**Assignment 1**  
Neural Networks

short line

Rohit Sharma

Student ID: 24590960  
02/04/2024

94691 – Deep Learning

Master of Data Science and Innovation

University of Technology of Sydney

Table of Contents

[Introduction 3](#_Toc162965697)

[Data Understanding and Preparation 3](#_Toc162965698)

[Data Preparation 3](#_Toc162965699)

[Methodology 4](#_Toc162965700)

[Training and Evaluation 5](#_Toc162965701)

[Result Analysis 6](#_Toc162965702)

[Limitations and Recommendations 9](#_Toc162965703)

[Conclusion 10](#_Toc162965704)

# Introduction

Neural networks are at the forefront of the rapidly evolving knowledge of artificial intelligence, driving breakthroughs in a variety of industries, including the automotive and healthcare sectors. Because of its robust ecosystem and dynamic computing graph, PyTorch has become a preferred tool for neural network development and training among practitioners and researchers.

The practical use of PyTorch to train a neural network serves as the foundation for this paper. This assignment not only demonstrates the efficiency and flexibility of the framework, but also emphasizes the vital role that neural networks play in parsing and understanding complicated datasets.

We explore the complexities of training, the architecture of neural networks, and the real-world obstacles to obtaining the best possible model performance.

# Data Understanding and Preparation

Here, we examine the 70,000 images of handwritten Hiragana characters found in the Japanese MNIST dataset. The dataset complexity is evident in its diverse range of characters, categorized into 10 different classes, presenting a unique challenge in pattern recognition and classification. Each image, originally in 28x28 pixel format, provides a grayscale snapshot of individual Hiragana characters, showcasing various handwriting styles.

## Data Preparation

The preprocessing of the dataset involved several steps to ensure the data is in the optimal format for neural network processing:

1. Reshaping images: Each image was initially in a 28x28 pixel format, without a channel dimension. We reshaped these images to include the channel dimension, adjusting them to a (row\_number, height, width, channel) format. This modification is crucial for feeding the data into our neural network, which expects a specific input shape.
2. Type Conversion: The image matrices were then cast into ‘float32’ data type. This conversion is necessary for computational efficiency and compatibility with the neural network framework that processes floating-point numbers more effectively.
3. Standardization: The pixel values, originally ranging from 0 to 255, were standardized to fall within a 0 to 1 range. This normalization step is vital for neural network models, as it helps in speeding up the convergence during training by ensuring that all input features are on a similar scale.
4. Flattening: Post Standardization, the images were flattened from a 3D format to a 1D vector (784,1). This step transforms each image into a single vector, making it compatible with the input layer of our fully connected neural network.

Through these steps, the dataset was transformed from its raw form into a structured format, ready for training neural network models.

# Methodology

Our experiments are built around the definition and iteration of neural network topologies. PyTorch, an effective and flexible deep learning framework that makes neural network construction and training easier, was used to build the designs.

ReLU activation functions and a series of fully-connected (linear) layers made up the baseline model. The input layer of the network architecture was flattened from the 28x28 photos to a size of 784. This was followed by two hidden layers, each with 512 neurons, and an output layer with 10 neurons, one for each of the dataset's ten classes. Non-linearity was introduced using ReLU activations, which is necessary to identify intricate patterns in the data.

In later trials, dropout layers were added to prevent overfitting. Dropout is a regularization approach that helps keep the network from becoming overly dependent on any one neuron by randomly setting a portion of input units to 0 during training. In order to determine the ideal ratio between regularization and model complexity, dropout rates were tried.

Three experiments were conducted, each varying in architecture.

Our starting point model was Experiment 1, where we focused on establishing a baseline performance without adding dropout layers. A straightforward stack of fully connected layers made up the architecture, and each of the two hidden layers had 512 neurons. By introducing non-linearity using ReLU activation functions, the model was able to learn complicated patterns. Since there were no regularization strategies to reduce overfitting, we were able to evaluate the model's inherent ability to classify the characters due to its initial architecture's simplicity. As we initiated the training, the values plummeted to zero in the late epochs, which might raise initial impressions of model performance. Considering overfitting—a phenomena in which the model learns the training data to such a degree that it performs poorly on unknown data due to its inability to generalize—this pattern does, however, merit more investigation. The test set's impressive **92.30%** accuracy rate indicates that, despite possible overfitting worries, the model was still able to demonstrate a respectable ability to generalize to new data. Given the lack of regularization methods like dropout, which are usually used to reduce overfitting, this performance is noteworthy.

Experiment 2 introduced Dropout as a key differentiation. Following each hidden layer, dropout layers were added at a rate of 0.3 to introduce unpredictability to the training process by temporarily dropping out a fraction of neurons. This method is well-known for its ability to prevent overfitting by limiting the model's dependence on any one neuron. We added more layers and neurons to the network, which increased its complexity in addition to dropout. This extension was motivated by the idea that a more intricate model may capture finer details in the data and attain higher accuracy when appropriately regularized with dropout.

The first epoch of the experiment began with a moderate loss of 0.449, indicating the first discrepancy between the model's predictions and the actual data. An impressive increase in loss reduction was shown as training went on through the epochs, indicating the model's increasing ability to identify and categorize handwritten Hiragana letters.  
The loss dramatically decreased to 0.049 by the tenth epoch, demonstrating the potency of the model architecture and training schedule. Though at a reduced rate, this trend of declining loss persisted, suggesting that the model was getting close to its ideal condition where making further weight adjustments would only have a diminishing return on loss reduction.

The ensuing periods of time showed a steady but slow decrease in loss, with variations. These variations showed how the model was searching the weight space for combinations that would reduce the loss even more. The model achieved an astonishingly low loss by the 500th epoch, yet overall, the trend remained clearly towards loss minimization despite these fluctuations. Experiment 2's completion was indicated by a test accuracy of **93.42%**.

Further in Experiment 3, to achieve a fine balance between regularization and the network's ability to learn from the training data, we raised the dropout rate to 0.5 in this iteration. The reasoning behind this was that stronger regularization from a larger dropout rate could further reduce overfitting. We also tweaked additional hyperparameters, such lowering the learning rate, to improve the training dynamics of the model. This modification was based on the idea that, given a highly regularized environment, a slower learning rate could facilitate a smoother convergence of the model to an optimal set of weights. The first epoch showed a significant drop in loss, from 0.645 to a more accurate 0.309 by the second, suggesting that the model learned and adjusted quickly. As the model became more adept at correctly classifying the handwritten Hiragana letters, the trend of decreasing loss persisted; by the 500th epoch, the loss had stabilized at 0.0149. This experiment produced an impressive result, with a test accuracy of **92.14%**.

# Training and Evaluation

In the training and evaluation process, Adam optimizer was chosen to train the models because of its effectiveness in managing sparse gradients and adjusting the learning rate for various parameters. The 500 epochs that made up the training regimen were chosen to provide the models enough time to learn and adjust to the complexities of the data without having to stop the learning process too soon. The loss function, more specifically the cross-entropy loss, which measures the difference between the predicted probabilities and the actual class labels, was the main statistic used to monitor the training process. The models underwent a thorough assessment to see how well they performed on unobserved data. In order to make sure that the models properly learnt the training set and applied their newly acquired knowledge to previously unseen data, this evaluation was essential. The accuracy statistic, which measures the percentage of test photos that the models properly identified, was the main focus of the evaluation process.

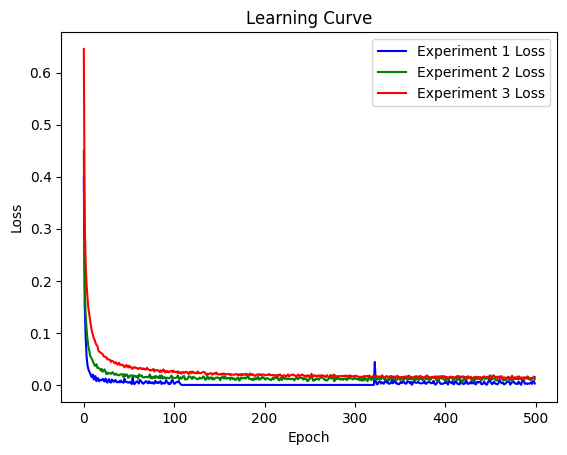
Experiment 1 produced a robust baseline with no dropout and achieved a noteworthy accuracy, suggesting that the model has the innate capacity to identify and classify the characters with a high degree of success. Nonetheless, the inclusion of dropout in Experiment 2 and additional optimization in Experiment 3 demonstrated how regularization works to prevent overfitting, a common mistake made while training neural networks.   
The training and assessment stages were distinguished by a strong emphasis on accuracy enhancement and loss reduction, with the Adam optimizer being essential in overcoming the difficulties presented by the dataset. The resulting high accuracy rates—particularly Experiment 2's 93.26% accuracy—tell a powerful story about how well-calibrated models may perform astonishingly well on challenging categorization tasks.

# Result Analysis

All three experiments' learning curves show the typical early drop in loss that denotes significant learning gains. The pace of loss reduction eventually reaches a plateau, indicating that the models are getting close to a minimal error state where more learning modifications won't have much of an impact. The models' ability to learn effectively from the training data is indicated by this pattern.

The first experiment's learning curve quickly stabilizes, indicating that the model might be prone to overfitting in the absence of regularization strategies like dropout. This is typical of models without methods to stop noise from being learned alongside the signal in the training set.

These findings are further supported by the confusion matrices. While the matrix in Experiment 1 displays great accuracy along the diagonal, there are some noticeable misclassifications that offset this positive result. These inaccuracies draw attention to the model's difficulties differentiating between particular classes, which may be lessened by additional model tuning or data preprocessing.



*Fig 1: Learning Curve*

A graph with blue squares and numbers

Description automatically generated

*Fig 2: Confusion Matrix for Experiment 1*

Dropout regularization is introduced in Experiment 2, and the corresponding confusion matrix illustrates the advantages of this strategy. A slight decrease in the amount of misclassifications suggests better model generalization. This improvement is supported by the learning curve, which approaches the plateau more gradually, suggesting that the model is making better use of the dropout to generalize outside of the training set.

A graph with blue squares

Description automatically generated

*Fig 2: Confusion Matrix for Experiment 2*

Even further regularization in the confusion matrix from Experiment 3 shows improved class differentiation. The learning curve's downward slope before plateauing indicates that a more cautious approach to learning combined with higher dropout rates results in a model that is more generalizable and less prone to overfitting.

A graph with blue squares

Description automatically generated

*Fig 3s: Confusion Matrix for Experiment 3*

# Limitations and Recommendations

The possibility of overfitting was one important limitation that we found, especially early in our research. A model is said to be overfit when it learns the training set too well, so capturing noise in addition to the underlying pattern. This impairs the model's capacity to generalize to new sets of data. Without regularization strategies like dropout, experiment 1 demonstrated excellent accuracy on the training set but raised questions about the model's capacity for generalization. Moreover, Experiment 3's dropout method added complexity to the model training procedure even though it was intended to act as a regularization strategy to reduce overfitting. Determining the optimal dropout rate and learning rate required iterative tuning and could vary significantly across different datasets and model architectures.

Even though it's essential for maximizing model performance, hyperparameter tuning has time and computing resource constraints, particularly when working with bigger and more complicated datasets. Although our models showed remarkable performance in handwritten Hiragana character classification, the difficulties we faced underscore the continuous difficulties in machine learning. Improvements can be achieved in the following areas: hyperparameter adjustment, model architectural decisions, dataset variability, and overfitting. To overcome these constraints, novel methods for training and developing models are needed, in addition to a deeper comprehension of the underlying data and the difficulties it poses.

We can explore several methods to improve future models by tackling the difficulties and constraints we have found when investigating neural network models.  
Convolutional layers could be integrated into model design to advance its handling of spatial hierarchies in picture data, providing a more sophisticated learning capability. This could improve the ratio of learning depth to generalization, along with a wider use of regularization approaches beyond dropout.

Automated methods for optimizing hyperparameters hold the potential to simplify the process of model tuning, hence facilitating a more effective pursuit of the ideal model configuration.  
All together, these suggestions seek to improve neural network models' performance and usefulness in picture classification tasks, directing future research efforts toward the creation of increasingly complex, precise, and broadly applicable machine learning solutions.

# Conclusion

Neural network model creation and evaluation for the Japanese MNIST dataset has highlighted the promise and difficulties associated with machine learning activities. The limits that were found brought to light the necessity of continued investigation and study in the area.  
The suggestions made seek to overcome these obstacles and give a path forward for further research that expands on our discoveries. By investigating more complex architectures, improving regularization methods, and fine-tuning hyperparameters, we can get closer to creating machine learning models that are resilient, generalizable, and highly accurate for a variety of applications.