OPTIMIZER TRONG DEEP LEARNING

Thuật toán tối ưu là phương pháp điều chỉnh các tham số (weights, bias) của mô hình sao cho hàm mất mát (loss function) được giảm thiểu.

NGƯỜI TRÌNH BÀY
Phạm Công Minh

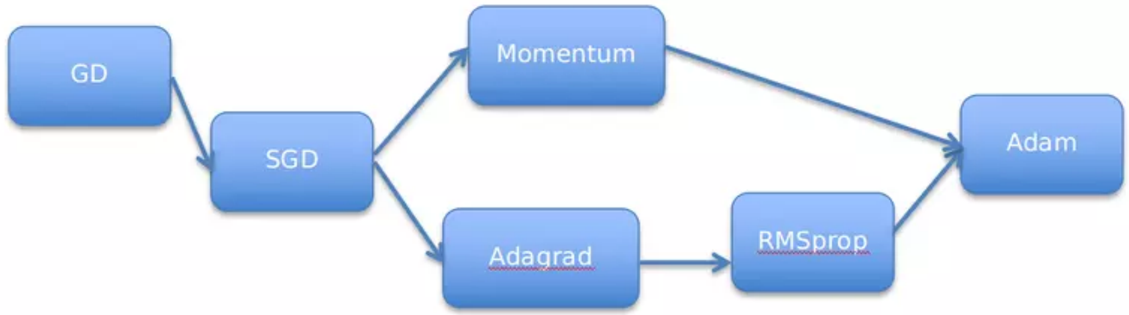
MENTOR
Phạm Đình Hải

MỤC LỤC

Tầm quan trọng và sơ đồ phát triển của thuật toán tối ưu

Thuật toán tối ưu đóng vai trò xương sống trong huấn luyện mô hình học sâu.

- Quyết định tốc độ học
- Ảnh hưởng đến chất lượng nghiệm tìm được
- Cân bằng hiệu suất và khả năng tổng quát hóa
- Xử lý dữ liệu phức tạp và sparse



GD và SGD Algorithm

GD(Gradient descent)

Cần tính toán giá trị hàm loss trên toàn bộ điểm dữ liệu → gây chậm, tốn bộ nhớ, không phù hợp cho bài toán online learning

SGD(Stochastic Gradient Descent)

Thay vì tính giá trị hàm loss trên toàn bộ dữ liệu, thuật toán sẽ chia tập dữ liệu thành các batchsize nhỏ hơn, sau đó tính toán hàm loss và cập nhật trọng số, phù hợp với online learning

$$\theta_{t+1} = \theta_t - \eta \nabla L(\theta_t)$$

Nhược điểm chung:

- Đều nhạy cảm với learning rate → dễ chậm hoặc không hội tụ
- Dễ mắc kẹt ở local minimum
- Không có cơ chế thích ứng (adaptive learning rate) cho từng tham số.
- Không nhớ hướng cũ → dễ bị kẹt hoặc dao động.

MOMENTUM ALGORITHM

Là thuật toán mở rộng của GD giúp vươn tới những điểm global minimum nhờ có thêm “quán tính”

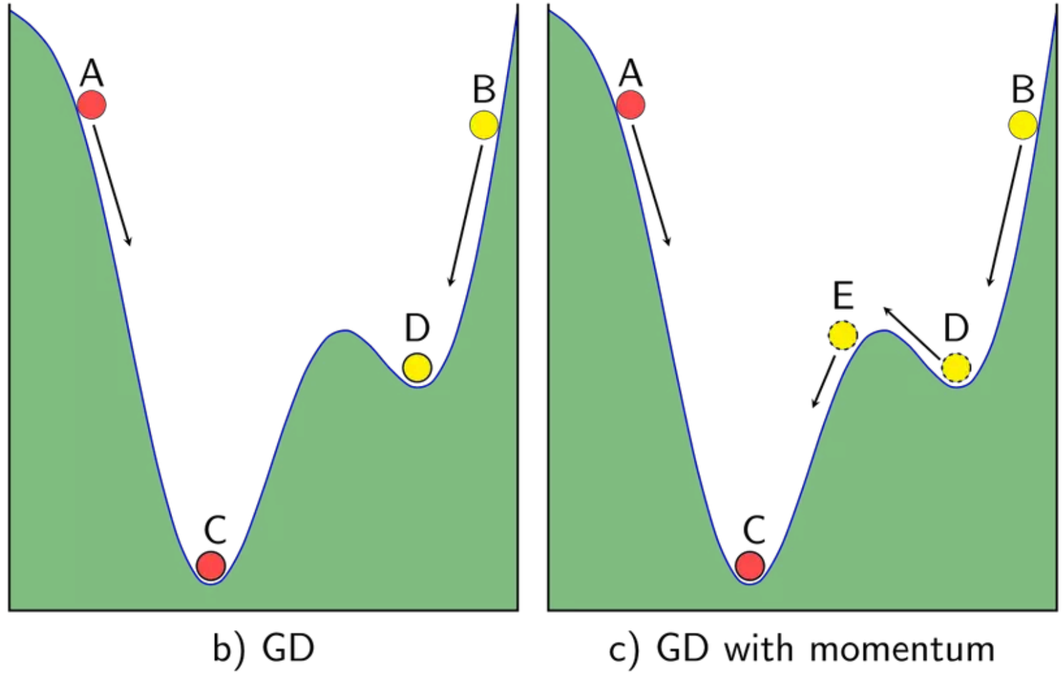
$$v_t = \beta v_{t-1} + (1-\beta)\nabla L(\theta_{t-1})$$
$$\theta_t = \theta_{t-1} - \eta v_t$$

Ưu điểm:

- Nhanh hội tụ, không bị mắc kẹt ở local minimum

Nhược điểm:

- Dễ bị giao động quanh điểm global minimum do có “quán tính”



b) GD c) GD with momentum

ADAGRAD ALGORITHM

Là một thuật toán tối ưu mở rộng từ SGD, có khả năng tự điều chỉnh learning rate cho từng tham số riêng lẻ

- Lưu trữ tổng bình phương gradient của từng tham số:

$$G_t = G_{t-1} + g_t^2$$

- Cập nhật tham số:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t} + \epsilon} \cdot g_t$$

Ưu điểm :

- Tự điều chỉnh tốc độ học cho từng tham số
- Thích hợp với dữ liệu sparse (ví dụ NLP, recommendation)

Nhược điểm

- Learning rate giảm quá nhanh → mô hình có thể dừng học sớm, không hội tụ tới global minimum.
- Không tích hợp momentum → hội tụ chậm ở một số hàm mất mát phức tạp.

RMSPROP ALGORITHM

Là thuật toán được sinh ra để khắc phục nhược điểm giảm learning rate quá nhanh của Adagrad.
Sử dụng moving average (trung bình động) để tránh việc learning rate trở nên giảm quá nhanh như Adagrad

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho) g_t^2$$

Ưu điểm:

- Điều chỉnh tham số learning rate riêng cho từng tham số
- Tránh việc learning rate giảm mạnh → học được lâu dài

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Nhược điểm:

- Chưa tích hợp momentum
- Phụ thuộc vào việc chọn tham số ρ

ADAM ALGORITHM

Là một thuật toán tối ưu phổ biến trong deep learning , kết hợp của Rmsprop và Momentum

- Momentum → tích lũy quán tính của gradient.
- RMSprop → điều chỉnh learning rate theo moving average của gradient bình phương.

→ Adam giúp hội tụ nhanh, ổn định, ít cần tuning learning rate, phù hợp với dữ liệu lớn và phi tuyến.

Nhược điểm:

- Phụ thuộc vào hyperparameter
- Tốn bộ nhớ hơn do phải lưu moment và RMS

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

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

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

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

ADAMW ALGORITHM

Là thuật toán cải tiến hơn của Adam khi tách riêng Weight decay(W)

Weight Decay là một kỹ thuật regularization trong machine learning/deep learning.

Mục tiêu:

- Giảm overfitting bằng cách trừng phạt các trọng số lớn
- Ổn định quá trình học

Cách làm thông thường:

- Thêm một hàm phạt L2 vào loss function

$$L'(\theta) = L(\theta) + \frac{\lambda}{2}\|\theta\|^2$$

Mất đi ý nghĩa của weight decay do quá trình tính toán gradient descent

→ AdamW ra đời tách riêng Weight decay $\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} - \eta\lambda\theta_t$

Ưu điểm:

- Giúp huấn luyện ổn định và kết quả tốt hơn so với Adam + L2.

Nhược điểm:

- Phụ thuộc vào hyperparameter
- Tốn kém bộ nhớ

CODE + KẾT QUẢ