What is Adam Optimizer?

Last Updated : 04 Oct, 2025

Adam (Adaptive Moment Estimation) optimizer combines the advantages of Momentum and RMSprop techniques to adjust learning rates during training. It works well with large datasets and complex models because it uses memory efficiently and adapts the learning rate for each parameter automatically.

How Does Adam Work?

Adam builds upon two key concepts in optimization:

1. Momentum

Momentum is used to accelerate the gradient descent process by incorporating an exponentially weighted moving average of past gradients. This helps smooth out the trajectory of the optimization allowing the algorithm to converge faster by reducing oscillations.

The update rule with momentum is:

$$w_{t+1} = w_t - \alpha m_t$$

where:

- m_t is the moving average of the gradients at time t
- α is the learning rate
- w_t and w_{t+1} are the weights at time t and $t+1$, respectively

The momentum term m_t is updated recursively as:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \frac{\partial L}{\partial w_t}$$

where:

- β_1 is the momentum parameter (typically set to 0.9)
- $\frac{\partial L}{\partial w_t}$ is the gradient of the loss function with respect to the weights at time t

RMSprop is an adaptive learning rate method that improves upon AdaGrad. While AdaGrad accumulates squared gradients and RMSprop uses an exponentially weighted moving average of squared gradients, which helps overcome the problem of diminishing learning rates.

The update rule for RMSprop is:

$$w_{t+1} = w_t - \frac{\alpha_t}{\sqrt{v_t + \epsilon}} \frac{\partial L}{\partial w_t}$$

where:

- v_t is the exponentially weighted average of squared gradients:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \left(\frac{\partial L}{\partial w_t} \right)^2$$

- ϵ is a small constant (e.g., 10^{-8}) added to prevent division by zero

Combining Momentum and RMSprop to form Adam Optimizer

Adam optimizer combines the momentum and RMSprop techniques to provide a more balanced and efficient optimization process. The key equations governing Adam are as follows:

- **First moment (mean) estimate**:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \frac{\partial L}{\partial w_t}$$

- **Second moment (variance) estimate**:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \left(\frac{\partial L}{\partial w_t} \right)^2$$

- **Bias correction**: Since both m_t and v_t are initialized at zero, they tend to be biased toward zero, especially during the initial steps. To correct this bias, Adam computes the bias-corrected estimates:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

- **Final weight update**: The weights are then updated as:

Key Parameters

- α: The learning rate or step size (default is 0.001)
- β_1 and β_2: Decay rates for the moving averages of the gradient and squared gradient, typically set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$
- ϵ: A small positive constant (e.g., 10^{-8}) used to avoid division by zero when computing the final update

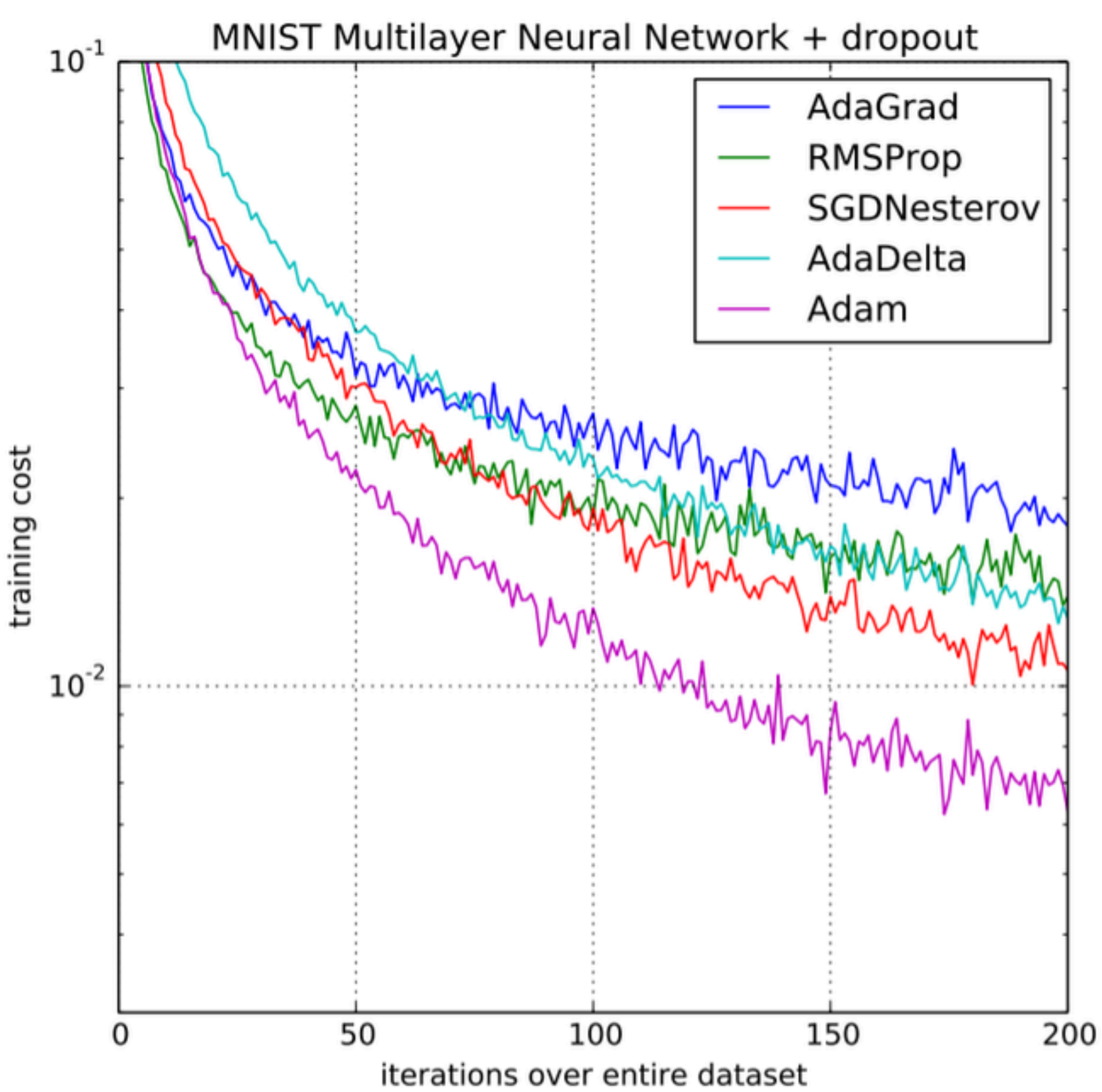
Why Adam Works So Well?

Adam addresses several challenges of gradient descent optimization:

- **Dynamic learning rates**: Each parameter has its own adaptive learning rate based on past gradients and their magnitudes. This helps the optimizer avoid oscillations and get past local minima more effectively.
- **Bias correction**: By adjusting for the initial bias when the first and second moment estimates are close to zero helping to prevent early-stage instability.
- **Efficient performance**: Adam typically requires fewer hyperparameter tuning adjustments compared to other optimization algorithms like SGD making it a more convenient choice for most problems.

Performance of Adam

In comparison to other optimizers like SGD (Stochastic Gradient Descent) and momentum-based SGD, Adam outperforms them significantly in terms of both training time and convergence accuracy. Its ability to adjust the learning rate per parameter combined with the bias-correction mechanism leading to faster convergence and more stable optimization. This makes Adam especially useful in complex models with large datasets as it avoids slow convergence and instability while reaching the global minimum.



Performance Comparison on Training cost

In practice, Adam often achieves superior results with minimal tuning, making it a go-to optimizer for deep learning tasks.

What does the Adam optimizer primarily combine?

A Momentum and Gradient Descent

B RMSProp and Gradient Descent

C Momentum and RMSProp

D Stochastic Gradient Descent and Adagrad

Login to View Explanation

1/6

< Previous Next >

Comment

P prakha... + Follow 31

Article Tags : Deep Learning AI-ML-DS Neural Network AI-ML-DS With Python

Explore

Deep Learning Basics

Neural Networks Basics

Deep Learning Models

Deep Learning Frameworks

Model Evaluation

Deep Learning Projects

(201305)

Registered Address:
K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

Advertise with us
GFG Corporate Solution
Campus Training Program

DevOps
CS Core Subjects
Interview Preparation
Software and Tools

DevOps & Cloud
GATE
Trending Technologies

Web Development
Data Science
CS Subjects

System Design