MADAM: A METHOD FOR STOCHASTIC OPTIMIZATION USING NAG (NESTEROV ACCELERATED GRADIENT)

Aman Bhadkariya

23124004, Mathematics and computing,

IIT BHU

aman.bhadkariya.mat23@itbhu.ac.in

Anurag Gupta

23124010, Mathematics and computing,

IIT BHU

anurag.gupta.mat23@itbhu.ac.in

ABSTRACT

We introduce Madam (Modified Adaptive Moment Estimation), a novel neural network optimization algorithm inspired by a popular optimizer Adam (Adaptive Moment Estimation), for enhancing performance and convergence stability Madam introduces some additional parameters which dynamically helps in gradient updates to improve the optimization of the neural network. It also implements a unique look-ahead mechanism which is inspired by the concept of NAG (Nesterov Accelerated Gradient) which recalculates update direction of parameter periodically by maintaining effective balancing between momentum and adaptive learning rate adjustments. The implementation of Madam is fairly straightforward and is computationally efficient but has some memory requirements like Adam

1. INTRODUCTION

Effective neural network optimization is a central challenge in training of a deep learning model, as training requires minimizing a loss function over a very high-dimensional parameter spaces. Other traditional optimizers are simple and efficient but suffers problems like slower convergence to minimum loss and may also trap in local minima, especially in complex or non-convex optimization landscapes and give suboptimal solutions such Stochastic Gradient Descent (SDG). Introduction of adaptive optimizers like Adam (Adaptive Moment Estimation) provides a better alternative because of their ability to dynamically adjust learning rates for every parameter which enhances the speed of convergence to minima and stability in most of the scenarios.

Adam combines the mechanism of adaptive learning rate and momentum, making Adam a versatile and robust optimizer, but Adam can sometimes face some issues and may require a proper hyperparameter tuning in a complex scenarios. Inspired by these challenges in neural network optimizations, we introduce Madam a new optimization algorithm that extends Adam with additional features for accelerated convergence and stabilized optimization.

Madam dynamically adjust gradient updates by introducing an extra set of parameters, this enhances the optimizer's ability to navigates landscapes of complex loss surfaces. Madam integrates a unique look-ahead mechanism which is inspired by Nesterov Accelerated Gradient (NAG), this mechanism helps Madam to recalculate updates direction of all the parameters periodically, efficiently balancing momentum with adaptive learning rates. Look-ahead mechanism helps Madam to anticipate and reduce the unnecessary oscillations occurred due to abrupt changes in the direction of gradient while navigating the loss surface when approaching minima.

In the implementation of Madam, computational efficiency remains similar to Adam but due it to implementation of its additional components results in slightly higher memory usage, as it maintains more information for each parameter. But Madam's memory requirements doesn't cause issues because Madam's memory requirements still remains practical most of deep learning applications, making it feasible for research purposes and industry use.

In this report, we will explain the detail and implementation of Madam and evaluates its performance on various deep learning tasks. **We will demonstrate that Madam can achieve competitive and superior results in some cases**, convergence and stability in comparison of Adam and other optimizers.

2. METHODOLOGY

The base of Madam optimizer is original Adam algorithm but with additional adjustments and integrating a new look-ahead mechanism which is inspired by Nesterov Accelerated Gradient (NAG) to improve convergence stability and gradient optimization. In this section we will discuss about the architecture and implementation details of Madam.

2.1 Overview of Madam's algorithm

Madam extends the original Adam optimizer by introducing some additional terms that assist in dynamic gradient updates while navigating the loss surface. In the original Adam optimizer, the momentum and adaptive learning rate components contribute to efficient optimization by balancing gradient accumulation with velocity adjustment. Madam introduces an extra set of parameters (k values), as well as a look-ahead mechanism to periodically update the gradient direction, thus achieving faster convergence and minimizing oscillations.

2.2 Dynamically Adjusting Gradients

In addition to Adam's momentum (m) and velocity (v), Madam introduces an additional term k, which recalibrates the gradient updates for each iteration. This term (k) helps the optimizer achieve a more refined balance between exploration and exploitation of the gradient, enhancing stability while navigating challenging loss surfaces.

2.2.1 Momentum Update

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot \Delta w_t$$

where Δw_t represents the gradient at the current step, and β_1 is the decay rate for momentum. This term controls the moving average of past gradients, smoothing oscillations and enabling Madam to maintain direction over time.

2.2.2 Calculating new parameter k

k_i is updated with a weighted average of squared gradients, similarly to the velocity term but with a more adaptive scale.

$$k_t = \beta_2 \cdot k_{t-1} + (1 - \beta_2) \cdot (\Delta w_t)^2$$

where β_2 is the decay rate for this adaptive term. Here, k_t adjusts based on the squared gradient, accounting for the changing gradient scale and mitigating issues like gradient explosions.

2.2.3 Look-Ahead Update with Variance Adjustment:

Additional adjustment is applied with the adaptive term v_t

$$v_t = \beta_1 \cdot v_{t-1} + \eta \cdot \frac{m_t}{\sqrt{k_t} + \epsilon}$$

2.3 Look-Ahead Mechanism

Madam incorporates a look-ahead mechanism, inspired by Nesterov Accelerated Gradient (NAG). This approach helps to stabilize updates and avoid overshooting when approaching minima.

2.3.1 Look-Ahead Weight Calculation:

$$w_{t_{look-ahead}} = w_t - v_t$$

This term anticipates the effect of the momentum and provides a "preview" of the next step based on past accumulated gradients.

2.3.2 Final Weights Updates:

$$w_{t+1} = w_t - \eta \cdot \frac{w_{t_{look-ahead}}}{\sqrt{k_t} + \epsilon}$$

The look-ahead weight is then combined with the adaptive term, yielding a stabilized and responsive update that can adaptively handle changes in gradient direction near optimal points.

This mechanism enables Madam to keep a balance between the momentum-driven updates and the dynamically adjusted learning rate, effectively navigating complex and non-convex loss surfaces.

2.4 Computational Complexity and Memory Requirements

Similar to Adam, Madam remains computationally efficient while introducing little additional memory requirements to store the modified terms k_t and v_t. These additions are minimal and keep the overall memory usage within practical limits, making Madam suitable for real-world deep learning applications without significant hardware constraints.

3. EXPERIMENTS AND RESULTS

3.1 Testing on a simple convex function

To assess the performance of the Madam optimizer, we conducted experiments comparing it aga3inst the original Adam optimizer on a simple convex function $f(x, y) = x^2 + y^2$. The primary objective was to evaluate Madam's convergence rate and stability, using both the final loss values and the trajectory taken by each optimizer.

3.1.1 Experiment Setup

- Objective Function: $f(x, y) = x^2 + y^2$, a convex function where the minimum lies at the origin.

Optimizers: We implemented both the original Adam optimizer and the modified Madam optimizer, using the following parameters:

- **Step Size (Learning Rate)**: α=0.02
- **Momentum Terms**: β_1=0.8, β_2=0.999
- **Iterations**: epochs = 60

3.1.2 Results and Analysis

Contour Plot of Optimizer Paths:

We plotted the paths of both optimizers on a filled contour plot of the objective function to observe how each algorithm navigates the search space. The trajectory taken by each optimizer from its initial position towards the minimum provides insight into the stability and efficiency of the updates.

Loss over Iterations:

In addition, we tracked the loss values across the iterations for both Adam and Madam optimizers. This plot allows us to assess the convergence rate of each optimizer and to determine whether Madam provides a smoother or more rapid approach to the minimum.

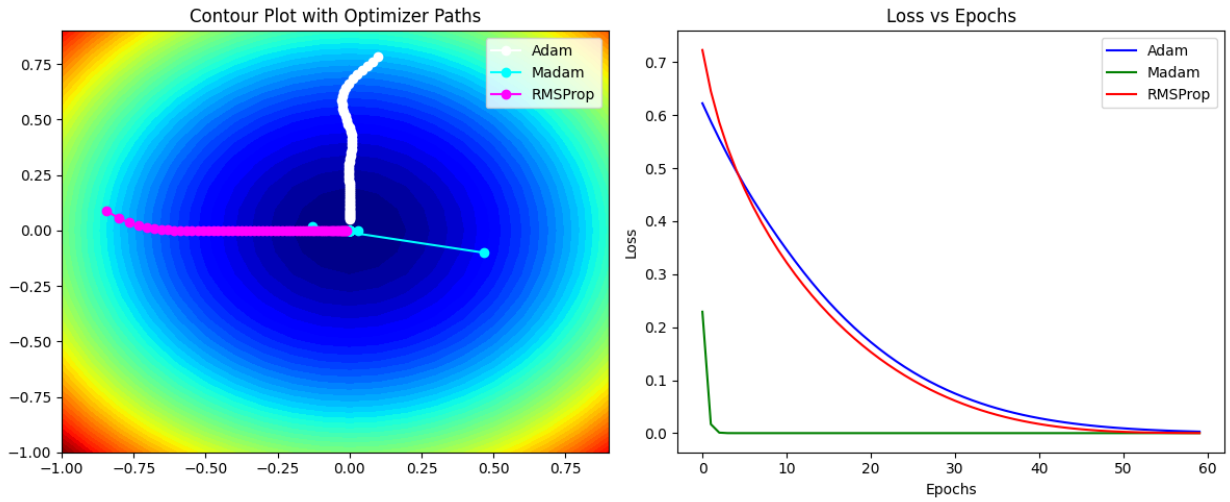


Fig.1 Contour Plots and Loss vs epochs graphs of Madam, Adam and RMSprop

- Final loss (Adam): 0.00278
- Final loss (Madam): 0.00000
- Final loss (RMSProp): 0.00014

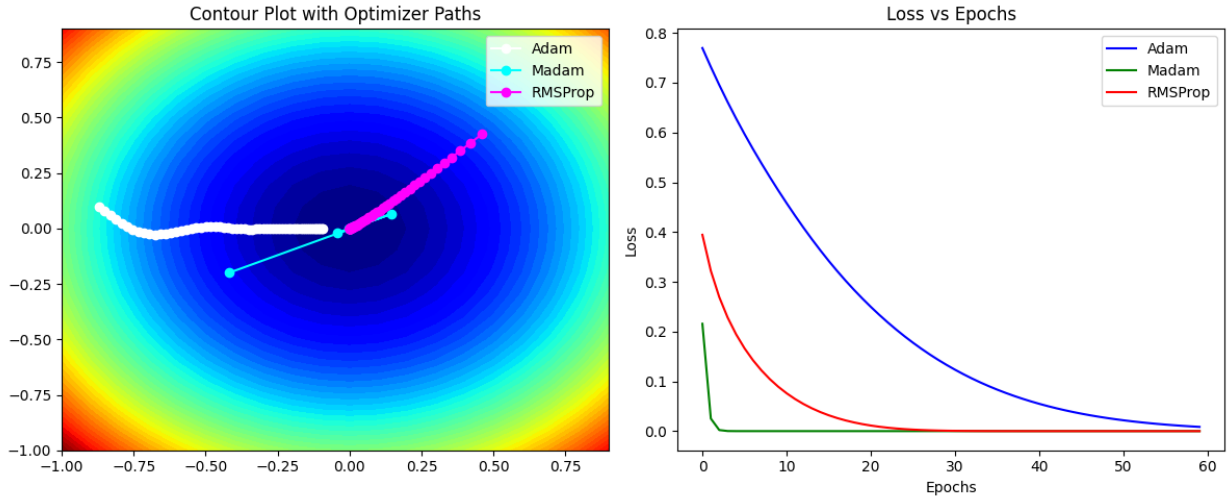


Fig.2 Contour Plots and Loss vs epochs graphs of Madam, Adam and RMSprop

- Final loss (Adam): 0.00851
- Final loss (Madam): 0.00000
- Final loss (RMSProp): 0.00000

3.2 Benchmarks on real world dataset

Let's extend the evaluation of the Madam optimizer to a real-world dataset, comparing its performance against the original Adam optimizer and other optimizers in terms of

convergence speed, stability, and final accuracy. By applying the optimizers to a practical problem, we aim to gain insights into how Madam adapts to more complex data distributions and gradients.

3.2.1 TRAINING ON REGRESSION

For regression task we will use a famous regression dataset like the California Housing dataset and calculate metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), and R^2 score after **20 epochs**

Table 1. Loss, MAE and R^2 score during training on California Housing dataset

optimizer	Loss	MAE	R^2 score
Madam	0.2711	0.3566	0.7972
Adam	0.2825	0.3669	0.7887
RMSprop	0.2748	0.3595	0.7945

Table 2. Loss, MAE and R^2 score during validation on California Housing dataset

optimizer	Loss	MAE	R^2 score
Madam	0.2867	0.3687	0.7808
Adam	0.2886	0.3657	0.7793
RMSprop	0.2861	0.3645	0.7810

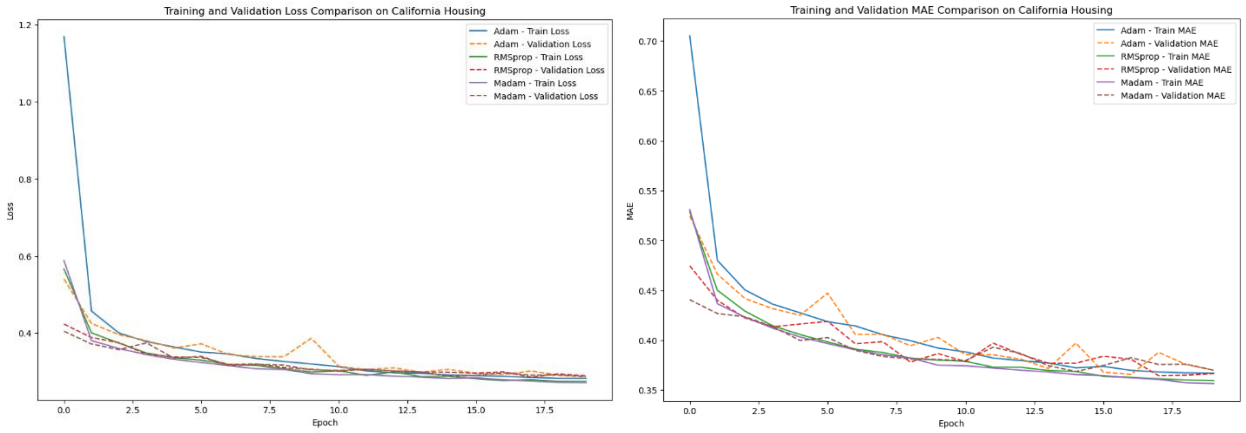
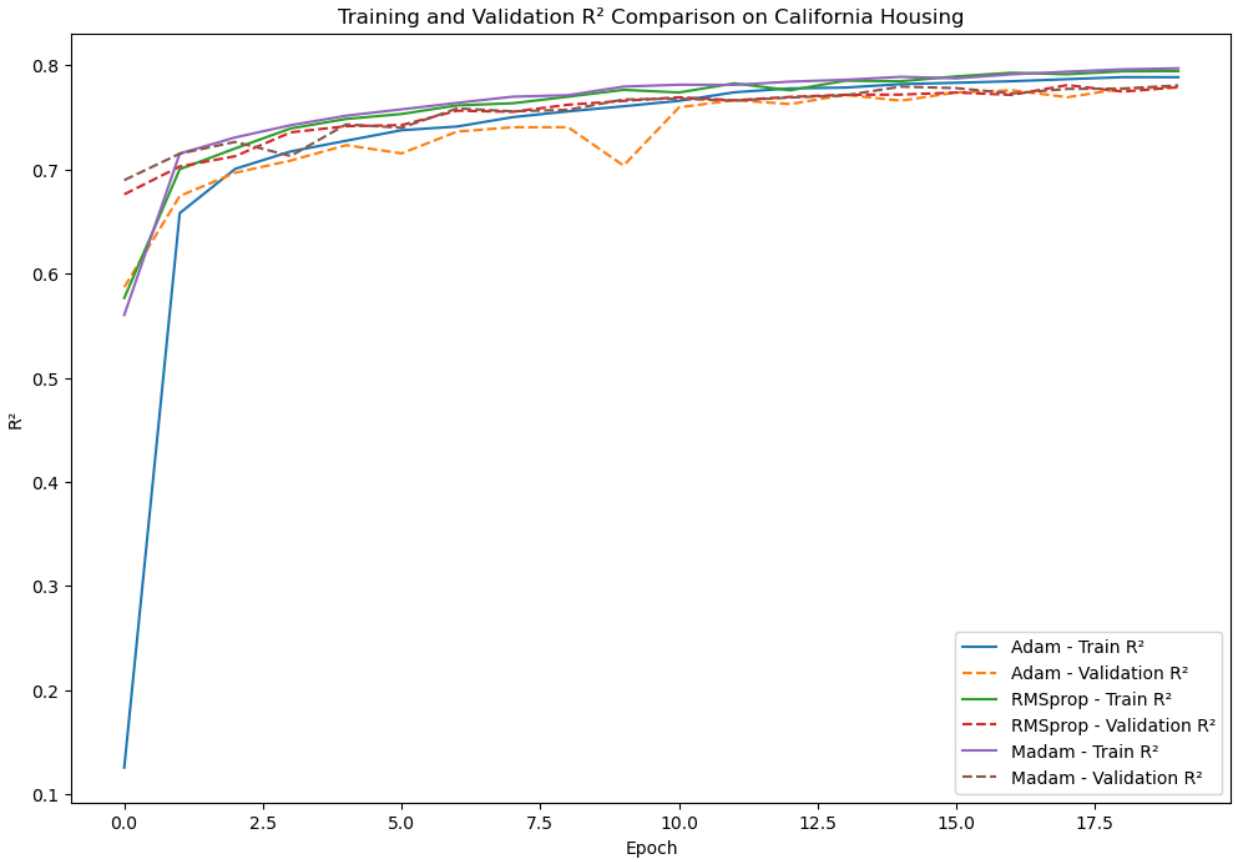


Fig. Loss vs epochs and MAE vs epochs graphs



3.2.2 TRAINING ON MNIST DATASET

The MNIST database is a large database of handwritten digits that is commonly used for training various image processing systems

Fig. MNIST dataset

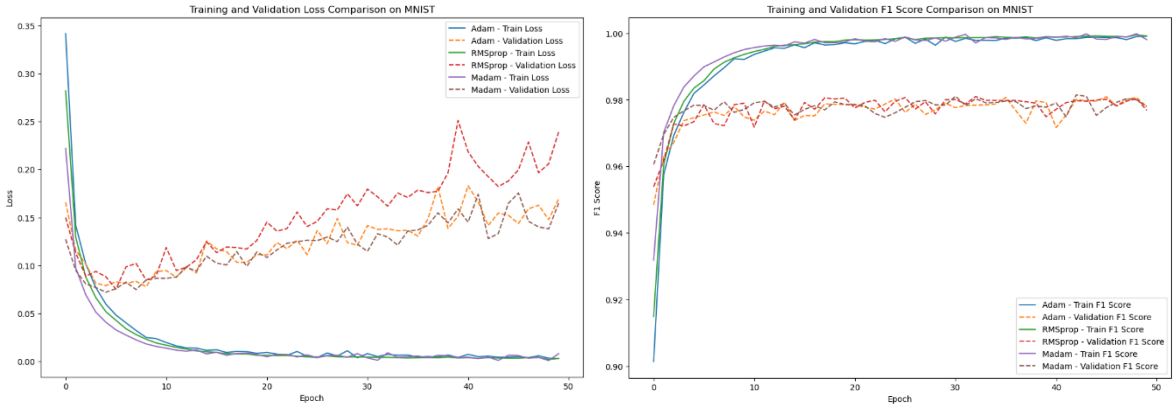
The following results were obtained by training using a simple neural network on the MNIST dataset using various optimizers: **Adam**, **RMSprop**, and **Madam** The performance was measured in terms of the minimum training loss, minimum validation loss, maximum training accuracy, maximum validation accuracy, and maximum F1 score. The training process was carried out for 50 epochs with each optimizer.

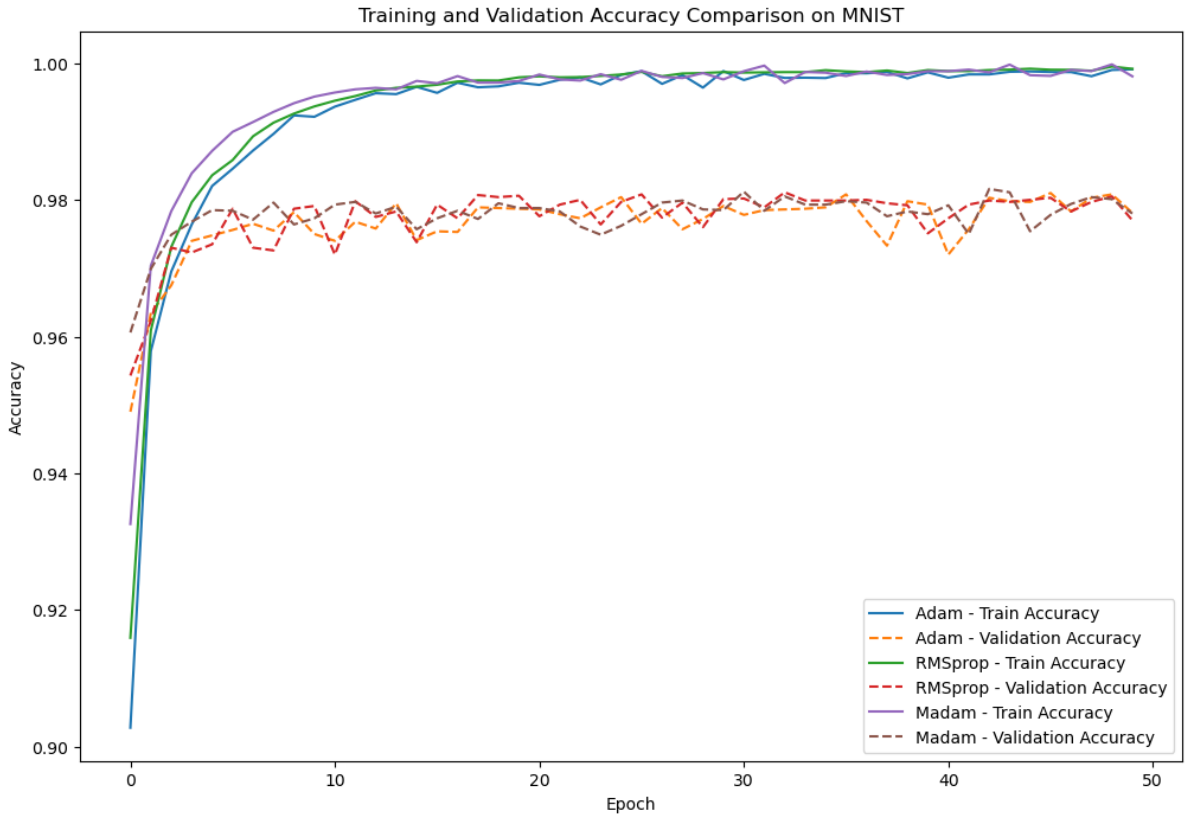
Table 1. Loss, Accuracy and F1 score during training on MNIST

optimizer	Loss	Accuracy	F1 score
Madam	0.0007	99.98%	0.9998
Adam	0.0029	99.91%	0.9991
RMSprop	0.0018	99.95%	0.9995

Table 2. Loss, Accuracy and F1 score during validation on MNIST

optimizer	Loss	Accuracy	F1 score
Madam	0.0720	98.16%	0.9815
Adam	0.0778	98.10%	0.9809
RMSprop	0.0752	98.11%	0.9810





3.2.3 TRAINING ON FASHION MNIST DATASET

Fashion-MNIST is a dataset of Zalando's article images consisting of a training set of 60,000 examples and a test set of 10,000 examples.

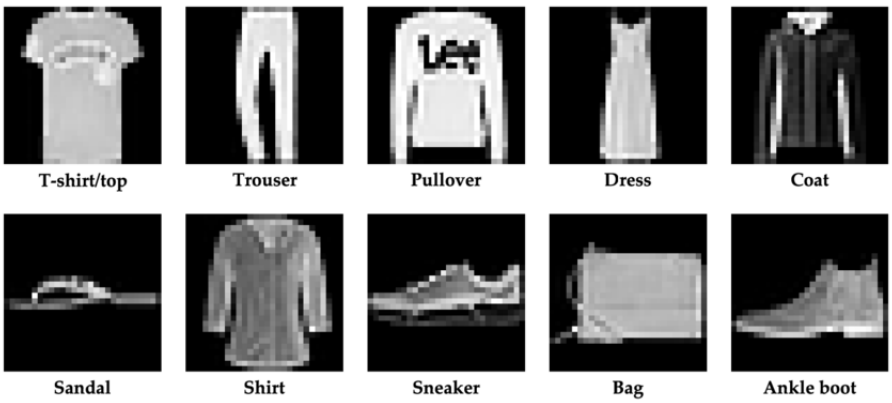


Fig. Fashion MNIST dataset

Similarly following results were obtained by training using a simple neural network on the Fashion MNIST dataset using various optimizers: **Adam**, **RMSprop**, and **Madam** The

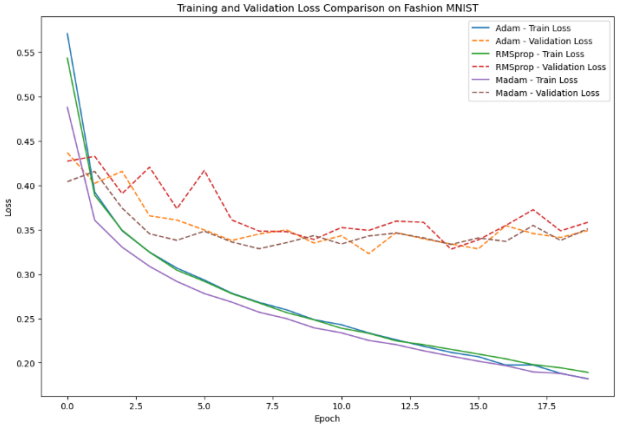
performance was measured in terms of the minimum training loss, minimum validation loss, maximum training accuracy, maximum validation accuracy, and maximum F1 score. The training process was carried out for **20 epochs** with each optimizer.

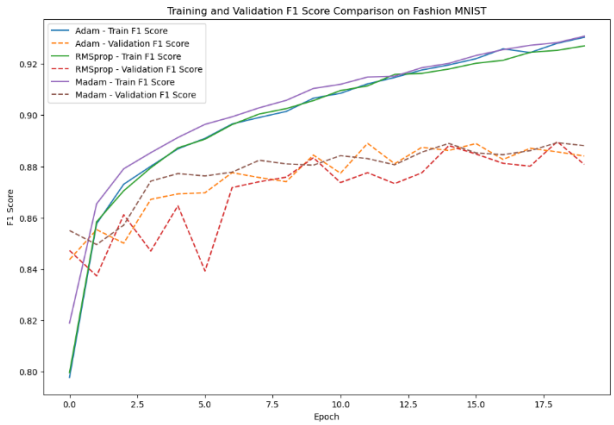
Table 1. Loss, Accuracy and F1 score during training on Fashion MNIST

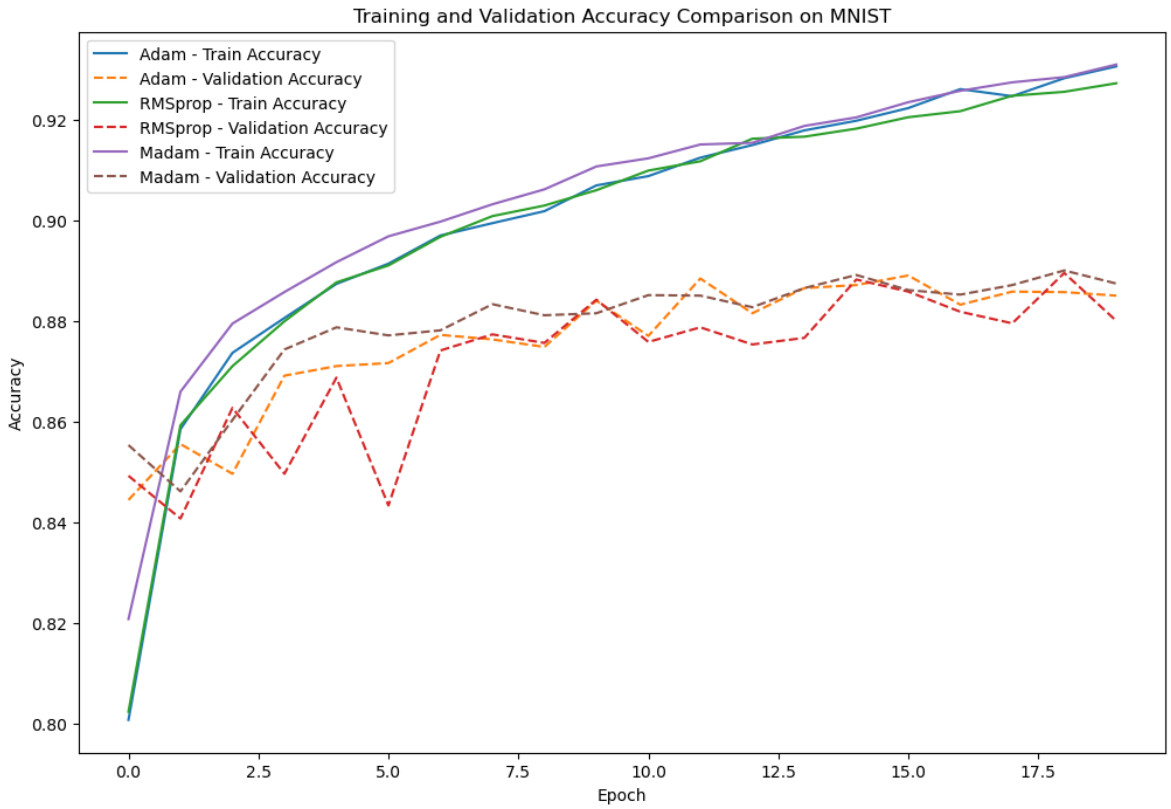
Optimizer	Loss	Accuracy	F1 score
Madam	0.1817	93.09%	0.9308
Adam	0.1818	93.05%	0.9304
RMSprop	0.1891	92.72%	0.9270

Table 2. Loss, Accuracy and F1 score during validation on Fashion MNIST

Optimizer	Loss	Accuracy	F1 score
Madam	0.3286	89.00%	0.8893
Adam	0.3231	88.90%	0.8890
RMSprop	0.3281	88.95%	0.8897







4. CONCLUSION

We have introduced a simple and computationally efficient algorithm for optimization of neural networks with look-ahead mechanism inspired by the concept of NAG (Nesterov Accelerated Gradient). In summary, the **Madam** optimizer demonstrates superior performance on the classification tasks and comparable on famous regression dataset like the California Housing dataset to RMSprop and better than Adam, showing its potential as a robust optimization technique for deep learning models.

Links for codes and notebooks:

- Code for contour plots (3.1.2): Contour plot code
- Testing loss on random regression sample data: Test on random data
- Training on MNIST: mnist notebook
- Training Fashion MNIST: Fashion MNIST notebook
- Training on California Housing dataset: California Housing Notebook

Repo Link