Adam Optimizer Derivation

Abdul Samad Khan

May 2025

1 Backpropagation Derivation

Let a neural network have layers $l = 1, \ldots, L$, with weights $W^{[l]}$, biases $b^{[l]}$, and activation $a^{[l]} = \sigma(z^{[l]})$, where $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$. For a loss function $\mathcal{L}(y, \hat{y})$, backpropagation computes gradients $\frac{\partial \mathcal{L}}{\partial W^{[l]}}$ and $\frac{\partial \mathcal{L}}{\partial b^{[l]}}$.

For the output layer ($l = L$), let $\delta^{[L]} = \frac{\partial \mathcal{L}}{\partial z^{[L]}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \sigma'(z^{[L]})$. For regression with MSE loss, $\mathcal{L} = \frac{1}{2}(y - \hat{y})^2$, so $\frac{\partial \mathcal{L}}{\partial \hat{y}} = (\hat{y} - y)$.

For hidden layers, recursively compute:

$$\delta^{[l]} = (W^{[l+1]})^T \delta^{[l+1]} \cdot \sigma'(z^{[l]}), \quad \frac{\partial \mathcal{L}}{\partial W^{[l]}} = \delta^{[l]}(a^{[l-1]})^T, \quad \frac{\partial \mathcal{L}}{\partial b^{[l]}} = \delta^{[l]}.$$

2 Adam Optimizer

Adam combines momentum and RMSProp. Let θ be parameters, $g_t = \nabla_\theta \mathcal{L}_t$ the gradient at step t, and hyperparameters α (step size), $\beta_1, \beta_2 \in [0, 1)$, ϵ (small constant).

First moment (mean):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t.$$

Second moment (uncentered variance):

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2.$$

Bias correction:

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

Update rule:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}.$$

3 Relevance to Finance

Adam's adaptive learning rate handles noisy financial data (e.g., futures prices) by stabilizing updates via momentum and variance normalization, crucial for training transformer-GNN models on volatile time series.