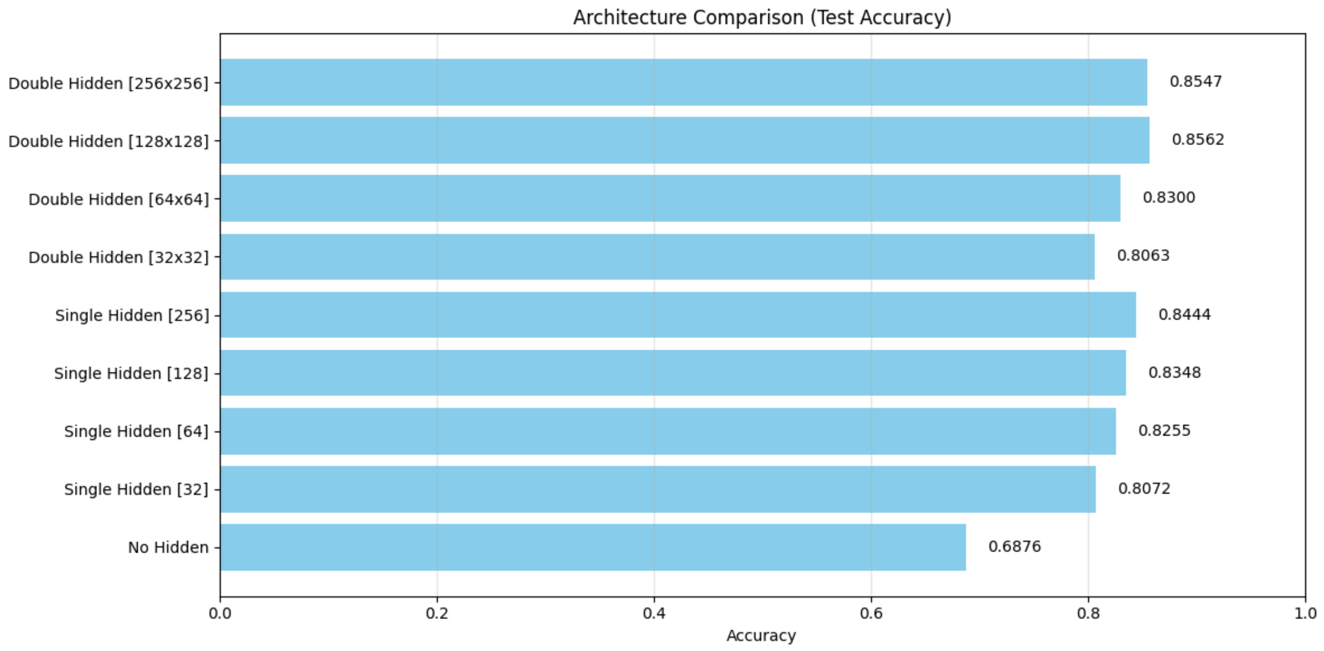

Figure 1: Comparing different MLP's

Comparing activation functions

To better understand the effect of activation functions in MLPs, we conducted an experiment which involved training three variations of a MLP with two hidden layers and 256 hidden units, each utilizing a different activation function: ReLU, Sigmoid, and Leaky-ReLU. The goal was to compare their performance based on test accuracy and analyze the reasons behind the observed differences. This was done by allowing or implementation of MLP to be called with different activation functions in its constructor.

As can be seen from Fig. 2., the results indicate that the choice of activation function significantly impacts both training dynamics and final model performance. The model using ReLU activation achieved a test accuracy of 85.56%, demonstrating strong generalization. The Leaky-ReLU variant performed slightly better, reaching 85.95%, while the Sigmoid-based model lagged considerably behind, with a test accuracy of only 68.35%. This discrepancy highlights the well-known limitations of Sigmoid activations in deep networks. Unlike ReLU and Leaky-ReLU, which are piecewise linear and do not saturate for positive inputs, the Sigmoid function squashes values between 0 and 1, leading to the vanishing gradient problem. As a result, weight updates become less effective in deeper layers, slowing convergence and ultimately reducing performance.

The poor performance of the Sigmoid model aligns with expectations, as its (0,1) output range causes vanishing gradients, limiting weight updates during backpropagation—especially in deep networks. Additionally, since Sigmoid is centered at 0.5, it lacks the natural balance of positive and negative activations, making optimization inefficient. In contrast, ReLU maintains large gradients for positive inputs and zeroes out negative values, keeping neurons active, though it suffers from the "dying ReLU" problem when neurons become inactive. Leaky-ReLU mitigates this by allowing small gradients for negative inputs, explaining its slight performance edge over ReLU. If results had deviated, factors like weight initialization, batch size, learning rate, or insufficient training epochs—particularly for the slower-converging Sigmoid—could have played a role, though Sigmoid is unlikely to match ReLU-based models in this scenario.

Overall, the findings reinforce the common recommendation that ReLU and its variants are the best activation functions for deep networks due to their ability to maintain strong gradient flow and prevent saturation.

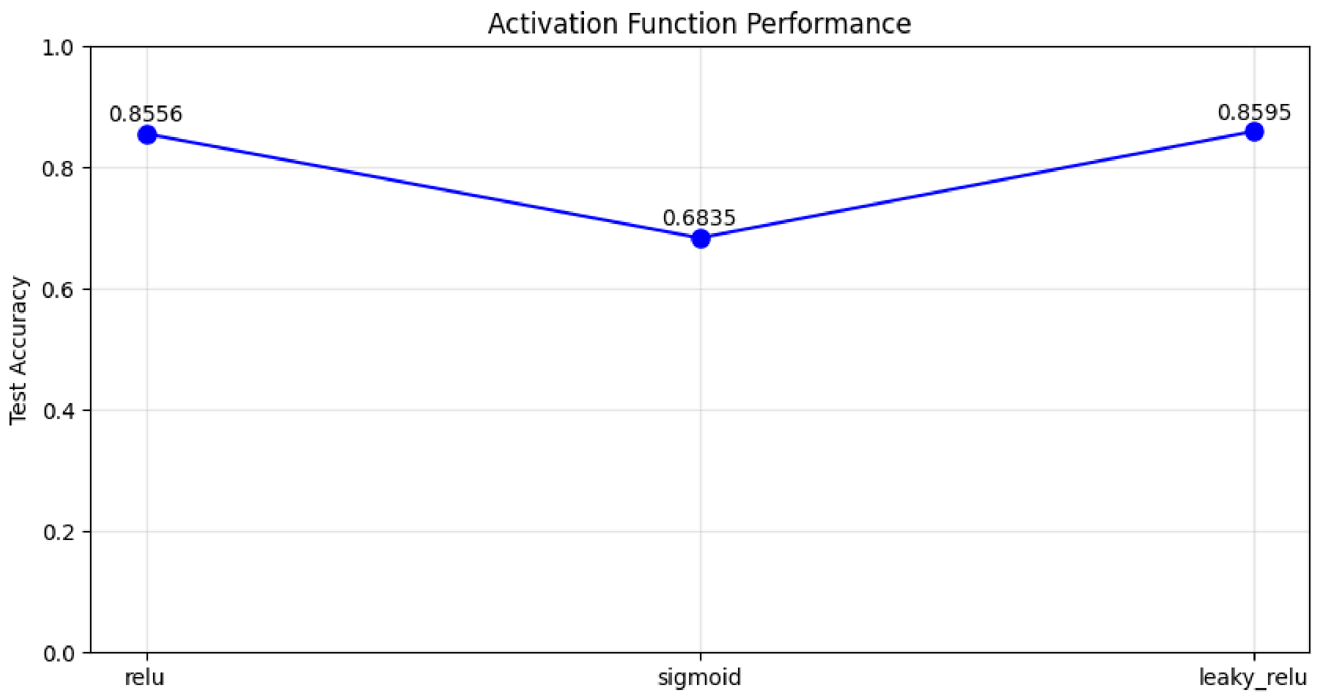

Figure 2: Accuracy of MLPs using different activation functions

Effect of L2 regularization

The experiment examined the effect of L2 regularization on an MLP with two hidden layers by varying the regularization strength (λ) and observing its impact on test accuracy. L2 regularization, also known as weight decay, penalizes large weight values by adding a term proportional to the squared magnitude of the weights to the loss function. This helps prevent overfitting by encouraging smaller weight values, leading to a more generalized model. The results indicate that the choice of λ has a subtle but noticeable influence on performance. The results from Fig. 3, indicate that with a very small λ (0.0001), the test accuracy was 84.84%. Increasing λ to 0.001 resulted in a slight decrease of the test accuracy of 84.41%. However, when λ was set to 0.01, the model was increased to 85.25% accuracy. The final λ was set to 0.1 and returned the best accuracy of 85.28%.

The theoretical expectations of L2 regularization are difficult to capture from these results. Typically, when λ is too small, the regularization effect is negligible, and the model behaves similarly to an unregularized version and as λ increases, the model becomes more resistant to overfitting, leading to slightly better generalization. However, from our results, the accuracy did not change by much from when L2 regularization was applied. Therefore, our findings expose that in this dataset, applying a regularization technique such as the L2 norm does not significantly impact the model's performance. Explanations for this vary, but distinctiveness among classes, equal sample sizes for all 10 classes can definitely help neglect the necessity for regularization of the weights.

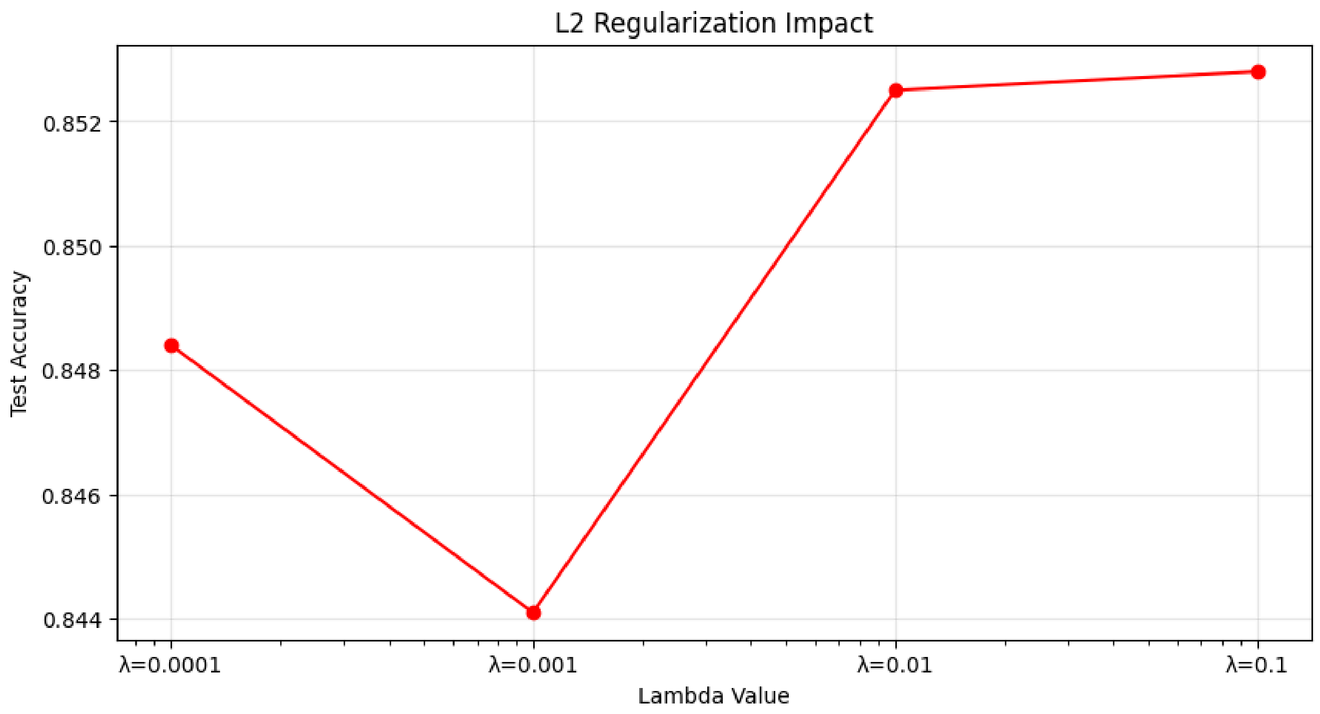

Figure 3: Effect of different lambda (λ) for L2 regularization on test accuracy

Convolution Neural Network (CNN)

In this experiment, we aim to improve the accuracy of handwritten character recognition by leveraging a Convolutional Neural Network (ConvNet) instead of a traditional Multi-Layer Perceptron (MLP). While MLPs can perform well on classification tasks, they struggle with image-based data due to their inability to exploit spatial relationships. ConvNets, on the other hand, are designed specifically for image processing, using convolutional layers to capture local patterns such as edges, curves, and textures, which are essential for distinguishing between different characters in datasets like Kuzushiji-MNIST.

To achieve this, we designed a ConvNet using the Keras library, with three convolutional layers followed by two fully connected layers, using ReLU activation throughout. The number of hidden units in the fully connected layers was treated as a hyperparameter, with values {32, 64, 128, 256} tested to identify the best-performing configuration. The network was trained on the Kuzushiji-MNIST dataset and its performance was evaluated using validation accuracy to select the most effective architecture.

The results as shown in Fig. 4 show that for 32 units, we get 97.93% accuracy, for 64 units, 98.15% , using 128 we get 98.25% and finally 256 hidden units in the fully connected layer provided the highest validation accuracy of 98.2%. Hence, we can select this 128 FC unit model to perform epoch training. Training this configuration for ten epochs resulted in a final test accuracy of 95.64%, demonstrating a significant improvement over MLP-based approaches. The enhanced performance of the ConvNet highlights its ability to learn hierarchical features that are crucial for character recognition, whereas MLPs, which lack spatial awareness, require significantly more parameters to achieve similar results.

Overall, this experiment confirms that ConvNets are more suitable for image classification tasks like Kuzushiji-MNIST, as they efficiently extract and utilize spatial information. The findings reinforce the advantages of convolutional architectures in deep learning and suggest that further improvements, such as data augmentation or deeper networks, could push performance even higher.

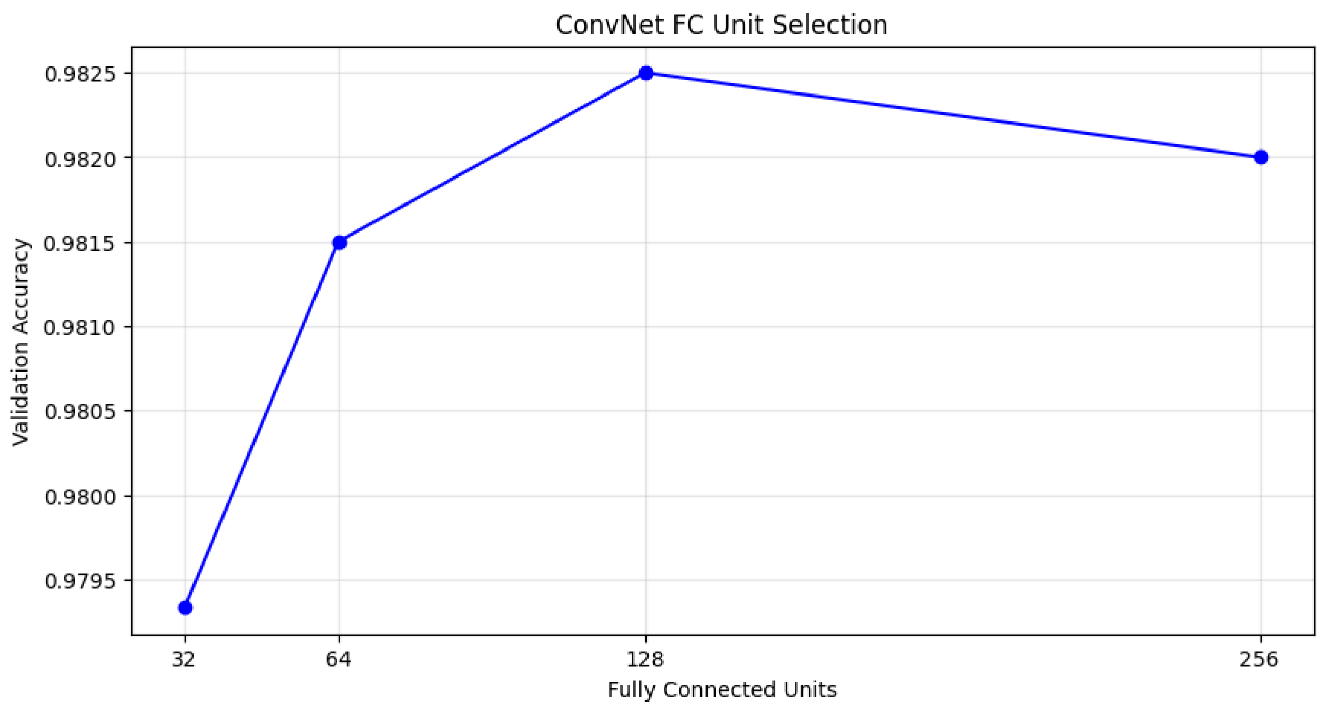

Figure 4: Convolution Neural Network accuracy for different units

Training curves

The training loss for both the Multi-Layer Perceptron (MLP) and the Convolutional Neural Network (ConvNet) decreased over time, but with significantly different behaviors. As can be seen in Fig. 5, for the MLP, the training loss started at 0.4496 and gradually decreased to 0.1242 by the 10th epoch. The decline was steady but relatively slow, and the same pattern was found for the validation loss. Comparatively, for ConvNet (fig. 6), the initial loss was high, but it dropped sharply by the 2nd epoch and continued to decrease, reaching 0.0111 by the 10th epoch. This rapid decline suggests that the ConvNet was able to learn much faster than the MLP. However, the validation loss behaved differently, fluctuating around 0.1 but never trending in a specific direction.

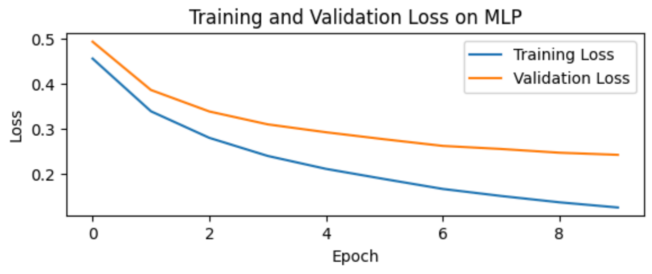

Figure 5: Training loss over epochs for MLP

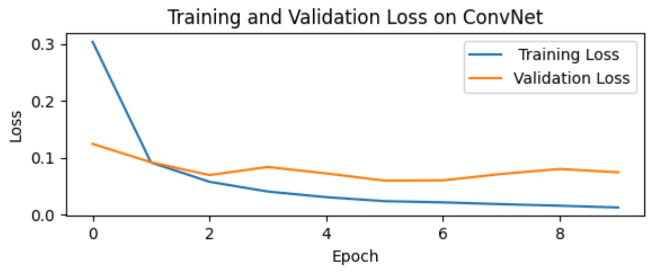

Figure 6: Training loss over epochs for ConvNet

For accuracy, the trend for MLP (fig. 7) was increasing gradually over time for both the training and validation accuracy. The validation accuracy reached 93.18% in the last epoch. In contrast, the ConvNet (fig. 8) had a sharper increase in training validation at the first epochs and then steadily increased slowly. The validation finished at 98.17% on the 10th epoch.

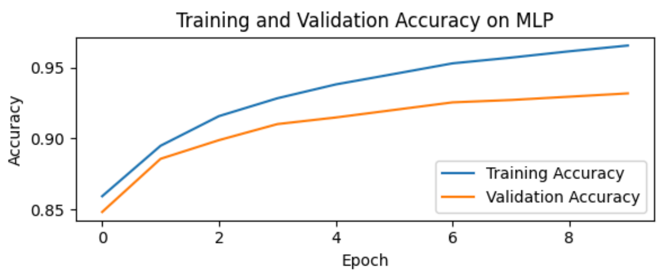

Figure 7: Training and validation accuracy as a function of epochs for MLP

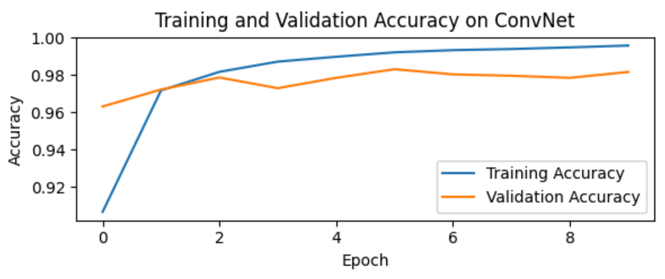

Figure 8: Training and validation accuracy as a function of epochs for ConvNet

Extra Experiments

To evaluate the impact of dropout regularization on convolutional neural network (ConvNet) performance, we trained our model with varying dropout rates and measured the resulting test accuracy. Dropout is a regularization technique that randomly deactivates a fraction of neurons during training, preventing overfitting and improving generalization. We conducted experiments with dropout rates ranging from 0.0 (no dropout) to 0.9 (fig. 9). Our results show that applying a small to moderate dropout rate (0.1–0.5) improved test accuracy compared to no dropout, with the highest accuracy (0.9549) observed at dropout rates of 0.1 and 0.3. Beyond this range, performance slightly decreased, with a noticeable drop at 0.9 (0.9366 accuracy), likely due to excessive neuron deactivation limiting the model's learning capacity. These findings suggest that a moderate level of dropout effectively enhances generalization, but excessive dropout can degrade performance by removing too much network capacity.

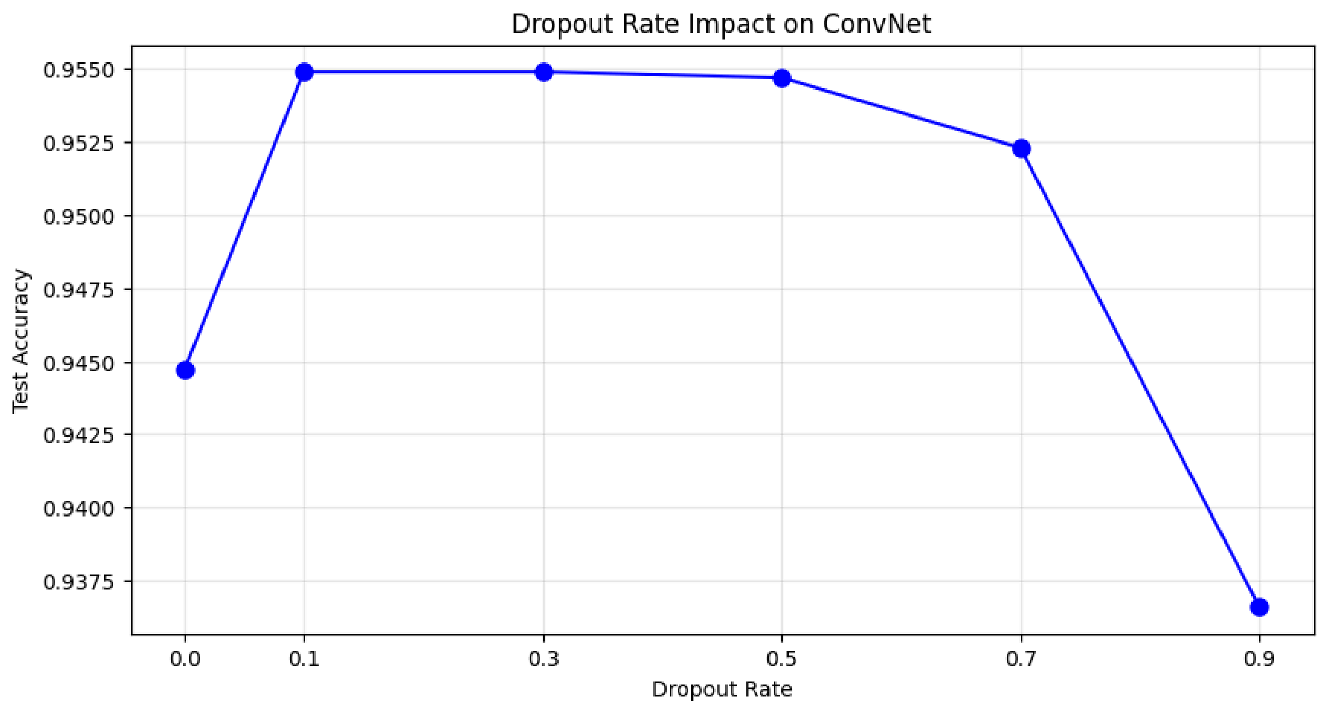

Figure 9: Dropout rate impact on testing accuracy of ConvNet

In the next experiment, we trained both a MLP and ConvNet with varying sample sizes to assess their performance and learning capabilities (fig. 10). This experiment allows us to analyze how much quicker and more accurate the ConvNet is compared to the MLP when trained on less data. For the MLP, we observed a steady decrease in loss over the 10 epochs for each sample size, indicating the model's progression in learning. However, the test accuracy remained low, especially for smaller sample sizes, with the highest accuracy (0.5567) achieved when using 10,000 samples. This suggests that the MLP requires a larger dataset to achieve significant generalization and performance. In contrast, the ConvNet demonstrated a more promising result, with a noticeable improvement in accuracy as the sample size increased. Starting with a test accuracy of 0.1000 for 1 sample, the ConvNet achieved 0.8832 accuracy with 10,000 samples. This indicates that ConvNets are more capable of learning from limited data, particularly when scaling up the dataset, likely due to their ability to capture spatial hierarchies within the images. Overall, the results highlight the effectiveness of ConvNets over MLPs, especially when larger datasets are available.

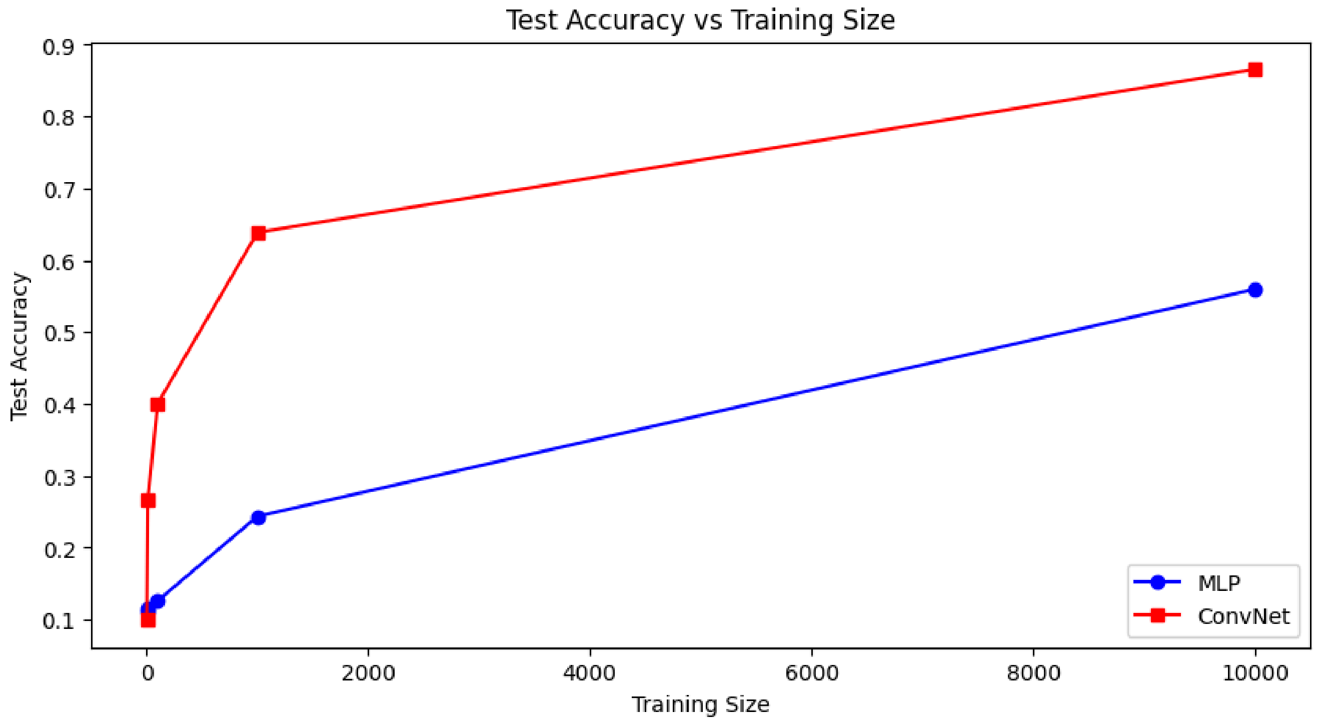

Figure 10: Testing accuracy vs Training size for MLP & ConvNet

Discussion and Conclusion

This project provided valuable insights into the effectiveness of different neural network architectures and techniques for classifying Kuzushiji-MNIST characters. Our experiments demonstrated that introducing non-linearity through hidden layers significantly improved classification accuracy, with diminishing returns after a certain depth. The choice of activation functions played a crucial role in network performance, as ReLU and Leaky-ReLU outperformed Sigmoid due to their ability to mitigate vanishing gradients. Additionally, the investigation into L2 regularization revealed that while moderate regularization can enhance generalization, its impact was less pronounced for this dataset, likely due to balanced class distributions and clear character distinctions.

One of the key takeaways from our findings is the importance of carefully selecting model complexity based on dataset characteristics. While deeper networks can extract more abstract features, a well-configured shallow network can perform nearly as well, offering computational efficiency without a significant accuracy tradeoff. Furthermore, activation function selection is critical, as inappropriate choices, such as Sigmoid in deep networks, can severely hinder performance due to gradient saturation. The L2 regularization results suggest that for well-structured datasets, the need for strong regularization techniques may be reduced, though hyperparameter tuning remains essential for optimization.

Future research directions could include seeing the impact of even deeper networks with more complex layers, or comparing the performance of MLP and ConvNet with more static models such as DT or KNN. Additionally, experimenting with hyperparameter tuning on the ConvNet can further improve the accuracy despite already achieving a high accuracy of 98.17%. Investigating alternative regularization methods, such as LASSO, might also give interesting results.

Statement of Contribution

Each team member contributed to the coding and writing part rather equally. Pair programming sessions were held where all members contributed to ensure full understanding of the models and implementation before going on to produce results (graphs) and writing the report.

References

[1] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278-2324.

[2] Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)*.

[3] Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., & Ha, D. (2018). Deep Learning for Classical Japanese Literature. *NeurIPS 2018 Workshop on Machine Learning for Creativity and Design*.