TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

A red and blue logo

Description automatically generated

**BÁO CÁO CUỐI KÌ**

**MÔN HỌC: NHẬP MÔN HỌC MÁY**

**Mã môn học: 503044**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

A red and blue logo

Description automatically generated

**BÁO CÁO CUỐI KÌ**

**MÔN HỌC: NHẬP MÔN HỌC MÁY**

*Người hướng dẫn*: **PGS.TS.Lê Anh Cường**

*Người thực hiện*:

**Nguyễn Văn Anh Quân - 52100924**

NHÓM **: 01**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

# LỜI CẢM ƠN

Trong suốt quá trình học tập và rèn luyện, em đã nhận được rất nhiều sự giúp đỡ tận tình, sự quan tâm và chăm sóc từ các thầy. Bên cạnh đó, em cũng đã được các thầy truyền đạt những kiến thức, phương pháp mới và thú vị trong các môn học. Thầy không chỉ giúp chúng em có được kiến thức mà còn tạo ra niềm vui và sự thoải mái trong quá trình học.

Em xin chân thành cảm ơn sự đóng góp và tận tâm của các thầy trong suốt thời gian học tập này. Tuy em nhận thức rằng kiến thức của em vẫn còn hạn chế và em gặp nhiều khó khăn trong quá trình học, nhưng em hy vọng nhận được những góp ý từ quý thầy về bài báo cáo này để em có thể rút kinh nghiệm và cải thiện trong những môn học sắp tới.

Cuối cùng, em xin chân thành cảm ơn quý thầy vì sự hỗ trợ và đóng góp của mình. Những điều em học được từ quý thầy không chỉ mang lại kiến thức mà còn giúp em phát triển cá nhân và trở thành những người học tập tốt hơn. Em sẽ luôn trân trọng những bài học và sự hỗ trợ mà chúng em nhận được từ quý thầy.

TP Hồ Chí Minh, ngày 24 tháng 12 năm 2023

Sinh viên:

Nguyễn Văn Anh Quân – 52100924

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của thầy thầy Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 24 tháng 12 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Văn Anh Quân*

# PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

# TÓM TẮT

Đây là phần trình bày cá nhân trong dự án cuối kì của môn Nhập môn Học máy (Machine Learning) tóm tắt về 2 vấn đề được tìm hiểu nghiên cứu trong bài báo cáo lần này.

* Câu1: Tìm hiểu, so sánh các phương pháp Optimizer trong huấn luyện mô hình học máy.
* Câu2: Tìm hiểu về Continual Learning và Test Production khi xây dựng một giải pháp học máy để giải quyết một bài toán nào đó.

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_Toc154266490)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN 3](#_Toc154266491)

[TÓM TẮT 4](#_Toc154266492)

[DANH MỤC HÌNH VẼ 6](#_Toc154266493)

[CHƯƠNG 1: NỘI DUNG 1](#_Toc154266494)

[1.1 Các phương pháp Optimizer trong huấn luyện mô hình học máy 1](#_Toc154266495)

[*1.1.1 Gradient Descent* 1](#_Toc154266496)

[*1.1.1.1 Khái niệm* 1](#_Toc154266497)

[*1.1.1.2 Learning rate* 2](#_Toc154266498)

[*1.1.1.3 Batch Gradient Descent* 6](#_Toc154266499)

[*1.1.2 Stochastic Gradient Descent (SGD)* 7](#_Toc154266500)

[1.2 Tìm hiểu về Continual Learning,Test Production và xây dựng giải pháp học máy 15](#_Toc154266501)

[*1.2.1 Continual Learning* 15](#_Toc154266502)

[*1.2.1.1 Catastrophic forgetting* 15](#_Toc154266503)

[*1.2.2* *Test Production* 21](#_Toc154266504)

[*1.2.2.1* *Định nghĩa* 21](#_Toc154266505)

[*1.2.2.2* *ML Tesing Component* 22](#_Toc154266506)

[*1.2.2.3* *Software Testing và ML Testing* 23](#_Toc154266507)

[**TÀI LIỆU THAM KHẢO** 25](#_Toc154266508)

# DANH MỤC HÌNH VẼ

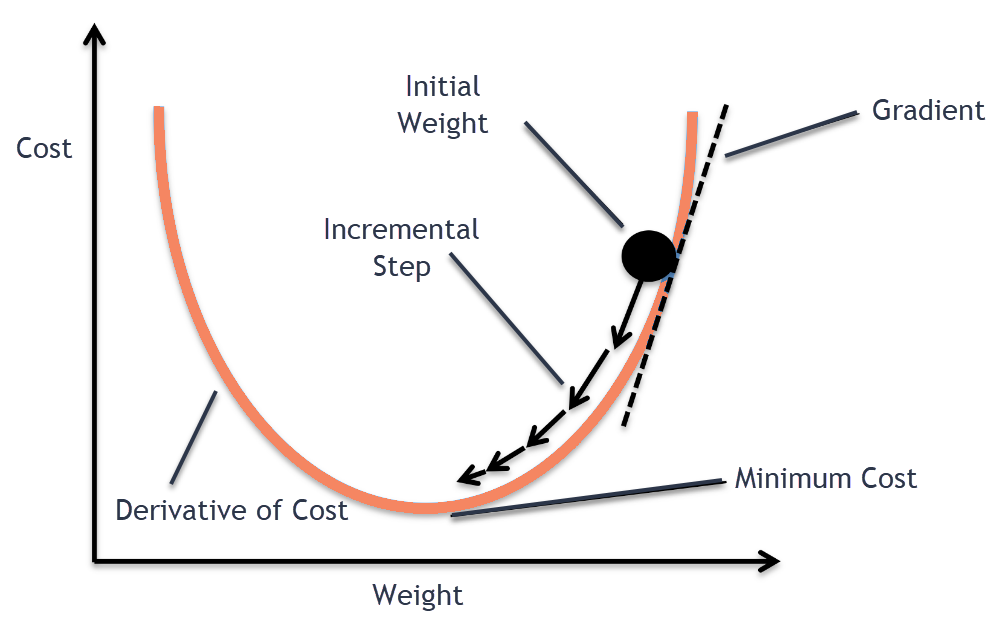
# CHƯƠNG 1: NỘI DUNG

## 1.1 Các phương pháp Optimizer trong huấn luyện mô hình học máy

### *1.1.1 Gradient Descent*

### *1.1.1.1 Khái niệm*

- Gradient Descent (GD) là một thuật toán tìm kiếm tối ưu phổ biến dùng cho các hàm số. Ý tưởng chính của GD là điều chỉnh các tham số một cách lặp lại thông qua mỗi dữ liệu huấn luyện, nhằm giảm thiểu hàm mục tiêu (hàm chi phí) đến một giá trị nhỏ nhất.



- Gradient Descent là một thuật toán tối ưu lặp (iterative optimization algorithm) được sử dụng rộng rãi trong lĩnh vực Machine Learning và Deep Learning. Thông thường, nó được áp dụng trong các bài toán tối ưu lồi (convex optimization) với mục tiêu tìm một tập hợp các biến nội tại (internal parameters) để tối ưu hóa các mô hình. Trong quá trình tối ưu, Gradient Descent tính toán và điều chỉnh các tham số theo hướng và độ lớn của đạo hàm của hàm mục tiêu (hàm chi phí). Trong đó:

* Gradient: là độ dốc của đường cong (rate of inclination or declination of a curve). Trong mặt toán học, Gradient của một hàm số là vector đạo hàm riêng của hàm số đó theo từng biến số của nó. Đối với hàm số một biến, chúng ta sử dụng thuật ngữ đạo hàm (derivative) thay cho Gradient.
* Descent: là từ viết tắt của descending, nghĩa là giảm dần.

- Gradient Descent có một số dạng khác nhau như Stochastic Gradient Descent (SGD) và Mini-batch Gradient Descent. Mặc dù có những khác biệt nhỏ, nhưng cơ bản, chúng được thực hiện theo cách sau đây:

* Khởi tạo biến nội tại.
* Đánh giá model dựa vào biến nội tại và hàm mất mát (Loss function).
* Cập nhật các biến nội tại theo hướng tối ưu hàm mất mát (finding optimal points).
* Lặp lại bước 2, 3 cho tới khi thỏa điều kiện dừng.
* Công thức cập nhật cho GD có thể được viết là:

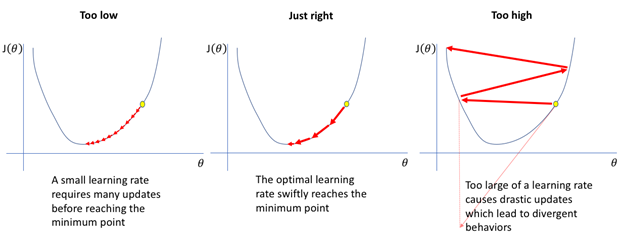
trong đó là tập các biến cần cập nhật,  là tốc độ học (learning rate),  là Gradient của hàm mất mát f theo tập θ.

### *1.1.1.2 Learning rate*

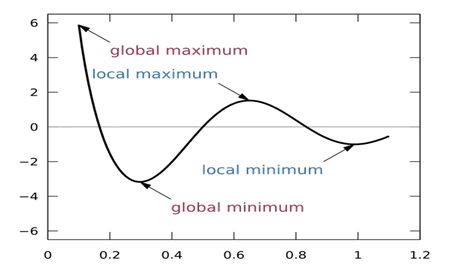
- Có một tham số quan trọng trong Gradient Descent là learning rate (tốc độ học), tương tự như độ dài bước di chuyển khi bạn leo xuống dốc. Learning rate quyết định độ lớn của mỗi bước di chuyển trong quá trình tối ưu hóa.

Nếu learning rate quá nhỏ, thuật toán sẽ di chuyển rất chậm và mất nhiều thời gian để hội tụ. Tuy nhiên, nếu learning rate quá lớn, thuật toán có thể vượt qua điểm cực tiểu và không thể hội tụ đúng kết quả mong muốn. Điều này có thể khiến thuật toán dao động hoặc không hội tụ.

Do đó, việc lựa chọn learning rate phù hợp là rất quan trọng trong Gradient Descent để đảm bảo thuật toán hội tụ nhanh chóng và đạt được kết quả tối ưu.



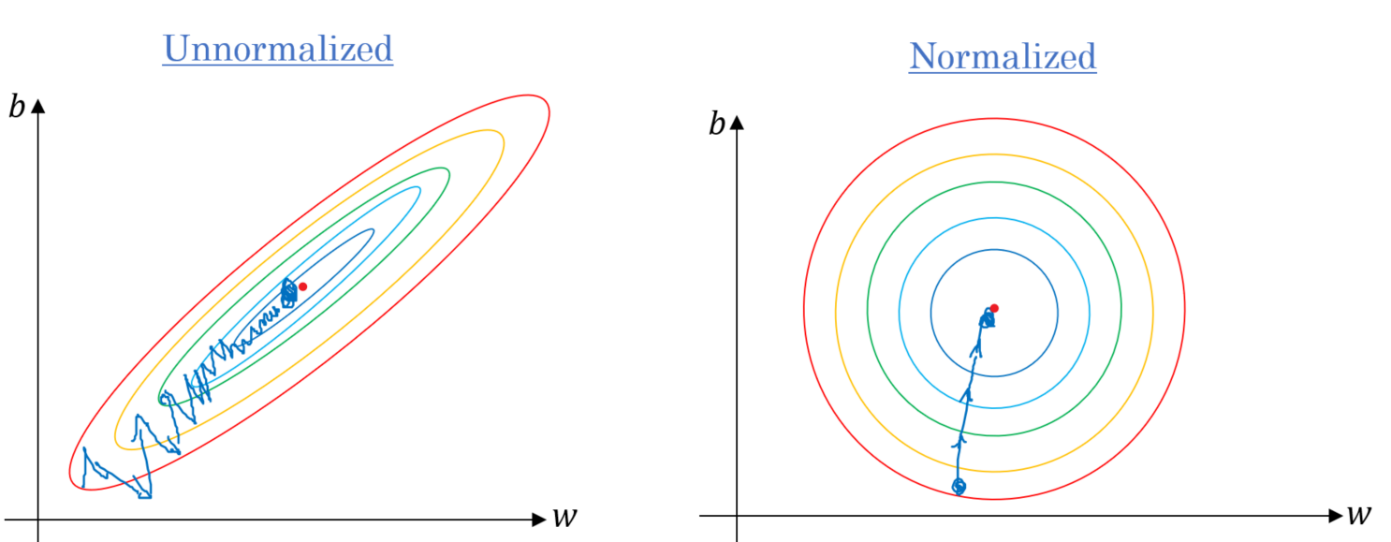
- Trong thực tế, hầu hết các hàm số không chỉ có một cực tiểu duy nhất. Thay vào đó, chúng ta gặp phải hai khái niệm quan trọng là cực tiểu cục bộ và cực tiểu toàn cục. Bạn có thể tưởng tượng chúng như các khe suy thoái hoặc các tảng đá trên một ngọn núi khi bạn leo xuống. Việc tìm ra cực tiểu trong trường hợp này trở nên khó khăn hơn. Xem hình dưới đây để biết thêm chi tiết:



* Sẽ có 2 vấn đề lúc này đối với GD:

Điểm xuất phát có thể nằm ở bên trái hoặc bên phải của đồ thị hàm số. Nếu xuất phát từ bên trái, thuật toán sẽ hội tụ tại cực tiểu địa phương (local minimum) mà không đạt được cực tiểu toàn cục (global minimum). Tương tự, nếu xuất phát từ bên phải, thuật toán sẽ mất thời gian để vượt qua vùng Plateau và đạt đến cực tiểu toàn cục, và nếu dừng thuật toán quá sớm, nó sẽ không đạt được cực tiểu toàn cục.

Trong bài trước, chúng ta sử dụng hàm chi phí MSE cho bài toán hồi quy tuyến tính, may mắn là hàm này là một hàm lồi. Điều này có nghĩa là nếu chúng ta vẽ một đường thẳng nối hai điểm bất kỳ trên đồ thị của hàm lồi, đường thẳng đó sẽ không cắt đồ thị. Điều này đồng nghĩa rằng không có cực tiểu địa phương mà chỉ có một cực tiểu toàn cục duy nhất. Hàm này cũng là một hàm liên tục và độ dốc của nó không thay đổi đột ngột. Như vậy, trong trường hợp này, Gradient Descent gặp một vấn đề, đó là nó không thể tiến gần cực tiểu toàn cục (trừ khi thời gian huấn luyện đủ lâu và learning rate đủ nhỏ).

Trên thực tế, hàm chi phí có dạng đồ thị giống một cái bát. Nếu các đặc trưng (feature) của đầu vào (phần tử của vector X) có phạm vi giá trị tương đồng, miệng bát sẽ trở nên tròn và Gradient Descent sẽ hội tụ nhanh hơn khi đi xuống đáy bát. Ngược lại, nếu các đặc trưng có phạm vi giá trị khác nhau, miệng bát sẽ bị kéo dài và quá trình đi xuống đáy bát sẽ mất thời gian hơn. Đây là lý do tại sao việc scaling (căn chỉnh) các đặc trưng của vector đầu vào X là cần thiết.

- Như chúng ta có thể quan sát, khi đi từ bên phải, thuật toán Gradient Descent di chuyển trực tiếp đến điểm tối thiểu, nhanh chóng đạt đến cực tiểu toàn cục. Trong khi đó, khi xuất phát từ bên trái, nó di chuyển gần như vuông góc với hướng của cực tiểu toàn cục, dẫn đến một hành trình dài trên một mặt phẳng gần bằng. Cuối cùng, thuật toán sẽ đạt đến mức cực tiểu, nhưng sẽ mất nhiều thời gian hơn.

### *1.1.1.3 Batch Gradient Descent*

- Để thực hiện thuật toán Gradient Descent, chúng ta cần xác định đạo hàm riêng của hàm chi phí đối với từng tham số của mô hình, ký hiệu là. Đạo hàm riêng cho chúng ta biết cách hàm chi phí thay đổi khi giá trị của  thay đổi. Đây là thông tin quan trọng để điều chỉnh các tham số và tối ưu hóa mô hình.A green line in a circle

Description automatically generated

- Biểu thức sau sẽ dùng để tính đạo hàm riêng của hàm chi phí cho tham số , được ký hiệu là :

* Thay vì tính từng đạo hàm thành phần, bạn có thể sử dụng công thức sau để tính tất cả trong 1 bước. Vector độ dốc, ký hiệu là đạo hàm riêng (vector độ dốc) cho các tham số ) của mô hình.
* Khi chúng ta có vector độc dốc và vị trí hiện tại, chúng ta chỉ cần đi ngược lại với vector độ dốc. Nghĩa là ta phải trừ θ đi 1 giá trị là Lúc này ta sẽ sử dụng tham số learning rate  để xác định giá trị của bước xuống dốc bằng cách nhân vào.

### *1.1.2 Stochastic Gradient Descent (SGD)*

- **Stochastic Gradient Descent (SGD):** Phương pháp Stochastic Gradient Descent (SGD) là một phương pháp đơn giản nhưng rất hiệu quả để áp dụng cho các bộ phân loại và hồi quy tuyến tính, như hồi quy Logistic. Mặc dù SGD đã tồn tại trong cộng đồng máy học từ lâu, gần đây nó đã nhận được sự chú ý đáng kể, đặc biệt trong ngữ cảnh của việc học trên quy mô lớn. SGD đã được áp dụng thành công cho các bài toán machine learning với dữ liệu thưa và quy mô lớn, đặc biệt là trong lĩnh vực phân loại văn bản và xử lý ngôn ngữ tự nhiên. Với dữ liệu thưa, các bộ phân loại trong mô hình này có khả năng mở rộng quy mô lên đến

105 tính năng.

A green line in a circle

Description automatically generated

- Thuật toán giảm độ dốc tiêu chuẩn cập nhật các tham số của mục tiêubằng:

trong đó kỳ vọng trong phương trình trên được ước tính gần đúng bằng cách đánh giá chi phí và độ dốc trên toàn bộ tập huấn luyện. Giảm dần độ dốc ngẫu nhiên (SGD) chỉ đơn giản là loại bỏ kỳ vọng trong bản cập nhật và tính toán độ dốc của các tham số chỉ bằng một hoặc một vài ví dụ đào tạo. Bản cập nhật mới được đưa ra bởi:

A black text on a white background

Description automatically generated

- Nói chung, trong SGD, thay vì cập nhật tham số dựa trên một mẫu đơn lẻ, ta tính toán cập nhật bằng một số mẫu huấn luyện hoặc một lô nhỏ mẫu. Lý do cho việc này có hai phần. Thứ nhất, việc sử dụng lô mẫu giúp giảm phương sai trong quá trình cập nhật tham số và có thể dẫn đến sự hội tụ ổn định hơn. Thứ hai, việc tính toán trên lô mẫu cho phép tận dụng tốt các phép toán ma trận được tối ưu hóa, đồng thời tăng hiệu suất tính toán cho việc vector hóa chi phí và độ dốc. Kích thước lô mẫu phổ biến là 256, tuy nhiên, kích thước lô mẫu tối ưu có thể khác nhau tùy thuộc vào ứng dụng và kiến trúc cụ thể.

- Trong SGD, tỷ lệ học tập α thường được đặt nhỏ hơn đáng kể so với tốc độ học tương ứng trong quá trình giảm độ dốc hàng loạt, do độ biến đổi lớn hơn trong cập nhật. Việc lựa chọn tốc độ học và lịch trình phù hợp có thể khá khó khăn. Một phương pháp thực tế hiệu quả là sử dụng tốc độ học không đổi đủ nhỏ để đạt được sự hội tụ ổn định trong các epoch ban đầu (qua toàn bộ tập huấn luyện) hoặc hai lần huấn luyện, sau đó giảm một nửa giá trị tốc độ học khi tốc độ hội tụ chậm lại. Một cách tiếp cận tốt hơn là đánh giá một tập hợp tổ chức sau mỗi epoch và giảm tốc độ học khi sự thay đổi mục tiêu giữa các epoch dưới một ngưỡng nhỏ.

- Một điểm cuối cùng nhưng quan trọng liên quan đến SGD là thứ tự của dữ liệu được trình bày cho thuật toán. Nếu dữ liệu được sắp xếp theo một thứ tự có ý nghĩa nào đó, điều này có thể gây sai lệch độ dốc và dẫn đến sự hội tụ kém. Một phương pháp tốt để tránh điều này là xáo trộn dữ liệu một cách ngẫu nhiên trước mỗi giai đoạn huấn luyện.

***1.1.3 Adam ( Adaptive Moment Estimation)***

***-*** Adam optimizer là một thuật toán tận dụng kết hợp của RMSprop và momentum trong quá trình tối ưu hóa. Thuật toán này sử dụng hai trạng thái nội tại là momentum (m) và squared momentum (v) của gradient đối với các tham số. Sau mỗi batch huấn luyện, giá trị của m và v được cập nhật thông qua phương pháp trung bình có trọng số theo hình dạng mô phỏng (exponential weighted averaging). Mã giải cho quá trình cập nhật của m và v được thực hiện để đảm bảo tính hiệu quả của thuật toán:

trong đó, beta được xem như là một siêu tham số:

trong đó, là learning rate, là giá trị được thêm vào để ngăn việc chia cho 0

- Để việc descent được thực hiện nhanh hơn, thuật toán đã sử dụng hai kỹ thuật:

* Áp dụng tính trung bình động theo hình dạng mô phỏng (exponential moving average) cho giá trị đạo hàm và lưu kết quả vào biến m, sau đó sử dụng nó như một tử số trong quá trình cập nhật hướng. Ý tưởng là khi giá trị của m lớn, quá trình descent đang di chuyển theo hướng chính xác và đòi hỏi bước nhảy lớn để tiến triển nhanh hơn. Ngược lại, khi giá trị của m nhỏ, quá trình descent có thể không đi theo hướng tối ưu và yêu cầu một bước nhảy nhỏ để thăm dò. Đây chính là thành phần momentum quan trọng của thuật toán..
* Áp dụng tính trung bình động theo hình dạng mô phỏng (exponential moving average) cho bình phương giá trị đạo hàm và lưu kết quả vào biến v, sau đó sử dụng nó như mẫu số trong quá trình cập nhật hướng. Ý tưởng là khi gradient mang giá trị dương và âm, việc cộng các giá trị lại theo công thức tính m sẽ tạo ra giá trị gần số 0, do sự lẫn nhau giữa các giá trị dương và âm. Tuy nhiên, trong trường hợp này, giá trị của v sẽ là một số lớn. Do đó, nếu v có giá trị thấp, các phần cập nhật sẽ lớn và ngược lại. Chúng ta đặt v ở mẫu số vì khi chia cho một giá trị lớn, giá trị của các phần cập nhật sẽ trở nên nhỏ, trong khi khi v có giá trị thấp, các phần cập nhật sẽ tăng lên. Điều này đại diện cho phần tối ưu RMSProp của thuật toán.
* Ở đây, m được xem như là moment thứ nhất, v được xem như là moment thứ hai, do đó, thuật toán có tên là 'Adaptive Moment Estimation' (Adam).
* Để lý giải tại sao Adam hội tụ nhanh hơn so với SGD, chúng ta có thể giải thích như sau:
* Exponential weighted averaging cung cấp cho chúng ta giá trị xấp xỉ gradient mượt mà qua mỗi lần lặp, dẫn tới sự tăng tính dừng. Sau đó, việc chia cho căng bậc 2 của giá trị v giảm đáng kể số lượng bước di chuyển khi phương sai của giá trị gradient tăng lên. Điều này, như đã giải thích ở trên, có nghĩa là khi hướng đi của mô hình không rõ ràng, thuật toán Adam thực hiện các bước di chuyển nhỏ như là một cách thăm dò. Ngược lại, khi hướng đi rõ ràng, thuật toán thực hiện các bước di chuyển lớn và nhanh.
* Tuy thuật toán Adam hoạt động khá hiệu quả, nhưng nó cũng mang theo một số vấn đề không hiệu quả. Tác giả của AdaBelief đã chỉ ra một số điểm không hiệu quả của thuật toán.

***1.1.4 Kết luận và so sánh các phương pháp vừa tìm hiểu***

|  |  |  |  |
| --- | --- | --- | --- |
|  | Gradient Descent | Stochastic Gradient Descent | Adam |
| Hiệu suất và tối ưu hóa | Gradient Descent có thể đạt được sự hội tụ đến điểm tối ưu toàn cục khi và chỉ khi tốc độ học được đặt ở mức đủ nhỏ và hàm mất mát là lồi (convex). | Stochastic Gradient Descent (SGD) thường hội tụ nhanh trên các tập dữ liệu lớn do nó chỉ tính toán gradient trên một mini-batch nhỏ. Tuy nhiên, SGD có thể không hội tụ chính xác đến điểm tối ưu toàn cục và có thể dao động xung quanh điểm tối ưu. | Adam là một thuật toán tối ưu hóa kết hợp cả momentum và RMSprop, tạo ra một phương pháp linh hoạt và hiệu quả. Thường thì, Adam đem lại kết quả tốt và hội tụ nhanh hơn so với SGD trên nhiều bài toán. |
| Tốc độ hội tụ | Gradient Descent có thể chậm hơn so với SGD và Adam trong việc hội tụ tới điểm tối ưu do phải tính toán gradient trên toàn bộ tập dữ liệu.  Top of Form | Stochastic Gradient Descent (SGD) hội tụ nhanh hơn Gradient Descent (GD) vì nó chỉ tính toán gradient trên một mini-batch nhỏ. Tuy nhiên, do sự ngẫu nhiên trong việc chọn mini-batch, SGD có thể dao động xung quanh điểm tối ưu và yêu cầu nhiều bước cập nhật hơn để hội tụ. | Adam thường hội tụ nhanh hơn so với SGD và GD trên nhiều bài toán nhờ tính chất tự điều chỉnh tốc độ học dựa trên gradient của tham số |
| Điểm tối ưu cục bố và yên ngựa | Đúng, Gradient Descent có thể mắc kẹt tại các điểm tối ưu cục bộ nếu hàm mất mát không lồi.  Top of Form | Stochastic Gradient Descent có khả năng tránh điểm tối ưu cục bộ hơn so với Gradient Descent và có thể nhảy qua các điểm yên ngựa do sự ngẫu nhiên trong việc chọn mini-batch. Điều này giúp nó có khả năng thoát khỏi các điểm tối ưu cục bộ và tiếp tục hội tụ đến điểm tối ưu toàn cục. | Adam có khả năng tránh điểm tối ưu cục bộ và điểm yên ngựa, nhưng trên một số bài toán cụ thể, nó có thể không hội tụ tốt như SGD.  Top of Form |
| Độ ổn định và đáng tin cậy | Gradient Descent có thể đảm bảo kết quả ổn định, tuy nhiên, điều này phụ thuộc vào việc tinh chỉnh tốc độ học và sự lựa chọn khởi tạo ban đầu.  Top of Form | Stochastic Gradient Descent có thể có độ dao động cao hơn do sự ngẫu nhiên trong việc chọn mini-batch, tuy nhiên, nó vẫn có thể đạt được kết quả tốt trên nhiều bài toán.  Top of Form | Adam thường mang lại kết quả ổn định và độ tin cậy cao, đặc biệt là trong các mô hình học sâu.  Top of Form |

- Kết luận: Gradient Descent là phương pháp cơ bản nhưng có tốc độ chậm và dễ mắc kẹt ở điểm tối ưu cục bộ. Stochastic Gradient Descent nhanh hơn và có khả năng tránh điểm tối ưu cục bộ, nhưng độ dao động cao. Adam là phương pháp linh hoạt và hiệu quả, tự điều chỉnh tốc độ học và có khả năng tránh điểm tối ưu cục bộ và yên ngựa, tuy nhiên, có thể không hội tụ tốt như SGD trên một số bài toán cụ thể. Lựa chọn phương pháp tối ưu hóa phụ thuộc vào yêu cầu cụ thể của bài toán, kích thước tập dữ liệu, loại mô hình và các yếu tố khác.Top of Form

## 1.2 Tìm hiểu về Continual Learning,Test Production và xây dựng giải pháp học máy

### *1.2.1 Continual Learning*

- Continual Learning (Lifelong Learning) là khái niệm xây dựng dựa trên việc liên tục học về thế giới bên ngoài để tự động phát triển kỹ năng và kiến thức ngày càng phức tạp. - Một hệ thống Continual Learning có thể được định nghĩa là một thuật toán thích nghi có khả năng học từ một luồng thông tin liên tục, với thông tin ngày càng trở nên có sẵn theo thời gian và số lượng nhiệm vụ không được xác định trước. Điều quan trọng là quá trình này phải xảy ra mà không gây quên hoặc nhiễu loạn đáng kể.

- Trong tình huống Continual Learning, mô hình học cần được xây dựng và cập nhật động các biểu diễn nội tại khi phân phối nhiệm vụ thay đổi qua thời gian sống của nó. Mục tiêu là duy trì một phần của các biểu diễn nội tại là tổng quát và không thay đổi đủ để sử dụng lại cho các nhiệm vụ tương tự, trong khi một phần khác phải bảo tồn và mã hóa các biểu diễn cụ thể cho từng nhiệm vụ.

- Một số khía cạnh quan trọng của Continual Learning bao gồm Catastrophic Forgetting (hiện tượng quên kiến thức cũ khi học tác vụ mới), Transfer Learning (tận dụng kiến thức từ tác vụ trước để giúp học tác vụ mới), Phân phối dữ liệu không đồng đều, và Quản lý bộ nhớ để lưu trữ và sử dụng lại kiến thức đã học.

### *1.2.1.1 Catastrophic forgetting*

- **Catastrophic interference:** hay còn gọi là Catastrophic forgetting, là hiện tượng xảy ra khi một mạng nơ-ron được huấn luyện trên một tập dữ liệu và sau đó quên hoàn toàn hoặc mất đi thông tin đã học khi nó được huấn luyện trên một tập dữ liệu mới. Đặc điểm nổi bật của hiện tượng này là khả năng của mạng không giữ lại kiến thức từ các phiên huấn luyện trước, thường dẫn đến việc thông tin mới ghi đè hoặc thay thế thông tin đã học trước đó. Catastrophic interference có thể là một thách thức đáng kể trong các kịch bản học liên tục hoặc học suốt đời, nơi một mô hình cần thích ứng với dữ liệu mới mà không làm mất đi kiến thức đã học từ trước.

- Ví dụ:

A close-up of a graph

Description automatically generated

Đây là dữ liệu từ thực tế và sau mỗi lần thu thập dữ liệu để train model tập mẫu của chúng thành 2 lớp:

A diagram of a solution

Description automatically generated

A diagram of a good and cheap

Description automatically generated

A diagram of a learning strategy

Description automatically generated

- Replay Strageti:

Ưu điểm:

+ Độ giảm thiểu cao

+ Đơn giản và dễ dàng

+ Bộ nhớ dồi dào

Nhược điểm:

+ Tuy bộ nhớ dồi dào nhưng không phải là vô hạn

+ Computation

A diagram of a strategy

Description automatically generated

- Latent replay:

+ Chỉ kích hoạt khi các latent đã được ghi nhớ trong bộ dữ liệu

+ Chỉ một phần portion của network cần được huấn luận lại với replay data

+ Cân bằng tốc độ và độ chính xác.

A diagram of a diagram

Description automatically generated

A table with numbers and text

Description automatically generated

A diagram of a structure

Description automatically generated

- Negative Generative replay:

A graph of different types of graphs

Description automatically generated with medium confidence

### *Test Production*

### *Định nghĩa*

* ML Testing: đề cập đến bất kỳ hoạt động nào được thiết kế để tiết lộ máy.

A diagram of a software development process

Description automatically generated

* ML Testing Workflow: là về cách tiến hành kiểm tra MLvới các hoạt động thử nghiệm khác nhau:

+ Role of Testing in ML Development:

Quy trình phát triển mô hình học máy thường bắt đầu bằng việc tạo một mô hình nguyên mẫu dựa trên dữ liệu lịch sử. Trước khi triển khai mô hình vào môi trường thực tế, quá trình kiểm thử ngoại tuyến được thực hiện để đảm bảo rằng mô hình đáp ứng các yêu cầu cần thiết, thường bao gồm việc sử dụng kỹ thuật cross-validation để đánh giá hiệu suất của mô hình.

Sau khi triển khai, mô hình sẽ tiến hành dự đoán và tạo ra các kết quả mới. Dữ liệu mới có thể được đánh giá thông qua quá trình kiểm thử trực tuyến để đánh giá mức độ tương tác của mô hình với hành vi người dùng. Các chỉ số kinh doanh như tỷ lệ mở, thời gian đọc, và tỷ lệ nhấp chuột có thể được theo dõi để đánh giá hiệu suất thực tế của mô hình trong môi trường sản xuất.

Quan trọng là kiểm thử trực tuyến, vì nó cung cấp cái nhìn động và chi tiết hơn về cách mô hình tương tác với dữ liệu thực tế và người dùng. Kiểm thử ngoại tuyến có thể không đại diện đầy đủ cho tất cả các trường hợp sử dụng và không thể kiểm thử các tình huống đặc biệt trong thực tế, do đó, kiểm thử trực tuyến là cần thiết để đảm bảo sự tin cậy và hiệu suất của mô hình trong môi trường thực tế.

+ Offline Testing: Phân tích yêu cầu, chuẩn bị dữ liệu kiểm thử, thực hiện kiểm thử, đánh giá kết quả kiểm thử, phát hiện lỗi trong các thành phần ML.

### *ML Tesing Component*

* Để xây dựng mô hình học máy, nhà phát triển phần mềm ML phải tiến hành thu thập và gắn nhãn dữ liệu, thiết kế cấu trúc chương trình học tập và sử dụng các khung học tập. Quá trình này tương tác với các thành phần như dữ liệu, chương trình học tập và khung học tập, và mỗi thành phần có thể xuất hiện lỗi. Do đó, trong quá trình kiểm thử ML, nhà phát triển cần phát hiện và sửa lỗi trong các thành phần này.A diagram of a workflow

  Description automatically generated

A diagram of a diagram

Description automatically generated

### *Software Testing và ML Testing*

Các khác biệt chính giữa kiểm thử phần mềm truyền thống và kiểm thử máy học (ML) có thể được tổng kết như sau:

- Thành phần cần kiểm thử: Kiểm thử phần mềm truyền thống tập trung vào tìm lỗi trong mã code, trong khi kiểm thử ML tìm lỗi trong dữ liệu, chương trình học, và khung học, mỗi thành phần đóng vai trò quan trọng trong xây dựng mô hình ML.

- Hành vi được kiểm thử: Mã code phần mềm truyền thống thường có hành vi cố định sau khi yêu cầu được xác định, trong khi hành vi của mô hình ML có thể thay đổi thường xuyên khi dữ liệu huấn luyện được cập nhật.

- Đầu vào kiểm thử: Đầu vào kiểm thử trong kiểm thử phần mềm truyền thống thường là dữ liệu đầu vào khi kiểm thử mã code, trong khi đầu vào kiểm thử ML có thể có nhiều hình thức đa dạng hơn.

- Test oracle: Kiểm thử phần mềm truyền thống giả định sự hiện diện của test oracle, trong khi kiểm thử ML đôi khi gặp thách thức trong việc xác định test oracle và thường sử dụng các kỹ thuật như Metamorphic relations để giảm thiểu vấn đề này.

- Tiêu chí đầy đủ kiểm thử: Các tiêu chí đầy đủ kiểm thử phổ biến cho phần mềm truyền thống như line coverage, branch coverage có thể không phản ánh đầy đủ cho kiểm thử ML, do sự khác biệt về mô hình lập trình và biểu diễn logic.

- Số lỗi giả trong việc phát hiện lỗi: Kiểm thử ML có thể tạo ra nhiều kết quả giả-positives trong việc báo cáo lỗi do khó khăn trong xác định các oracle đáng tin cậy.

- Vai trò của người kiểm thử: Trong kiểm thử ML, các lỗi có thể tồn tại không chỉ trong chương trình học mà còn trong dữ liệu hoặc thuật toán, đòi hỏi sự đóng góp của nhà khoa học dữ liệu hoặc thiết kế thuật toán trong vai trò của người kiểm thử.

**TÀI LIỆU THAM KHẢO**

**Tiếng Anh**

[1] *"Adam: A Method for Stochastic Optimization" by Diederik P. Kingma và Jimmy Ba.*

[2] *"SGD - Stochastic Gradient Descent Optimization" by Leon Bottou, Frank E. Curtis và Jorge Nocedal.*

[3] *"Continual Lifelong Learning with Neural Networks: A Review" by German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan.*

[4] "RMSprop: Divide the Gradient by a Running Average of Its Recent Magnitude" by Geoffrey Hinton.

[5] "Pattern Recognition and Machine Learning" by Christopher M. Bishop.

**Tiếng Việt**

[1] *"Tối ưu hoá trong Machine Learning" của Trần Trung Hiếu.*

[2] *"Kiểm thử mô hình học máy" của Nguyễn Đức Thành, Trần Trung Hiếu.*

[3] *"Các phương pháp tối ưu trong Deep Learning" của Nguyễn Tấn, Trần Minh Khang, Nguyễn Hoàng Vũ.*

[4] "Xây dựng mô hình học máy" của Lê Ngọc Thành, Lê Thị Quý.

[5] "Machine Learning Yearning" của Andrew Ng (Bản dịch: "Khao khát Học máy").