**CSE5ML**

**Machine Learning**

Assessment 2: Report

**Name:** Ben Kereopa-Yorke

**Email:** 21340711@students.latrobe.edu.au

**Student Number:** 21340711

**Date:** 17.9.2023

**Table of Contents**

Cover Page Page 1

Table of Contents Page 2

Introduction Page 3-4

Optimiser and Learning Rates Page 5

Experiments and Performance Page 5-6

Discussion on Neural Network Results Page 7

Conclusion and Neural Network Rankings Page 7

**Introduction**

The MNIST dataset, an acronym for Modified National Institute of Standards and Technology database, stands as a cornerstone in the machine learning domain, particularly for beginners delving into the field. Consisting of 60,000 training images and an additional 10,000 for testing, these grayscale snapshots of handwritten digits, each intricately bounded within a 28x28 pixel frame, serve as a standard benchmark for classification algorithms. The real complexity arises not just from recognizing the nuances of human handwriting, but in precisely assigning these images into one of the ten numerical classes (0-9) they signify. Its ubiquity in academic research and courses underpins its foundational importance in pattern recognition and deep learning exercises.

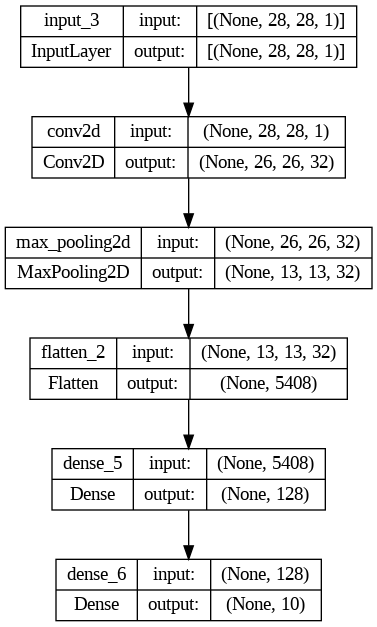
I present below the visual representations of the deep learning models constructed for the purpose of classifying handwritten digits from the MNIST dataset. Each diagram offers a structured view of the neural network's layers, showcasing how data flows and transforms as it progresses from input to output. The initial model diagrams serve as my foundational architecture, while the improved models reflect modifications made to enhance performance. By juxtaposing these architectures, you can gain a comprehensive understanding of the network's complexity and the strategic choices made in model design to better capture patterns and features within the handwritten digits.

A diagram of a computer

Description automatically generatedA diagram of a function

Description automatically generated

Figure – Model 1 Figure – Model 1 Improved

A diagram of a computer program

Description automatically generated

Figure – Model 2 Figure – Model 2 Improved

**Optimiser and Learning Rates**

In Task 2, I employed the Adam optimiser with its default learning rate of 0.001. This optimiser adjusts learning rates for each parameter, combining the strengths of AdaGrad and RMSProp.

For Task 3, I transitioned to the Stochastic Gradient Descent (SGD) optimiser with a learning rate of 0.01. While SGD can often converge faster than other methods, its constant learning rate can pose challenges. The chosen learning rate balanced efficient convergence with model stability. It's worth noting that the learning rate plays a crucial role: too high, and we risk overshooting the optimal state; too low, and convergence may be painstakingly slow.

**Experiments and Performance**

In the experiments, I evaluated the performance of different neural network architectures across a series of training epochs. The performance metrics provided for each epoch include the loss and accuracy scores. After training, I also tested the models on a separate dataset to gauge their overall accuracy. Here's a summarisation of the results:

**Table: Model Training and Testing Results**

| **Model** | **Epoch** | **Loss** | **Training Accuracy** | **Test Accuracy** |
| --- | --- | --- | --- | --- |
| **Model 1** | 1 | 0.2691 | 0.9226 | 0.9747 |
|  | 2 | 0.1202 | 0.9645 |  |
|  | 3 | 0.0820 | 0.9755 |  |
|  | 4 | 0.0612 | 0.9814 |  |
|  | 5 | 0.0476 | 0.9855 |  |
| **Model 1 Improved** | 1 | 0.2032 | 0.9404 | 0.9775 |
|  | 2 | 0.0844 | 0.9738 |  |
|  | 3 | 0.0564 | 0.9819 |  |
|  | 4 | 0.0435 | 0.9856 |  |
|  | 5 | 0.0338 | 0.9888 |  |
| **Model 2** | 1 | 0.1477 | 0.9564 | 0.9842 |
|  | 2 | 0.0489 | 0.9850 |  |
|  | 3 | 0.0324 | 0.9900 |  |
|  | 4 | 0.0213 | 0.9932 |  |
|  | 5 | 0.0133 | 0.9956 |  |
| **Model 2 Improved** | 1 | 0.1149 | 0.9650 | 0.9879 |
|  | 2 | 0.0388 | 0.9877 |  |
|  | 3 | 0.0251 | 0.9921 |  |
|  | 4 | 0.0181 | 0.9944 |  |
|  | 5 | 0.0137 | 0.9958 |  |
| **Model 3 (SGD Optimizer)** | 1 | 0.0062 | 0.9980 | 0.9928 |
|  | 2 | 0.0038 | 0.9989 |  |
|  | 3 | 0.0032 | 0.9991 |  |
|  | 4 | 0.0027 | 0.9994 |  |
|  | 5 | 0.0024 | 0.9994 |  |

The table showcases the training performances over 5 epochs for each model and their corresponding test accuracies. We can see that, over time, each model shows improvement in terms of loss and accuracy. Model 3, using SGD as an optimiser, exhibits exemplary training accuracy nearing 100%. However, it's vital to interpret these results, considering overfitting, especially when the training accuracy approaches perfection.

**Discussion on Neural Network Results**

Upon closely analysing the experiments' outcomes, it's evident that the progression from initial models to their enhanced versions contributed to superior performance, especially in terms of accuracy metrics. The architectural refinements, including the incorporation of additional layers and neurons, endowed the networks with a heightened capability to discern nuanced patterns inherent in the MNIST dataset. Notably, the optimiser plays a pivotal role in the training dynamics. While Adam's adaptiveness furnished commendable results, the traditional SGD, when paired with a judiciously selected learning rate, rivalled its performance. This reiterates the nuanced interplay of hyperparameters and architecture in determining a model's efficacy. It also emphasises that, despite the allure of more sophisticated algorithms, foundational techniques, when appropriately calibrated, can remain highly relevant in the ever-evolving machine learning landscape.

**Conclusion and Neural Network Rankings**

**Table of Neural Network Performance Rankings:**

| **Rank** | **Model Name** | **Test Accuracy** |
| --- | --- | --- |
| 1 | Model 3 (SGD Optimizer) | 0.9928 |
| 2 | Model 2 Improved | 0.9879 |
| 3 | Model 2 | 0.9842 |
| 4 | Model 1 Improved | 0.9775 |
| 5 | Model 1 | 0.9747 |

In summarising the performance of the neural networks across all tasks, the rankings reveal that Model 3, which utilised the SGD optimiser with a learning rate of 0.01, achieved the highest test accuracy. This was closely followed by the improved version of Model 2 and then its initial version. The initial and improved configurations of Model 1 took the fourth and fifth spots respectively. The results underscore the significance of model architecture, optimiser choice, and hyperparameter tuning in influencing the outcome. It's also evident that traditional optimisation strategies, like SGD, can still excel when correctly implemented, even when compared to more adaptive methods like Adam.