# ĐỒ ÁN MÔN HỌC

## Abstract

## Introduction

## Optimize Algorithm

1. Gradient Descent
   1. Ý tưởng chính của Gradient Descent

Trong lĩnh vực Machine Learning và Toán Tối ưu nói chung, việc tìm giá trị nhỏ nhất (hoặc lớn nhất của hàm số) là rất quan trọng . Gradient Descent là một trong những thuật toán tối ưu quan trọng và phổ biến nhất trong lĩnh vực *Machine Learning* và *Deep Learning.* Mục tiêu quan trọng của Gradient Descent là tìm giá trị tham số tối ưu của hàm mất mát sao cho hàm mất mát đạt giá trị nhỏ nhất.

Ta gọi:

Để hàm mất mát đạt cực tiểu tại điểm , độ dốc dạo hàm của hàm mất mát bằng không:

Tuy nhiên, hàm số trong thực tế rất phức tạp, phi tuyến, nhiều chiều, thường rất khó để tìm ra nghiệm tối ưu nhất cho bài toán. Do đó, ý tưởng chính để giải quyết bài toán này là tìm một phương pháp xấp xỉ hàm - sử dụng những phương trình đơn giản, dễ đạo hàm với giả định rằng sau một thời gian *t* cập nhật giá trị, điểm sẽ xấp xỉ ).

Để xem xét về hướng đi của bước để đưa x về điểm tối ưu, ta khai triển chuỗi Taylor bậc 1 xấp xỉ hàm xung quanh điểm , ta có được:

*với mọi nằm gần*

Sau khi đã xấp xỉ bằng biểu thức khai triển Taylor trên, ý tưởng chính của Gradient Descent là di chuyển sao cho hàm được tối thiểu hóa. Dựa vào biểu thức xấp xỉ trên, bằng cách di chuyển bằng một đại lượng > 0 theo hướng ngược hướng với đạo hàm , ta biểu diễn được bằng công thức sau:

Ta gọi là điểm khởi tạo ban đầu, tham số được xác địng là bước nhảy *(learning rate)*

* 1. *L-smooth* functions (hàm *L-smooth*)

Để có thể đưa ra được tốc độ hội tụ cụ thể cho thuật toán Gradient Descent, ta đặt ra giới hạn (bound) về mức độ thay đổi của gradient khi ta di chuyển một chút theo bất kỳ hướng nào. Khi ta bắt đầu di chuyển điểm theo một hướng ngược hướng với đạo hàm từ , giá trị gradient mới sẽ sẽ không thay đổi quá nhanh so với , từ đó giữ cho luôn là một đại lượng độ dốc giúp hàm hội tụ ổn định trong 1 khoảng thời gian

Giả sử, độ dốc đạo hàm là liên tục 𝐿-Lipschitz với hằng số , điều kiện này trên lý thuyết thường được gọi là *L-smoothness*. Xét hàm khả vi trong miền lồi . Hàm được gọi là *L-smoothness* nếu đạo hàm của nó là liên tục 𝐿-Lipschitz, ta có:

(1)

Hệ quả quan trọng của *L-smoothness* được phát biểu rằng: giá trị của hàm có thể bị thay đổi, và bị chặn trên bởi một biểu thức bậc 2:

Biểu thức trên có thể được chứng minh như sau:

Với bất đẳng thức Cauchy-Schwarz, ta có:

Xét khoảng cách giữa điểm hiện tại so với x, sau khi qua thời điểm:

Từ định nghĩa (1), ta có:

Thay vào (2), ta có:

Để chứng minh mỗi bước Gradient Descent đều làm giảm giá trị hàm mục tiêu , ta xét điểm cập nhật tiếp theo với bước cập nhật . Từ , ta có:

Chọn bước nhảy

từ ta có bổ đề Gradient Descent:

Ta thấy nếu là *L-smooth*, với giá trị , tại mỗi bước của Gradient Descent luôn đảm bảo gradient của hàm số luôn giảm với ít nhất một lượng tỉ lệ với bình phương độ lớn gradient nếu gradient của hàm số hiện tại khác 0.

* 1. Convergence in convex function value: The Euclidean mirror descent lemma

Những chứng minh trên đảm bảo gradient của hàm mục tiêu luôn giảm, điều này không đồng nghĩa với việc giá trị của hàm mục tiêu cũng giảm xuống giá trị cực tiểu. Việc gradient của hàm mục tiêu rất nhỏ sau khi thực hiện nhiều bước cập nhật cho thấy thuật toán hội tụ về điểm tối ưu cục bộ (local optima), không đồng nghĩa với việc đạt điểm tối ưu toàn cục (global optimum).

* + - *Bổ đề “Mirror Descent Euclid” (Euclidean Mirror Descent Lemma)*

Giả sử : là hàm lồi (convex) và khả vi. Với bất kì bước nhảy 𝜂, hai điểm được cập nhật bởi gradient descent đều thỏa:

Proof (Chứng minh lemma trên):

Nếu là hàm lồi, ta có được bất đẳng thức sau (hệ quả ko cần chứng minh):

Từ bất đẳng thức trên, ta có:

Từ (7), ta có:

* + - *Telescoping:*

Theo định nghĩa của Gradient Descent:

Theo bổ đề “Mirror Descent Euclid”, ta có thể thay phần với , bổ đề thành:

(8)

Giả sử là *L-smooth* và , ta có thể dùng bổ đề Gradient Descent để chặn:

(9)

Từ (8) và (9), ta có hệ quả sau:

Thực tế, có tham số tối ưu để hàm đạt giá trị nhỏ nhất, khi :

Theo bổ đề Gradient Descent, 𝑓() không thay đổi theo , và 𝑓() càng nhỏ sau

T vòng lặp, nên:

Ta có thể viết như sau:

### SGD

* 1. Phương pháp ngẫu nhiên (Stochastic Method)

Ở trong hầu hết các ứng dụng Machine Learning, hàm mục tiêu (object function) thường được tạo thành bởi tổng của các điểm dữ liệu, do đó việc tính toán đạo hàm (gradient) cho các tham số sẽ rất tốn kém về thời gian và tài nguyên.

Để áp linh hoạt và tiết kiệm tài nguyên hơn, thay vì áp dụng phương pháp tính chính xác, ta thay thế bằng phép toán tiết kiệm hơn, nhưng không làm chênh lệch khỏi điểm cực tiểu quá nhiều. Cách tiếp cận này dựa trên tính ngẫu nhiên (stochasticity) dẫn tới các họ thuật toán này gọi là *stochastic gradient descent.*

*Stochastic gradient descent* đánh đổi giữa đại lượng gradient chính xác (exact gradient) với chi phí tính toán rẻ hơn, kết quả di chuyển bằng những bước ngẫu nhiên, thất thường (erratic) thay vì các bước giảm đều như Gradient Descent. Những đánh đổi này là hoàn toàn hợp lý vì những yếu tố sau đây:

* Trong thực tiễn, cùng một khoảng thời gian, thuật toán stochastic gradient descent có thể thực hiện nhiều bước hơn thuật toán gradient descent truyền thống. Trong một số trường hợp, Gradient Descent trên toàn bộ dữ liệu sẽ không thực hiện được bởi vì giới hạn tài nguyên tính toán cho phép. Vì vậy, Stochastic Gradient Descent (SGD), với phương pháp ngẫu nhiên, là một sự lựa chọn thay thế, tuy không hoàn hảo, những vẫn có được tốc độ và độ chính xác gần như chính xác.
* Ở một khía cạnh khác trong việc thiết lập các bài toán Machine Learning trong thực nghiệm, việc thực hiện các bước ngẫu nhiên (thay vì sử dụng phương pháp tính chính xác như Gradient Descent) có thể cho ra kết quả tốt hơn, ngay cả khi các thuật toán chính xác có tốc độ thực thi nhanh và chạy được trong môi trường tài nguyên cho phép. Đối với các bài toán Machine Learning, cực tiểu hóa hàm mục tiêu trên tập huấn luyện là quan trọng nhất, tuy nhiên không đồng nghĩa với việc tìm ra mô hình tốt nhất cho bài toán thực tế. Nói cách khác, vai trò của tối ưu hóa trong Machine Learning là (i) tạo ra mô hình khớp tốt với dữ liệu của tập huấn luyện (train set), nhưng đồng thời (ii) không được quá fit (overfit) khi đánh giá trên tập thử nghiệm (test set), có nghĩa là tổng quát hóa tốt với các ví dụ mới (cùng phân phối với tập huấn luyện). Việc sử dụng SGD được chứng minh là giúp giảm hiện tượng overfitting và tăng khả năng tổng quát hóa, cho phép thuật toán thoát khỏi các điểm cực tiểu cục bộ (local minima) và tiến gần hơn đến điểm cực tiểu toàn phần (global minima).
  1. Vấn đề ERM trong Machine Learning

Tất cả các loại bài toán trong Machine Learning có thể được mô hình hóa thành các định nghĩa như sau:

* Cho tập dataset có examples (điểm dữ liệu) , với là vector đặc trưng và là nhãn của điểm dữ liệu.
* Gọi mô hình là một hàm ánh xạ từ không gian đặc trưng đầu vào sang không gian đầu ra dự đoán : ℝⁿ → ℝᵐ. Mô hình được được xác định bởi bộ tham số , với mục tiêu tìm ra bộ tham số θ tối ưu khớp tốt nhất với dữ liệu huấn luyện. Ta làm được điều này bằng cách cực tiểu hóa hàm mất mát của mô hình, được định nghĩa như sau:
* Trong đó:
  + ℓ : ℝ𝑚 × ℝ là hàm mất mát, đo độ khác nhau giữa giá trị mà mô hình dự đoán trên điểm dữ liệu () và nhãn của nó ()
  1. Stochastic Gradient Descent

Hàm mất mát được định nghĩa ở phần trên là trung bình mất mát trên điểm dữ liệu của tập huấn luyện. Khi tập dữ liệu lớn, việc tính toán đạo hàm của hàm mất mát  tại mỗi lần lặp (sử dụng thuật toán Gradient Descent) có thể tốn rất nhiều chi phí tính toán vì phải tính đạo hàm trên tất cả k điểm dữ liệu. Do đó, ta có thể thay thế ước lượng đạo hàm chính xác với xấp xỉ ước lượng đạo hàm chính xác của nó, thỏa mãn:

(5)

Do đó, thuật toán Stochastic Gradient Descent (SGD) được viết dưới dạng:

(6)

Lựa chọn các biến thể khác nhau của SGD dẫn đến lựa chọn hàm mất mát khác nhau cho thuật toán SGD. Biến thể được sử dụng nhiều nhất đó là: lấy trung bình đạo hàm của 1 tập nhỏ ngẫu nhiên các điểm dữ liệu trong dữ liệu huấn luyện, chiến lược này được gọi là mini-batch.

Tính toán đạo hàm trên mini-batch một cách ngẫu nhiên thường dễ thực hiện bằng cách sử dụng phép xáo trộn ngẫu nhiên trên tập dữ liệu. Mỗi lần lặp, tính toán đạo hàm trên mini-batch gồm *b* điểm dữ liệu. Thông thường 𝑏 = 32 or 𝑏 = 64.

* 1. Analysis of Stochastic Gradient Descent

Khi phân tích các biến thể của thuật toán Gradient Descent, các nhà nghiên cứu thường dựa vào hai lemma cơ bản về hạ dốc đó là “Gradient descent lemma” và “(Euclidean) mirror descent lemma”

Stochastic gradient descent lemma

Xét hàm *L-smooth* : ℝⁿ → ℝ có giới hạn trên bậc hai:

Xét điểm tiếp theo :

*(Chứng minh tương tự 1b)*

*(Từ 6)*

Xét là bước nhảy ngẫu nhiên đến vị trí tiếp theo, hai điểm lặp liên tiếp được thuật toán stochastic gradient descent cập nhật ra thỏa mãn:

Phần điều khiển sự ảnh hưởng của hàm bậc 2 phía trước, hay còn được gọi là variance của độ dốc ngẫu nhiên

Convergence in gradient norm

Giả sử được chặn bởi một giá trị G tại mọi thời điểm , gradient descent lemma sẽ được viết lại như sau:

Sử dụng khái niệm telescope sum trên đoạn t đến T, ta được:

Giả sử bị chặn dưới bởi một giá trị rất nhỏ: , từ đó, ta có:

Chia T cho cả hai vế, ta được:

Xét:

* , ta thấy phương trình phụ thuộc T, tỉ lệ nghịch với độ tăng của
* , ta thấy phương trình tỉ lệ thuận với độ tăng của

Để đảm bảo hàm mất mát giảm dần theo T, ta cần chọn để làm cho vế phải nhỏ nhất có thể, ta có:

với C là một hằng số bất kỳ

Xét , ta có:

Bằng việc , sau T lần lặp, chắc chắn sẽ có ít nhất một lần , hàm số sẽ hội tụ mà không quan tâm đến variance của hàm số.

Stochastic Euclidean mirror descent lemma

Giả sử hàm mục tiêu là hàm *lồi,* ta cóchặn dưới:

,

Với từ (5), ta có được:

từ (5)

Từ (7), ta có bất đẳng thức sau:

Từ những chứng minh trên, ta có được bổ đề Stochastic Euclidean mirror descent:

Convergence in convex function value

### Adagrad

### Momentum

### RMSprops

### Adam

### AdamW

## Experiment