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*:

**LÊ PHẠM HOÀNG PHƯƠNG - 52100922**

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:

Lê Phạm Hoàng Phương – 52100922

**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)*

*Lê Phạm Hoàng Phương*

# 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](#_Toc154061147)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN 3](#_Toc154061148)

[TÓM TẮT 5](#_Toc154061149)

[DANH MỤC HÌNH VẼ 7](#_Toc154061150)

[CHƯƠNG 1: NỘI DUNG 1](#_Toc154061151)

[1.1 Các phương pháp Optimizer trong huấn luyện mô hình học máy 1](#_Toc154061152)

[*1.1.1 Gradient Descent* 1](#_Toc154061153)

[*1.1.1.1 Khái niệm* 1](#_Toc154061154)

[*1.1.1.2 Learning rate* 2](#_Toc154061155)

[*1.1.1.3 Batch Gradient Descent* 5](#_Toc154061156)

[*1.1.2 Stochastic Gradient Descent (SGD)* 7](#_Toc154061157)

[1.2 Tìm hiểu về Continual Learning,Test Production và xây dựng giải pháp học máy 13](#_Toc154061158)

[*1.2.1 Continual Learning* 13](#_Toc154061159)

[*1.2.1.1 Catastrophic forgetting* 14](#_Toc154061160)

[*1.2.2* *Test Production* 20](#_Toc154061161)

[*1.2.2.1* *Định nghĩa* 20](#_Toc154061162)

[*1.2.2.2* *ML Tesing Component* 21](#_Toc154061163)

[*1.2.2.3* *Software Testing và ML Testing* 21](#_Toc154061164)

[**TÀI LIỆU THAM KHẢO** 24](#_Toc154061165)

# DANH MỤC HÌNH VẼ

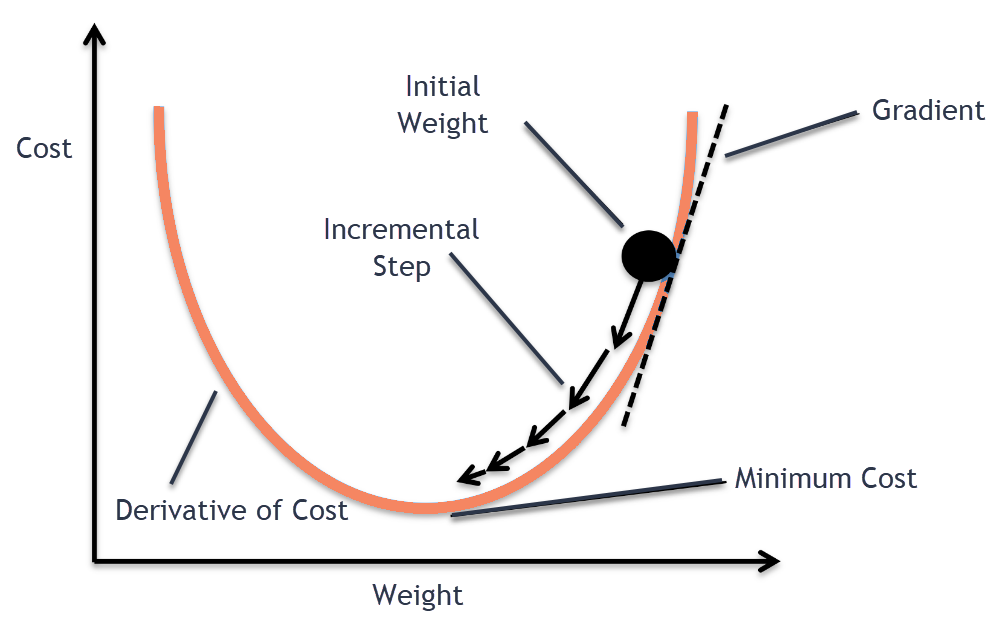
# CHƯƠNG 1: NỘI DUNG

## 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 thuật toán tìm tối ưu chung cho các hàm số. Ý tưởng chung của GD là điều chỉnh các tham số để lặp đi lặp lại thông qua mỗi dữ liệu huấn luyện để giảm thiểu hàm chi phí.



- Gradient Descent là một thuật toán tối ưu lặp (iterative optimization algorithm) được sử dụng trong các bài toán Machine Learning và Deep Learning (thường là các bài toán tối ưu lồi — Convex Optimization) với mục tiêu là tìm một tập các biến nội tại (internal parameters) cho việc tối ưu models. Trong đó:

* Gradient: là tỷ lệ độ nghiêng của đường dốc (rate of inclination or declination of a slope). Về mặt toán học, Gradient của một hàm số là đạo hàm của hàm số đó tương ứng với mỗi biến của hàm. Đối với hàm số đơn biến, chúng ta sử dụng khái niệm Derivative thay cho Gradient.
* Descent: là từ viết tắt của descending, nghĩa là giảm dần.

- Gradient Descent có nhiều dạng khác nhau như Stochastic Gradient Descent (SGD), Mini-batch SDG. Nhưng về cơ bản thì đều được thực thi như sau:

* 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:

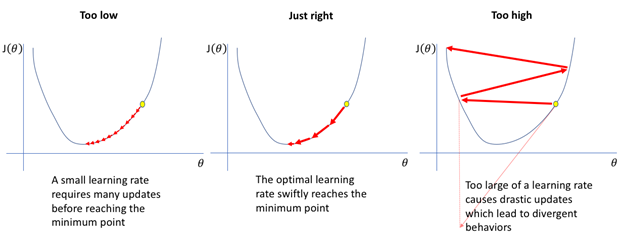
trong đó là tập các biến cần cập nhật,  là tốc độ học (learning rate),  là Gradient của hàm mất mát f theo tập θ.

### *1.1.1.2 Learning rate*

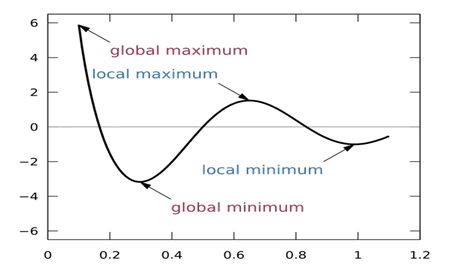
- Có 1 tham số quan trọng trong Gradient Descent đó là giá trị độ lớn của mỗi lần di chuyển (giống như độ dài sải chân khi bạn leo xuống dốc).

Tham số này được gọi là learning rate (tốc độ học). Nếu learning rate quá nhỏ, thuật toán sẽ phải thực hiện nhiều bước để hội tụ và sẽ mất nhiều thời gian.

Tuy nhiên nếu learning rate quá lớn sẽ khiến thuật toán đi qua cực tiểu, và vượt hẳn ra ngoài khiến thuật toán không thể hội tụ được.



- Trong thực tế, không phải hàm số nào cũng chỉ có 1 cực tiểu. Ta sẽ có khái niệm cực tiểu cục bộ và cực tiểu toàn cục. Hiểu nôm na nó giống như các hố hoặc các tảng đá ở trên núi khi bạn đang leo xuống núi. Lúc này việc tìm ra cực tiểu sẽ trở nên khó khăn hơn. Xem hình sau để biết chi tiết:



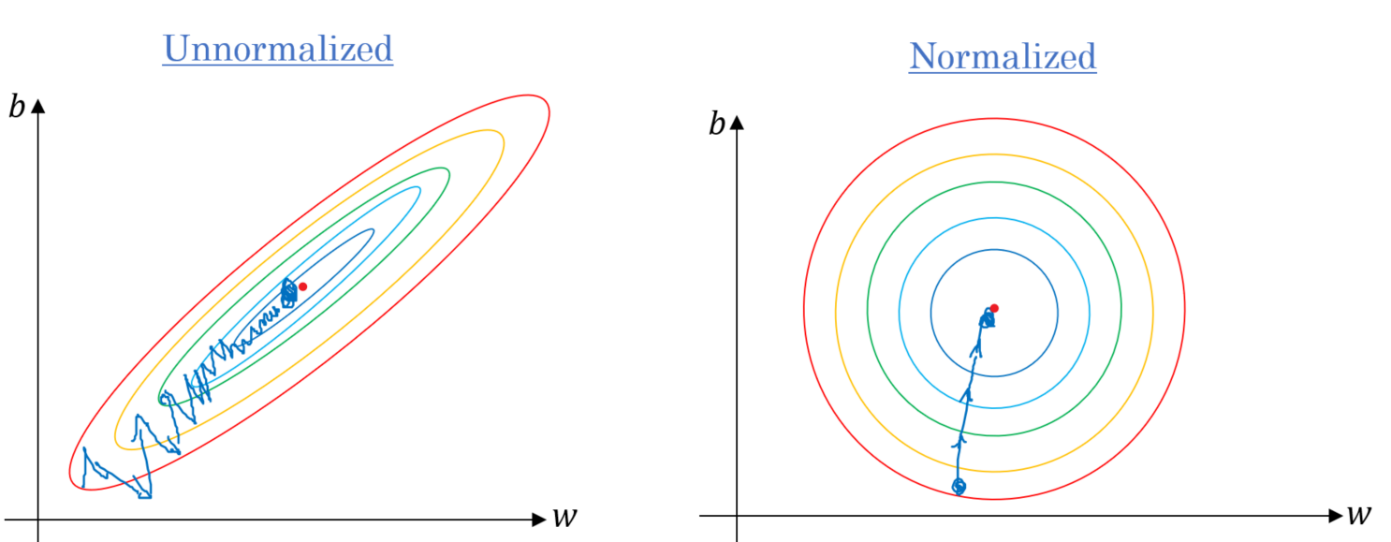
* Sẽ có 2 vấn đề lúc này đối với GD:

Điểm xuất phát có thể ở bên trái hoặc bên phải, nếu xuất phát từ bên trái, thuật toán sẽ hội tụ ở local minimum (cực tiểu địa phương) mà không đi đến được global minium (cực tiểu toàn cục).

Hoặc nếu xuất phát từ bên phải sẽ phải mất nhiều thời gian để vượt qua Plateau để đến được global minimum và nếu kết thúc thuật toán quá sớm thì sẽ không đến được global minimum.

Bài trước chúng ta có sử dụng hàm chi phí MSE cho bài toán hồi quy tuyến tính, rất may là hàm này là hàm lồi. Nghĩa là nếu 1 đường thẳng nối 2 điểm bất kì trên đồ thị hàm lồi thì đường thẳng này sẽ không cắt đồ thị. Điều này nghĩa là không có cực tiểu địa phương (local minimum) mà chỉ có 1 cực tiểu toàn cục. Đây cũng là một hàm liên tục có độ dốc không bao giờ thay đổi đột ngột. Vì vậy ỏ đây GD có 1 vấn đề, đó là nó sẽ không tiến gần đến được global minimum (trừ khi thời gian học đủ lâu và learning rate đủ nhỏ)

Trên thực tế, hàm chi phí có dạng đồ thị giống chiếc bát, nếu các feature (đặc điểm của đầu vào - thành phần của vector X) có cùng phạm vi giá trị, thì miệng bát sẽ tròn và để GD đi xuống đáy bát sẽ nhanh hơn. Nếu các feature khác phạm vi giá trị thì miệng bát sẽ bị kéo dài ra và việc đi xuống đáy bát sẽ tốn thời gian hơn. Đây là lý do vì sao các feature của vector đầu vào X cần phải được scaling (căn chỉnh).



- Như ta có thể thấy, ở bên phải thuật toán Gradient Descent đi thẳng về điểm tối thiểu, do đó nhanh chóng đạt được cực tiểu toàn cục, trong khi bên trái, nó đi theo hướng gần như trực giao với hướng về cực thiểu toàn cục, vì vậy nó kết thúc bằng 1 hành trình dài xuống một 1 mặt gần như bằng phẳng. Cuối cùng nó sẽ đạt đến mức cực tiểu, nhưng sẽ mất nhiều thời gian.

### *1.1.1.3 Batch Gradient Descent*

- Để thực hiện thuật toán Gradient Descent, chúng ta phải tìm được đạo hàm của hàm chi phí ảnh hưởng đến từng tham số của mô hình . Nói khác đi, cần phải xác định được giá trị hàm chi phí thay đổi thế nào nếu thay đổi . Cái này được gọi là đạo hàm riêng (partial derivative).

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)** là một cách tiếp cận đơn giản nhưng rất hiệu quả để phù hợp với các bộ phân loại và hồi quy tuyến tính theo các hàm như là Logictics Regression. Mặc dù SGD đã xuất hiện trong cộng đồng máy học từ lâu nhưng nó mới nhận được sự chú ý đáng kể gần đây trong bối cảnh học tập quy mô lớn.

- SGD đã được áp dụng thành công cho các bài toán machine learning thưa thớt và quy mô lớn thường gặp trong phân loại văn bản và xử lý ngôn ngữ tự nhiên. Do dữ liệu thưa thớt nên các bộ phân loại trong model này dễ dàng mở rộng quy mô thành các vấn đề với hơn 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, mỗi lần cập nhật tham số trong SGD được tính toán bằng một vài mẫu huấn luyện hoặc một lô nhỏ thay vì một mẫu đơn lẻ. Lý do cho điều này gồm hai phần: thứ nhất, điều này làm giảm phương sai trong cập nhật tham số và có thể dẫn đến sự hội tụ ổn định hơn, thứ hai, điều này cho phép tính toán tận dụng các phép toán ma trận được tối ưu hóa cao nên được sử dụng trong tính toán chi phí được vector hóa tốt và dốc. Kích thước minibatch điển hình là 256, mặc dù kích thước tối ưu của minibatch có thể khác nhau đối với các ứng dụng và kiến ​​trúc khác nhau.

- Ở SGD tỷ lệ học tập thường nhỏ hơn nhiều so với tốc độ học tương ứng trong quá trình giảm độ dốc hàng loạt vì có nhiều phương sai hơn trong bản cập nhật. Việc lựa chọn tốc độ học và lịch trình phù hợp (tức là thay đổi giá trị của tốc độ học khi tiến trình học) có thể khá khó khăn. Một phương pháp tiêu chuẩn hoạt động tốt trong thực tế là sử dụng tốc độ học không đổi đủ nhỏ để mang lại sự hội tụ ổn định trong kỷ nguyên ban đầu (đi qua toàn bộ tập huấn luyện) hoặc hai lần huấn luyện và sau đó giảm một nửa giá trị của tốc độ học khi tốc độ hội tụ chậm lại . Một cách tiếp cận thậm chí còn tốt hơn là đánh giá một tập hợp được tổ chức sau mỗi kỷ nguyên và giảm tốc độ học khi sự thay đổi mục tiêu giữa các kỷ nguyên dưới một ngưỡng nhỏ.

=> Một điểm cuối cùng nhưng quan trọng liên quan đến SGD là thứ tự chúng tôi trình bày dữ liệu cho thuật toán. Nếu dữ liệu được đưa ra theo một thứ tự có ý nghĩa nào đó, điều này có thể làm sai lệch độ dốc và dẫn đến độ hội tụ kém. Nói chung, 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 kết hợp kỹ thuật của RMS prop và momentum. Thuật toán sử dụng hai internal states momentum (m) và squared momentum (v) của gradient cho các tham số. Sau mỗi batch huấn luyện, giá trị của m và v được cập nhật lại sử dụng exponential weighted averaging.

Mã giải của việc cập nhật m và v

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:

* Tính exponential moving average của giá trị đạo hàm lưu vào biến m và sử dụng nó là tử số của việc cập nhật hướng. Với ý nghĩa là nếu m có giá trị lớn, thì việc descent đang đi đúng hướng và cần bước nhảy lớn hơn để đi nhanh hơn. Tương tự, nếu giá trị m nhỏ, phần descent có thể không đi về hướng tối tiểu và nên đi 1 bước nhỏ để thăm dò. Đây là phần momentum của thuật toán.
* Tính exponential moving average của bình phương gía trị đạo hàm lưu vào biến v và sử dụng nó là phần mẫu số của việc cập nhật hướng. Với ý nghĩa như sau: Giả sử gradient mang các giá trị dương, âm lẫn lộn, thì khi cộng các giá trị lại theo công thức tính m sẽ được giá trị m gần số 0. Do âm dương lẫn lộn nên nó bị triệt tiêu lẫn nhau. Nhưng trong trường hợp này thì v sẽ mang giá trị lớn. Do đó, trong trường hợp này sẽ không hướng tới cực tiểu. Chúng ta để v ở phần mẫu vì khi chia cho một giá trị cao, giá trị của các phần cập nhật sẽ nhỏ, và khi v có giá trị thấp, phần cập nhật sẽ lớn. Đây chính là phần tối ưu RMSProp của thuật toán.

Ở đây, m được xem như là moment thứ nhất, v xem như là moment thứ hai, nên thuật toán có tên là “Adaptive moment estimation”.

- Để lý giải vì sao Adam lại hội tụ nhanh hơn so với SGD, chúng ta có thể giải thích như sau:

=>E*xponential weighted averaging* cho chúng ta giá trị xấp xỉ gradient mượt hơn qua mỗi lần lặp, dẫn tới tăng tính dừng. Sau đó, việc chia cho căng bậc 2 của giá trị v làm số lước của chúng ta giảm mạnh khi phương sai của giá trị gradient tăng lên. Điều này , như giải thích ở trên, có nghĩa là, khi hướng đi của mô hình chỉ ra không rõ ràng, thuật toán Adam thực hiện các bước đi nhỏ coi như là thăm dò thôi. Và sẽ thực hiện các bước đi lớn, nhanh khi hướng đi rõ ràng.

Thuật toán Adam hoạt động khá hiệu quả, nhưng bản thân nó cũng có những vấn đề. Tác giả của AdaBelief đã chỉ ra một vài 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 hội tụ đến điểm tối ưu toàn cục nếu tốc độ học đủ nhỏ và hàm mất mát là lồi (convex). | Stochastic Gradient Descent thường hội tụ nhanh trên các tập dữ liệu lớn vì 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 kết hợp cả momentum và RMSprop để cung cấp một phương pháp tối ưu hóa linh hoạt và hiệu quả. Adam thường cho kết quả tốt và hội tụ nhanh hơn SGD trên nhiều bài toán. |
| Tốc độ hội tụ | Gradient Descent có thể chậm hơn SGD và Adam trong việc hội tụ tới điểm tối ưu do phải tính toán gradient trên toàn bộ tập dữ liệu. | Stochastic Gradient Descent hội tụ nhanh hơn GD vì nó chỉ tính toán gradient trên một mini-batch nhỏ. Tuy nhiên, do tính ngẫu nhiên của việc chọn mini-batch, SGD có thể dao động xung quanh điểm tối ưu và cần nhiều bước cập nhật hơn để hội tụ. | Adam thường hội tụ nhanh hơn 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 | Gradient Descent có thể bị mắc kẹt tại các điểm tối ưu cục bộ nếu hàm mất mát không lồi. | Stochastic Gradient Descent có khả năng tránh được điểm tối ưu cục bộ hơn GD và có thể nhảy qua các điểm yên ngựa. | Adam cũng có khả năng tránh được điểm tối ưu cục bộ và điểm yên ngựa, nhưng có thể không hội tụ tốt như SGD trên một số bài toán cụ thể. |
| Độ ổn định và đáng tin cậy | Gradient Descent có thể đạt được kết quả ổn định nhưng phụ thuộc vào tốc độ học và lựa chọn khởi tạo ban đầu. | Stochastic Gradient Descent có thể có độ dao động cao hơn do tính ngẫu nhiên của việc chọn mini-batch, nhưng nó có thể đạt được kết quả tốt trên nhiều bài toán. | Adam thường đạt được kết quả ổn định và có độ tin cậy cao, đặc biệt là trong các mô hình học sâu. |

- Kết luận: Gradient Descent là phương pháp cơ bản nhưng có tốc độ chậm và dễ bị mắc kẹt ở điểm tối ưu cục bộ. Stochastic Gradient Descent nhanh hơn và có khả năng tránh được điểm tối ưu cục bộ, nhưng có độ dao động cao hơn. Adam là phương pháp linh hoạt và hiệu quả, tự điều chỉnh tốc độ học và có khả năng tránh được điểm tối ưu cục bộ và yên ngựa, nhưng 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.

## 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 được xây dựng dựa trên ý tưởng học liên tục về thế giới bên ngoài nhằm khả năng phát triển tự động, tăng dần các kỹ năng và kiến thức ngày càng phức tạp hơn.

- Một hệ thống Continual Learning có thể được xác định là một thuật toán thích nghi có khả năng học từ một luồng thông tin liên tục, trong đó thông tin này dần dần trở nên có sẵn theo thời gian và số lượng nhiệm vụ cần được học (ví dụ: các lớp thành viên trong một nhiệm vụ phân loại) không được xác định trước. Quan trọng nhất, việc chứa đựng thông tin mới này phải xảy ra mà không gây quên hoặc nhiễu loạn đáng kể.

- Do đó, trong tình huống Continual Learning, một mô hình học cần được xây dựng từng bước và cập nhật động các biểu diễn nội tại khi phân phối nhiệm vụ thay đổi động qua thời gian sống của nó. Lý tưởng nhất, một phần của các biểu diễn nội tại này sẽ được tổng quát và không thay đổi đủ để có thể 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: Đây là hiện tượng khi mô hình học tác vụ mới, nó quên đi kiến thức đã học từ các tác vụ trước đó. Continual Learning cố gắng giải quyết vấn đề này bằng cách duy trì kiến thức cũ trong quá trình học tác vụ mới.

+ Transfer Learning: Continual Learning tận dụng kiến thức đã học từ các tác vụ trước đó để giúp định hình mô hình cho các tác vụ mới. Điều này giúp tăng tốc độ học và cải thiện hiệu suất của mô hình.

+ Phân phối dữ liệu không đồng đều: Trong môi trường học tiếp tục, phân phối của dữ liệu có thể thay đổi theo thời gian. Continual Learning phải đối mặt với việc xử lý dữ liệu không đồng nhất và thích ứng với các thay đổi này.

+ Quản lý bộ nhớ: Continual Learning đòi hỏi quản lý bộ nhớ hiệu quả để lưu trữ kiến thức đã học từ các tác vụ trước đó. Các phương pháp như replay buffer hoặc generative models được sử dụng để lưu trữ và phân phối lại dữ liệu huấn luyện cũ để giúp mô hình ghi nhớ và sử dụng lại kiến thức đã học.

### *1.2.1.1 Catastrophic forgetting*

- **Catastrophic interference ( *Catastrophic forgetting*)** 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. Hiện tượng này được đặc trưng bởi khả năng của mạng không thể giữ lại kiến thức từ các phiên huấn luyện trước và có xu hướng ghi đè hoặc thay thế thông tin đã học trước đó bằng thông tin mới. Đây có thể là một thách thức đáng kể trong các kịch bản học liên tục hoặc học suốt đời, nơi một mô hình cần.

- 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:

Đầu tiên, một mô hình nguyên mẫu được tạo dựa trên dữ liệu lịch sử. Trước khi triển khai mô hình, kiểm thử ngoại tuyến được tiến hành để đảm bảo rằng mô hình đáp ứng các yêu cầu cần thiết( cross-validation). Sau khi triển khai, mô hình sẽ thực hiện các dự đoán và tạo ra kết quả mới. Dữ liệu mới có thể được phân tích thông qua kiểm thử trực tuyến để đánh giá mức độ tương tác của mô hình với hành vi người dùng, bao gồm các chỉ số kinh doanh như tỷ lệ mở, thời gian đọc, và tỷ lệ nhấp chuột. Kiểm thử trực tuyến là cần thiết vì kiểm thử ngoại tuyến thường không đại diện đầy đủ cho dữ liệu trong tương lai và không thể kiểm thử các trường hợp đặc biệt trong các tình huống 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 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, mỗi thành phần có thể có lỗi. Do đó, trong quá trình thử nghiệm ML, nhà phát triển cần tìm 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 tóm tắt như sau:

* Thành phần cần kiểm thử: Kiểm thử phần mềm truyền thống 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: Hành vi của mã code phần mềm truyền thống thường được 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 kiểm thử ML, đầu vào kiểm thử có thể có nhiều hình thức đa dạng hơn.
* Test oracle: Kiểm thử phần mềm truyền thống thường giả định sự hiện diện của test oracle. Kết quả đầu ra có thể được xác minh với các giá trị kỳ vọng bởi nhà phát triển. Tuy nhiên, trong máy học, câu trả lời được tạo ra dựa trên một tập hợp các giá trị đầu vào sau khi được triển khai trực tuyến. Độ chính xác của số lượng lớn câu trả lời được tạo ra thường được xác nhận thủ công. Xác định test oracle vẫn là một thách thức, vì nhiều thuộc tính mong muốn khó có thể xác định một cách hình thức. Metamorphic relations [71] là một loại pseudo oracle được áp dụng để tự động giảm thiểu vấn đề oracle trong kiểm thử máy học.
* Tiêu chí đầy đủ kiểm thử: Tiêu chí đầy đủ kiểm thử được sử dụng để đo lường số liệu về mức độ kiểm thử của phần mềm mục tiêu. Hiện nay, nhiều tiêu chí đầy đủ được đề xuất và được áp dụng rộng rãi trong ngành công nghiệp, ví dụ như line coverage, branch coverage, dataflow coverage. Tuy nhiên, do sự khác biệt cơ bản về mô hình lập trình và định dạng biểu diễn logic cho phần mềm máy học và phần mềm truyền thống, cần có các tiêu chí đầy đủ kiểm thử mới để xem xét các đặc điểm của phần mềm máy học.
* Số lỗi giả trong việc phát hiện lỗi: Do khó khăn trong việc xác định các oracle đáng tin cậy, kiểm thử ML có xu hướng tạo ra nhiều kết quả giả-positives trong việc báo cáo lỗi.
* Vai trò của người kiểm thử: Các lỗi trong kiểm thử ML có thể tồn tại không chỉ trong chương trình học, mà còn trong dữ liệu hoặc thuật toán, do đó các nhà khoa học dữ liệu hoặc thiết kế thuật toán cũng có thể đóng 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.*

**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.*