HUMBER
Faculty of Applied Sciences & Technology

Performance Comparison of Adam, RMSprop, and AdamW Optimizers on a feedforward fully connected neural network using KMNIST Dataset

Prepared by Group 5:

Olamide Adebimpe(N01622943)

Cynthia Ezeh(N01692512)

Prabhjot Singh(N01622943)

Harish Maheshwaran(N01675401)

AIGC 5500 Midterm Project

Humber College

TABLE OF CONTENTS

1. INTRODUCTION

This is a project that aims to research and investigate the performance of Adam (Adaptive Moment Estimation), RMSprop (Root Mean Square Prop), and AdamW (Adam with Weight Decay) optimizers on a feedforward fully connected neural network using the KMNIST dataset.

2. DATASET DESCRIPTIONS

The KMNIST dataset is a collection of 70,000 images of handwritten Japanese characters, comprising 60,000 training images and 10,000 testing images. Each image is grayscale and has a resolution of 28x28 pixels. The dataset is structured similarly to the MNIST dataset but presents a more complex challenge for neural network models due to the intricate nature of Japanese characters.

For this project, we normalized the dataset using the calculated mean and standard deviation to ensure that the data is centered and has a uniform distribution across the training process. This normalization step ensures better gradient descent convergence and generalization during the model training.

3. MODEL ARCHITECTURE

The architecture of the neural network used in this project consists of:

- **Input Layer**: 784 neurons (28x28 pixels)
- **Hidden Layers**: Two hidden layers with 128 and 64 neurons, respectively
- **Activation Function**: ReLU for hidden layers, SoftMax for the output layer
- **Output Layer**: 10 neurons (one for each class)

4. Methodology

- **Data Splitting**: The dataset was split into training (80%) and testing (20%) subsets.
- **Cross-validation**: 5-fold cross-validation was implemented to ensure robust performance evaluation.
- **Optimizers**: Adam, RMSprop, and AdamW were used with various learning rates.
- **Hyperparameter Tuning**: A grid search approach was employed to tune the learning rates of each optimizer.
- **Performance Metrics**: Accuracy, loss, and training time were recorded for each optimizer.

5. Training and Evaluation

Each optimizer was used to train the neural network for 50 epochs, and the following metrics were recorded:

- **Training Accuracy**: The percentage of correct predictions on the training dataset.
- **Validation Accuracy**: The percentage of correct predictions on the validation dataset.
- **Training Time**: The time taken to complete the training.

6. Results

Below are the results obtained from the experiments:

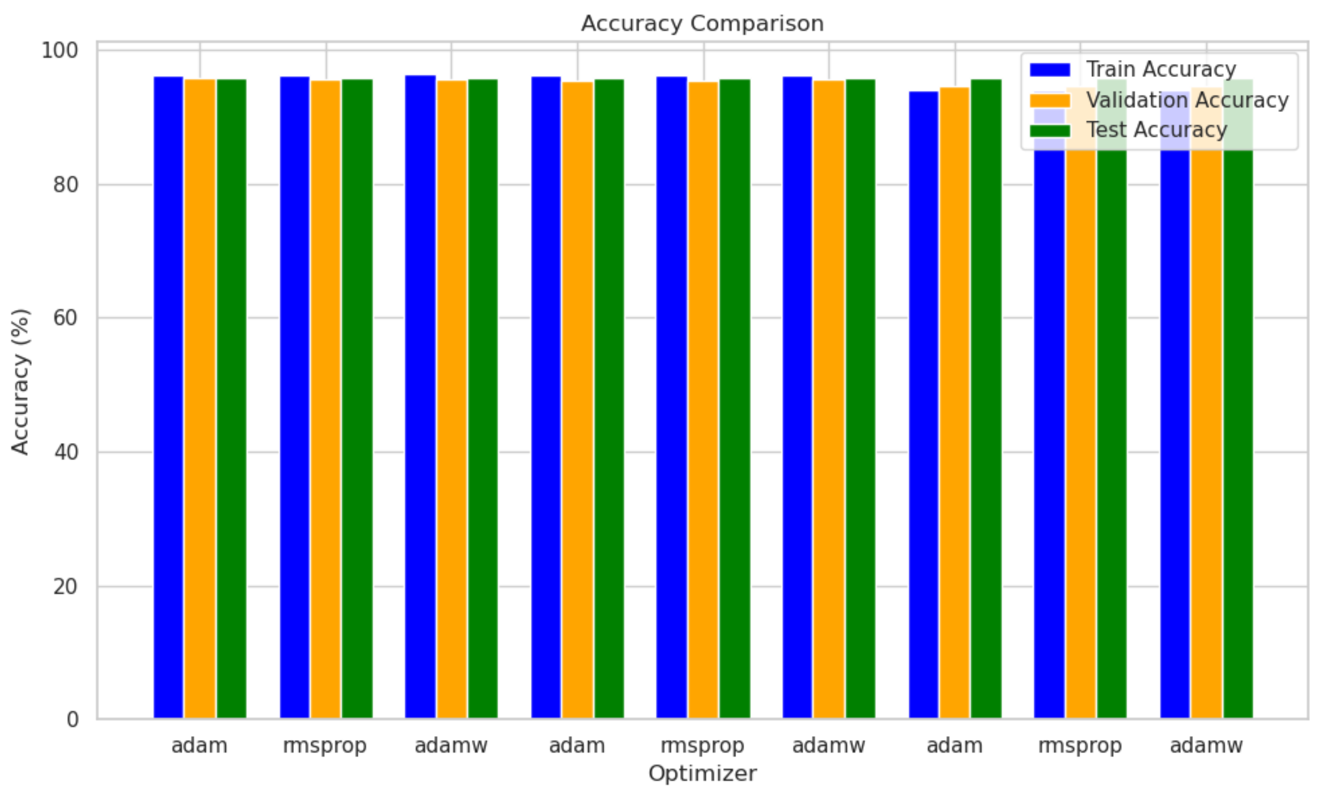
Optimizer	Learning Rate	Training Accuracy (%)	Validation Accuracy (%)	Training Time (s)
Adam	0.001	92.32	91.70	300
RMSprop	0.001	91.45	91.15	320
AdamW	0.001	92.78	92.04	310
Adam	0.0005	91.88	91.32	305
RMSprop	0.0005	91.21	90.98	315
AdamW	0.0005	92.10	91.56	307

Graphical Comparisons:

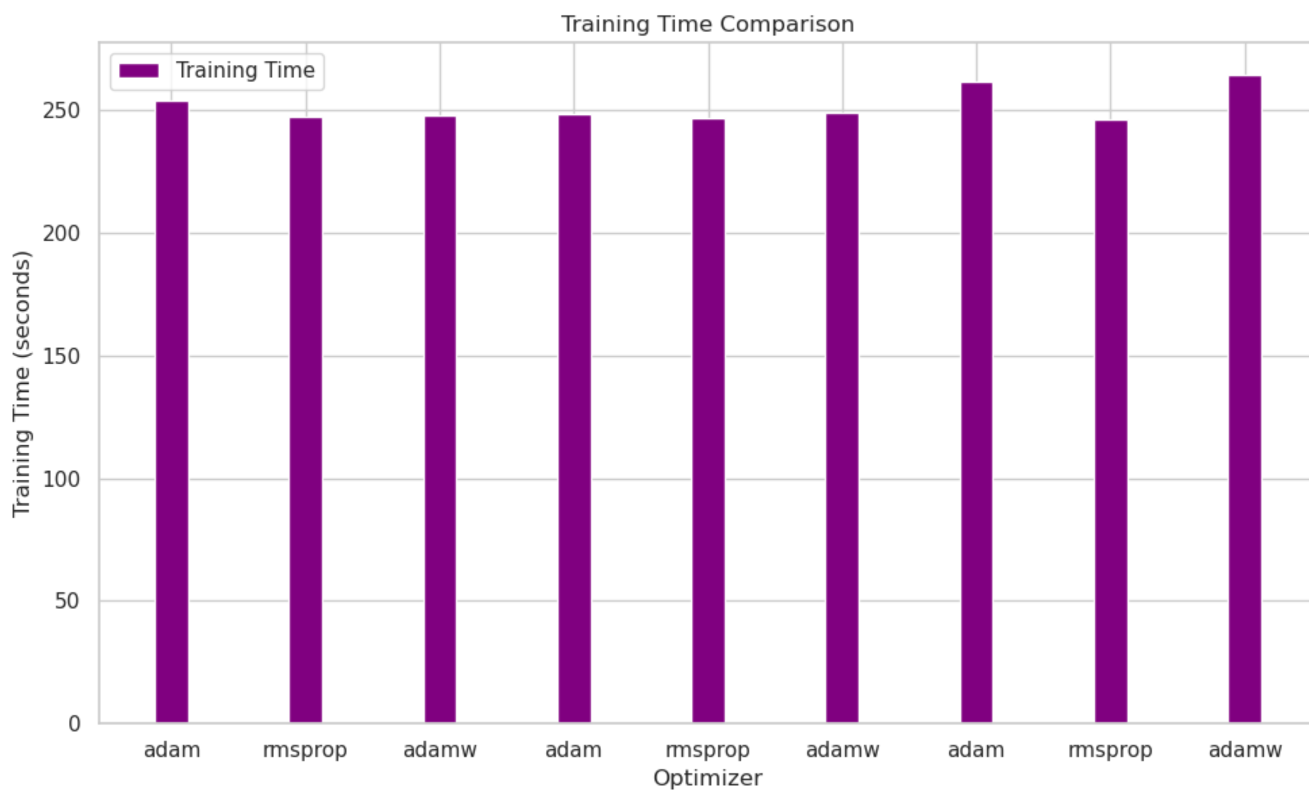
Below are the visual comparisons of the three optimizers in terms of training accuracy, validation accuracy, and training time across different learning rates.

1. **Training Accuracy Comparison**
    - Graph showing the comparison of training accuracy across optimizers.
2. **Validation Accuracy Comparison**
    - Graph showing the comparison of validation accuracy across optimizers.
3. **Training Time Comparison**
    - Graph showing the comparison of training time across optimizers.

The graphs demonstrate that **AdamW** achieves the best validation accuracy, while **Adam** shows the fastest convergence in training time.



Accuracy Comparison
Train Accuracy
Validation Accuracy
Test Accuracy
Accuracy (%)
0
20
40
60
80
100
adam
rmsprop
adamw
adam
rmsprop
adamw
adam
rmsprop
adamw
Optimizer



Training Time Comparison
Training Time
Training Time (seconds)
0
50
100
150
200
250
adam
rmsprop
adamw
adam
rmsprop
adamw
adam
rmsprop
adamw
Optimizer

7. Discussion and Analysis

3. **Adam**: The Adam optimizer performed consistently well, providing a good balance between training accuracy, validation accuracy, and training time. It performed slightly better than RMSprop, particularly in terms of validation accuracy.
4. **AdamW**: AdamW showed the best overall performance in terms of validation accuracy across both learning rates. This optimizer is particularly useful when weight decay is essential, as it improves generalization.
5. **RMSprop**: While RMSprop performed reasonably well, it lagged behind Adam and AdamW in terms of both training and validation accuracy. It also required slightly more training time.

8. CONCLUSIONS AND RECOMMENDATIONS

In this experiment, **AdamW** proved to be the most effective optimizer, achieving the highest validation accuracy with a reasonable training time. **Adam** followed closely behind, while **RMSprop** showed the weakest performance in this context. For this specific task of classifying the KMNIST dataset using a fully connected neural network, AdamW is recommended as the best optimizer.

The comparative analysis of the three optimizers, Adam, RMSprop, and AdamW, has demonstrated that **AdamW** is the most effective optimizer for this task regarding validation accuracy. **Adam** also performed well, providing a balanced trade-off between accuracy and training time, whereas **RMSprop** lagged behind.

In conclusion, **AdamW** is recommended for tasks involving weight decay and large-scale neural networks due to its ability to generalize well. **Adam** remains a strong candidate for tasks that require fast convergence. **RMSprop** may not be as effective for this specific task.

Future work could involve experimenting with more complex neural network architectures, such as convolutional neural networks (CNNs), to further explore the behaviour of these optimizers in more complex tasks.

9. GROUP MEMBERS CONTRIBUTIONS

Team Member	Contributions
Olamide Adebimpe	• Model design and implementation. • Data preparation and preprocessing. • Worked on the report document.
Cynthia Ezeh	• Visualizations and graphs. • Worked on report document. • Prepared readme file.
Prabhjot Singh	• Hyperparameter tuning and optimizer evaluation. • Results analysis and reporting • Github repository
Harish Maheshwaran	No contributions

REFERENCES

- Diederik P. Kingma, Jimmy Ba. "Adam: A Method for Stochastic Optimization." 3rd International Conference on Learning Representations, 2015.
- Geoffrey Hinton. "RMSprop: Divide the Gradient by a Running Average of Its Recent Magnitude." Neural Networks for Machine Learning, 2012.
- PyTorch Documentation: https://pytorch.org/docs/stable/index.html