TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**MAI XUÂN HÀ - 52100885**

**TÌM HIỂU LÝ THUYẾT HỌC MÁY**

**BÁO CÁO CUỐI KỲ**

**NHẬP MÔN HỌC MÁY**

**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**



**MAI XUÂN HÀ - 52100885**

**TÌM HIỂU LÝ THUYẾT HỌC MÁY**

**BÁO CÁO CUỐI KỲ**

**NHẬP MÔN HỌC MÁY**

Người hướng dẫn

**TS. LÊ ANH CƯỜNG**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

**LỜI CẢM ƠN**

Đầu tiên, em xin cảm ơn Trường Đại học Tôn Đức Thắng vì đã đưa môn nhập môn Học máy vào chương trình giảng dạy. Đặc biệt, em xin cám ơn sâu sắc đến giảng viên của bộ môn Thầy Lê Anh Cường đã truyền đạt những kiến thức quý báu cho em trong quá trình học tập. Đây sẽ là những kiến thức quý báu trong hành trang để chúng em có thể đi vươn xa trong tương lai.

Đây là một môn học vô cùng thú vị, bổ ích và có tính thực tế. Đảm bảo cung cấp đầy đủ kiến thức và gắn liền với nhu cầu thực tiễn của sinh viên Công Nghệ Thông Tin. Mặc dù em đã cố gắng hết sức những dự án sẽ có thể có những thiếu sót và những sai sót trong dự án, kính mong thầy cô xem xét và góp ý để em được hoàn thiện hơn.

Em xin chân thành cảm ơn.

*TP. Hồ Chí Minh, ngày 15 tháng 12 năm 2023*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

|  |
| --- |
| *Hà*  *Mai Xuân Hà* |

**CÔNG TRÌNH ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khóa học của 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 Dự 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 Dự 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 15 tháng 12 năm 2023*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

|  |
| --- |
| *Hà*  *Mai Xuân Hà* |

**TÌM HIỂU LÝ THUYẾT HỌC MÁY**

**TÓM TẮT**

Tìm hiểu, so sánh các phương pháp Optimizer trong huấn luyện mô hình học máy.

* Giới thiệu vấn đề
* Các phương pháp cơ bản, công thức, ưu nhược điểm của nó
* So sánh

Tìm hiểu về Continual Learning và Test Production khi xây dựng một giải pháp học máy để giải quyết một bài toán nào đó

* Tìm hiểu về Continual Learning
* Tìm hiểu về Test Production
* Ứng dụng

**MỤC LỤC**

[DANH MỤC HÌNH VẼ vi](#_Toc154253914)

[DANH MỤC BẢNG BIỂU vii](#_Toc154253915)

[DANH MỤC CÁC CHỮ VIẾT TẮT viii](#_Toc154253916)

[CHƯƠNG 1. CÁC PHƯƠNG PHÁP OPTIMIZER 9](#_Toc154253917)

[1.1 Giới thiệu 9](#_Toc154253918)

[1.2 Các phương pháp cơ bản 9](#_Toc154253919)

[1.2.1 Gradient Descent và các biến thể 9](#_Toc154253920)

[1.2.2 Momentum và các biến thể 10](#_Toc154253921)

[1.2.3 Một số Optimazier khác 12](#_Toc154253922)

[1.2.4 Bảng so sánh 17](#_Toc154253923)

[CHƯƠNG 2. Continual Learning và Test Production 18](#_Toc154253924)

[2.1 Continual learning 18](#_Toc154253925)

[2.1.1 Định nghĩa 19](#_Toc154253926)

[2.1.2 Mục đích 19](#_Toc154253927)

[2.1.3 Các cách triển khai 19](#_Toc154253928)

[2.1.4 Xây Dựng Hệ Thống Học Liên Tục 19](#_Toc154253929)

[2.1.5 Các Giai Đoạn Xây Dựng 21](#_Toc154253930)

[2.1.6 Xác Định Tần Suất Cập Nhật Mô Hình 21](#_Toc154253931)

[2.2 Test Production 22](#_Toc154253932)

[2.2.1 Các cách kiểm tra phổ biến trong thử nghiệm sản xuất 22](#_Toc154253933)

[2.2.2 Ý nghĩa của thử nghiệm sản xuất trong nghiên cứu học liên tục 22](#_Toc154253934)

[2.2.3 Quá trình thử nghiệm sản xuất 23](#_Toc154253935)

[2.2.4 Những thách thức trong thử nghiệm sản xuất trong nghiên cứu học liên tục 23](#_Toc154253936)

[2.2.5 Kết luận 24](#_Toc154253937)

[TÀI LIỆU THAM KHẢO 25](#_Toc154253938)

# DANH MỤC HÌNH VẼ

**No table of figures entries found.**

# DANH MỤC BẢNG BIỂU

**No table of figures entries found.**

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| GD | Gradient Descent |
| SCD | Stochastic Gradient Descent |
| NAG | Nesterov accelerated gradient |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# CÁC PHƯƠNG PHÁP OPTIMIZER

## Giới thiệu

Trong quá trình huấn luyện một mô hình học máy, việc điều chỉnh trọng số của mô hình rất quan trọng để đảm bảo các độ chính xác và tối ưu hóa của dự đoán tốt nhất có thể.

Các phương pháp tối ưu hóa sẽ liên kết hàm mất mát và các tham số của mô hình bằng cách cập nhật mô hình dựa trên kết quả của hàm mất mát và tối ưu hóa mô hình tốt nhất bằng cách chỉnh sửa các trọng số.

Các phương pháp đó ảnh hưởng đến tốc độ hội tụ của mô hình, tổng quát hóa và hiệu suất của mô hình trên dữ liệu mới.

## Các phương pháp cơ bản

### Gradient Descent và các biến thể

#### Gradient Descent (GD)

GD là một thuật toán tối ưu hóa được sử dụng trong nhiều lĩnh vực, bao gồm học máy, học sâu và khoa học dữ liệu. Mục đích của GD là tìm giá trị tối ưu của một hàm bằng cách di chuyển dần dần trong không gian tìm kiếm theo hướng có độ dốc lớn nhất (tốc dộ ảnh hưởng bởi learning rate).

Công thức:

* là giá trị của trọng số mới
* lá giá trị của trọng số hiện tại
* là learning rate
* là đạo hàm của hàm số tại trọng số đó

Ưu điểm:

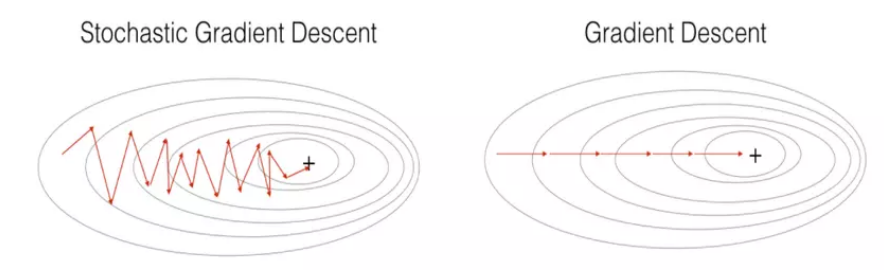
* Đơn giản và dễ hiểu
* Có thể được áp dụng cho nhiều loại hàm mục tiêu
* Có thể được sử dụng để tối ưu hóa nhiều tham số cùng lúc

Nhược điểm:

* Có thể bị mắc kẹt tại cực đại hoặc cực tiểu cục bộ
* Có thể chậm hội tụ
* Có thể nhạy cảm với learning rate α
* Không phù hợp với bộ dữ liệu được cập nhật liên tục

#### Stochastic Gradient Descent (SGD)

Đối với tập dữ liệu bình thường GD hoạt động rất tốt đến việc tìm ra điểm cực tiểu và ít ảnh hưởng bởi độ nhiễu và độ ngẫu nhiên hơn. Nhưng đối với tập dữ liệu siêu lớn, việc sử dụng GD là không khả thi vì nó lượng tài nguyên sử dụng để chạy qua toàn bộ dữ liệu là siêu lớn. Để giải quyết vấn đề này, SGD được áp dụng với bài toán có dữ liệu lớn. Phương pháp này sẽ chọn ra một mẫu dữ liệu ngẫu nhiên để thực hiện GD trên nó trong mỗi lần lặp, từ đó cải thiện thời gian huấn luyện và tốc độ đạt tới điểm cực tiểu.



Ưu điểm:

* Thuật toán giải quyết được đối với cơ sở dữ liệu lớn mà GD không làm được.

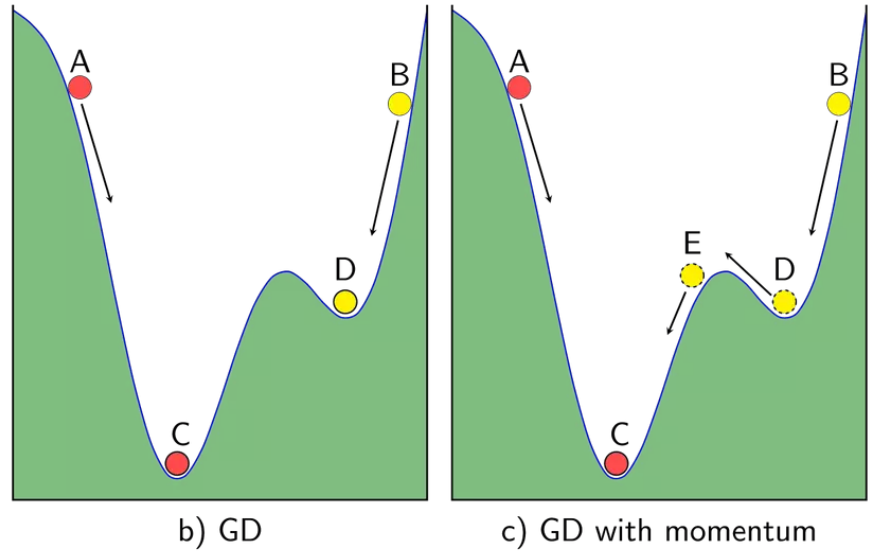
Nhược điểm:

* Đường đi tới điểm cực tiểu bị ảnh hưởng nhiều hơn bởi dữ liệu nhiễu
* Vẫn chịu ảnh hưởng trực tiếp bởi learning rate

### Momentum và các biến thể

#### Momentum

Phương pháp Momentum khắc phục các hạn chế của GD bằng cách thêm vào giá trị vận tốc (đà) nhằm giúp thuật toán GD vượt qua các điểm tối thiểu cục bộ nhanh hơn.



Công thức:

* là giá trị của trọng số mới
* lá giá trị của trọng số hiện tại
* là learning rate
* là đạo hàm của hàm số tại trọng số đó
* là tham số điều chỉnh vận tốc (đà)
* là giá trị vận tốc trước đó của trọng số w trước đó

Ưu điểm:

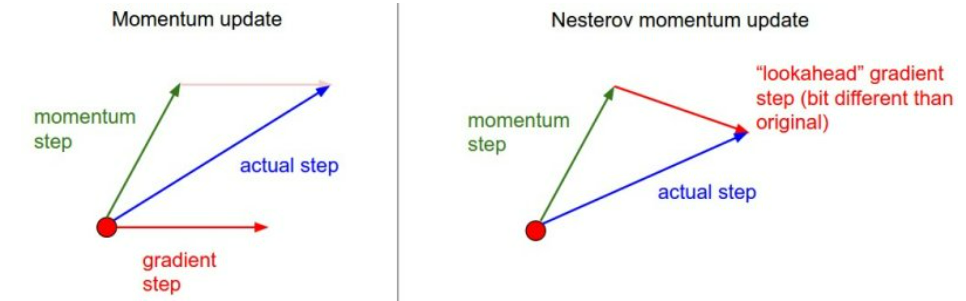
* Giúp GD vượt qua các điểm local minium nhanh hơn và tiến tới global minium dễ hơn

Nhược điểm:

* Bởi vì có thêm vận tốc (đà) nên sẽ dao động quanh global minium làm mất thời gian để triệt tiêu vận tốc và dừng lại.

#### Nesterov accelerated gradient (NAG) (Nesterov Momentum)

Thuật toán momentum tuy giúp GD vượt qua các local minium dễ hơn, tuy nhiên do có vận tốc nên việc dừng lại ở global minium tốn rất nhiều thời gian. Thuật toán NAG được cải thiện từ thuật toán momentum bằng cách tính giá trị GD của điểm tiếp theo.



### Một số Optimazier khác

#### Adagard

Khác với các phương pháp trước đó (giữ nguyên learning rate), adagard xem learning rate là 1 tham số và điều chỉnh nó sau mỗi bước t.

Tham số có độ dốc cao hơn hoặc cập nhật thường xuyên sẽ có tốc độ học chậm hơn.

Các tham số có độ dốc thấp hoặc cập nhật không thường xuyên sẽ có tốc độ học nhanh hơn để chúng được huấn luyện nhanh chóng.

Khi tổng các gradient trong quá khứ bình phương có giá trị cao, về nó sẽ chia tốc độ học cho một giá trị cao, do đó tốc độ học sẽ trở nên ít hơn. Ngược lại với gradient thấp.

Tốc độ học tỷ lệ nghịch với tổng bình phương của tất cả các gradient trước đó của tham số.

Công thức

* là giá trị của trọng số mới
* lá giá trị của trọng số hiện tại
* là learning rate
* là đạo hàm của hàm số tại trọng số đó
* là ma trận chéo mà mỗi phần tử trên đường chéo (i,i) là bình phương của đạo hàm vectơ tham số tại thời điểm t.
* là giá trị tránh lỗi (tránh trường hợp Gt=0)

Ưu điểm:

* Tự điều chỉnh learning rate cho từng tham số
* Không còn nhạy cảm với learning rate

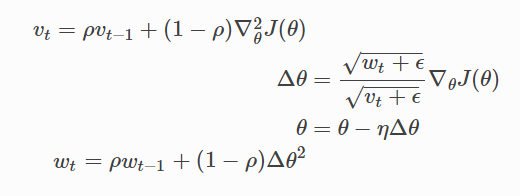
Nhược điểm:

* Tốc độ hội tụ chậm
* Có thể bị mắc kẹt tại các điểm stationary point

#### Adadelta

Adadelta, một thuật toán giảm độ dốc ngẫu nhiên, cung cấp một cải tiến đáng kể về tốc độ hội tụ so với Adagrad. Thay vì tính tất cả các gradient trước đó của tham số, adadelta theo dõi và tính toán dựa trên thông tin động về các cập nhật gradient gần đây.

Công thức:



Ưu điểm:

* Tự điều chỉnh learning rate cho từng tham số
* Không nhạy cảm với learning rate
* Tốc độ hội tụ nhanh hơn Adagrad

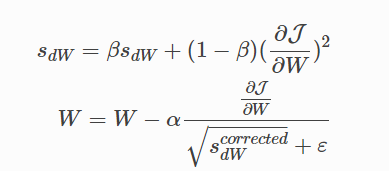
Nhược điểm:

* Tốn kém bộ nhớ hơn Adagrad

#### RMSprop

Root Mean Square Prop (RMSProp) cải thiện tốc độ hội tụ của Adagrad bằng việc bằng cách giữ mức trung bình theo cấp số nhân của bình phương của các gradient trong quá khứ sau đó chia tốc độ học cho mức trung bình này để tăng tốc độ hội tụ.

Công thức:



* S trung bình có trọng số theo cấp số nhân của các bình phương độ dốc trong quá khứ
* độ dốc chi phí của lớp hiện tại
* W trọng số hiện tại
* β tham số điều chỉnh S
* α learning rate
* ϵ eluavation (giá trị nhỏ để tránh lỗi chia cho 0)

Ưu điểm:

* Tự điều chỉnh learning rate cho từng tham số
* Không nhạy cảm với learning rate
* Tốc độ hội tụ nhanh hơn Adagrad và Adadelta

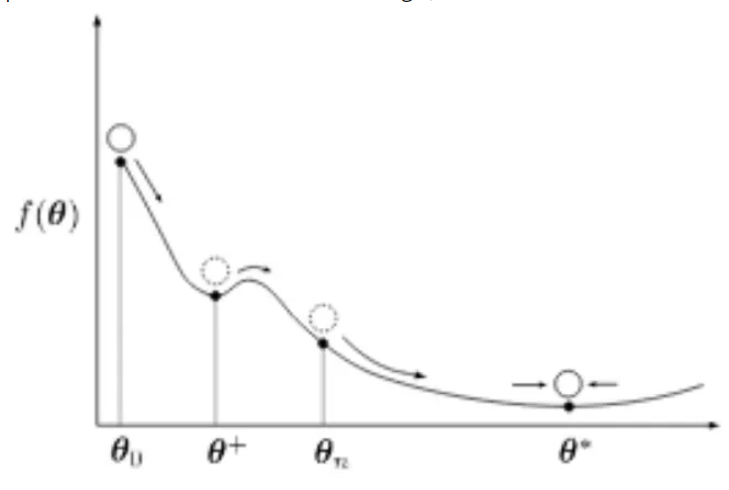
Nhược điểm:

* Có thể bị mắc kẹt tại các điểm stationary point

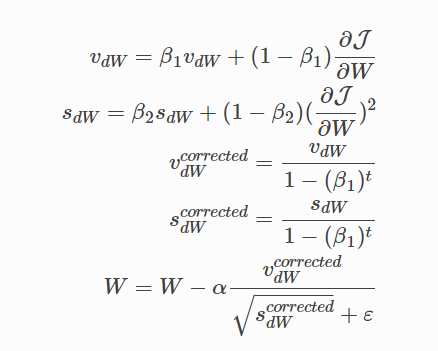
#### Adam

Adam kết hợp các ý tưởng từ cả RMSProp và momentum. Nó tính toán learning rate thích ứng cho từng tham số và hoạt động như sau:

* Đầu tiên, nó tính toán mức trung bình có trọng số theo cấp số nhân của độ dốc trong quá khứ (vdW).
* Thứ hai, nó tính giá trị trung bình theo cấp số nhân của bình phương của các gradient trong quá khứ (sdW).
* Thứ ba, các giá trị trung bình này có độ lệch về 0 và để tránh điều này, ta áp dụng hiệu chỉnh sai lệch.
* Cuối cùng, các tham số được cập nhật bằng cách sử dụng thông tin từ mức trung bình được tính toán.



Công thức:



* vdW - mức trung bình vận tốc
* sdW - mức trung bình của các bình phương độ dốc
* độ dốc chi phí của lớp hiện tại
* W trọng số hiện tại
* tham số điều chỉnh vận tốc v
* tham số điều chỉnh trung bình trọng số
* α learning rate
* ϵ eluavation (giá trị nhỏ để tránh lỗi chia cho 0)

Ưu điểm:

* Kết hợp các ưu điểm của Momentum, RMSprop và Adadelta
* Tự điều chỉnh learning rate cho từng tham số
* Không nhạy cảm với learning rate
* Tốc độ hội tụ nhanh
* Ít bị mắc kẹt tại các điểm stationary point

Nhược điểm:

* Có thể tốn kém bộ nhớ hơn các phương pháp khác

### Bảng so sánh

|  |  |  |  |
| --- | --- | --- | --- |
| Phương Pháp | Mục Đích | Ưu Điểm | Nhược Điểm |
| Gradient Descent (GD) | Tìm giá trị tối ưu bằng cách di chuyển theo hướng đối nghịch với độ dốc. | Đơn giản và dễ hiểu. Áp dụng cho nhiều loại hàm mục tiêu. | Có thể mắc kẹt tại cực đại hoặc cực tiểu cục bộ.  Chậm hội tụ và nhạy cảm với learning rate. |
| Stochastic Gradient Descent (SGD) | Áp dụng GD trên một mẫu dữ liệu ngẫu nhiên để giảm độ phức tạp khi làm việc với dữ liệu lớn. | Giải quyết được với cơ sở dữ liệu lớn. | Ảnh hưởng nhiều hơn bởi dữ liệu nhiễu.  Vẫn chịu ảnh hưởng trực tiếp từ learning rate. |
| Momentum | Vượt qua các điểm tối thiểu cục bộ nhanh hơn bằng cách thêm vận tốc. | Giúp GD vượt qua local minimum nhanh hơn. | Dao động quanh global minimum, làm mất thời gian để triệt tiêu vận tốc. |
| Nesterov Accelerated Gradient (NAG) | Cải thiện momentum bằng cách tính giá trị GD của điểm tiếp theo. | Cải thiện vấn đề dừng lại ở global minimum. | Đôi khi có thể không ổn định. |
| Adagrad | Điều chỉnh learning rate cho từng tham số dựa trên lịch sử của độ dốc. | Tự điều chỉnh learning rate cho từng tham số. | Tốc độ hội tụ chậm. Có thể mắc kẹt tại các điểm stationary point. |
| Adadelta | Cải thiện tốc độ hội tụ so với Adagrad bằng cách tính toán dựa trên thông tin động về các cập nhật gradient gần đây. | Tự điều chỉnh learning rate cho từng tham số. | Tốn kém bộ nhớ hơn Adagrad. |
| RMSprop | Cải thiện tốc độ hội tụ của Adagrad bằng cách giữ mức trung bình theo cấp số nhân của bình phương của các gradient trong quá khứ. | Tự điều chỉnh learning rate cho từng tham số.  Tốc độ hội tụ nhanh hơn Adagrad và Adadelta. | Có thể mắc kẹt tại các điểm stationary point. |
| Adam | Kết hợp các ý tưởng từ RMSProp và momentum, tính toán learning rate thích ứng cho từng tham số. | Kết hợp ưu điểm của Momentum, RMSprop, và Adadelta.  Tự điều chỉnh learning rate cho từng tham số.  Tốc độ hội tụ nhanh và ít bị mắc kẹt tại các điểm stationary point. | Có thể tốn kém bộ nhớ hơn các phương pháp khác. |

# Continual Learning và Test Production

## Continual learning

### Định nghĩa

Học liên tục là quá trình cập nhật liên tục mô hình máy học (ML) với dữ liệu mới để đáp ứng với những thay đổi trong phân phối dữ liệu trong môi trường sản xuất. Điều này giúp tránh hiện tượng "quên lãng" (catastrophic forgetting) và đảm bảo rằng mô hình duy trì hiệu suất cao khi dữ liệu đầu vào thay đổi.

### Mục đích

Mục đích chính của học liên tục là để đảm bảo rằng mô hình ML có thể thích ứng với môi trường sản xuất và duy trì hiệu suất tốt trong suốt vòng đời của nó. Môi trường sản xuất thường thay đổi theo thời gian, vì vậy mô hình cần được cập nhật thường xuyên để theo kịp những thay đổi này.

### Các cách triển khai

Có một số cách khác nhau để triển khai học liên tục.Một số cách phổ biến nhất bao gồm:

- Học có trạng thái : Trong học có trạng thái, mô hình được cập nhật liên tục với dữ liệu mới, nhưng kiến thức đã học trước đó vẫn được giữ lại. Nói cách khác, mô hình có thể nhớ những gì đã học được trong quá khứ và xây dựng dựa trên kiến thức đó khi học dữ liệu mới.

- Xử lý dữ liệu mới : Một cách khác để thực hiện học liên tục là tích hợp dữ liệu mới vào quá trình huấn luyện mô hình một cách liên tục. Điều này có thể được thực hiện theo thời gian thực hoặc theo lô.

- Đánh giá và kiểm thử : Học liên tục cũng bao gồm việc áp dụng các kỹ thuật kiểm thử liên tục để đảm bảo rằng hiệu suất của mô hình được duy trì. Điều này có thể được thực hiện bằng cách theo dõi hiệu suất của mô hình trên dữ liệu mới và thực hiện các điều chỉnh nếu cần thiết.

### Xây Dựng Hệ Thống Học Liên Tục

#### Thiết lập cơ sở hạ tầng cần thiết:

Để xây dựng hệ thống học liên tục, bạn cần thiết lập một số thành phần cơ sở hạ tầng cần thiết. Các thành phần này bao gồm:

* Cơ sở dữ liệu: Để lưu trữ dữ liệu mới được sử dụng cho việc huấn luyện mô hình.
* Hệ thống xử lý dữ liệu thời gian thực: Để xử lý dữ liệu mới theo thời gian thực và chuẩn bị dữ liệu cho mục đích huấn luyện.
* Lưu trữ mô hình: Để lưu trữ các phiên bản khác nhau của mô hình và theo dõi sự thay đổi của chúng theo thời gian.
* Hệ thống theo dõi và giám sát: Để theo dõi hiệu suất của mô hình và phát hiện bất kỳ vấn đề nào có thể xảy ra.

#### Xử lý dữ liệu

Một thành phần quan trọng khác của hệ thống học liên tục là cơ chế xử lý dữ liệu. Cơ chế này chịu trách nhiệm thu thập, xử lý và nhãn dữ liệu mới. Dữ liệu mới này sau đó được sử dụng để huấn luyện mô hình.

#### Cập nhật và triển khai mô hình

Khi mô hình đã được huấn luyện xong, nó cần được cập nhật và triển khai. Quá trình cập nhật và triển khai có thể được thực hiện theo cách thủ công hoặc tự động. Trong trường hợp cập nhật thủ công, người dùng phải chủ động tải xuống mô hình đã cập nhật và triển khai nó. Trong trường hợp cập nhật tự động, quá trình này được thực hiện tự động mà không cần sự can thiệp của người dùng.

#### Quản lý và kiểm soát

Để đảm bảo rằng hệ thống học liên tục hoạt động bình thường, bạn cần thiết lập các quy trình quản lý và kiểm soát. Các quy trình này bao gồm:

- Quản lý phiên bản mô hình: Để theo dõi các phiên bản khác nhau của mô hình và đảm bảo rằng chỉ có phiên bản mới nhất được triển khai.

- Thực hiện các biện pháp an ninh: Để bảo vệ hệ thống khỏi các cuộc tấn công mạng và các mối đe dọa khác.

- Quản lý rủi ro: Để xác định và giảm thiểu các rủi ro có thể xảy ra trong quá trình học liên tục.

### Các Giai Đoạn Xây Dựng

Có bốn giai đoạn chính trong quá trình xây dựng hệ thống học liên tục:

- Giai đoạn 1: Huấn luyện lại thủ công, không lưu trạng thái:

* Trong giai đoạn này, mô hình được huấn luyện lại từ đầu mỗi khi có dữ liệu mới. Quá trình huấn luyện lại này không lưu trạng thái, vì vậy mô hình phải học lại mọi thứ từ đầu.

- Giai đoạn 2 : Huấn luyện tự động:

* Trong giai đoạn này, mô hình được huấn luyện lại tự động khi có dữ liệu mới. Tuy nhiên, quá trình huấn luyện lại này vẫn không lưu trạng thái.

- Giai đoạn 3: Huấn luyện tự động, có trạng thái:

* Trong giai đoạn này, mô hình được huấn luyện lại tự động khi có dữ liệu mới và quá trình huấn luyện lại này có lưu trạng thái. Điều này cho phép mô hình nhớ những gì đã học được trong quá khứ và xây dựng dựa trên kiến thức đó khi học dữ liệu mới.

- Giai đoạn 4: Học liên tục:

* Trong giai đoạn này, hệ thống học liên tục được thiết lập hoàn chỉnh và mô hình được cập nhật liên tục với dữ liệu mới. Mô hình có thể thích ứng với môi trường sản xuất và duy trì hiệu suất cao trong suốt vòng đời của nó.

### Xác Định Tần Suất Cập Nhật Mô Hình

Tần suất cập nhật mô hình phụ thuộc vào một số yếu tố, bao gồm:

- Tốc độ thay đổi của môi trường sản xuất

- Khối lượng dữ liệu mới được tạo ra

- Tài nguyên tính toán có sẵn

Để xác định tần suất cập nhật mô hình phù hợp, bạn có thể thực hiện các bước sau:

- Thực hiện thử nghiệm về giá trị của sự tươi mới của dữ liệu: Thu thập dữ liệu mới và huấn luyện lại mô hình với dữ liệu này. Đánh giá sự cải thiện trong hiệu suất của mô hình khi được huấn luyện với dữ liệu mới hơn.

- Tính toán sự cải thiện trong hiệu suất: Tính toán sự cải thiện trong hiệu suất khi mô hình được cập nhật với dữ liệu mới hơn.

## Test Production

Thử nghiệm sản xuất trong nghiên cứu học liên tục trong Machine Learning là một phần quan trọng của quá trình triển khai và duy trì các mô hình máy học trong môi trường sản xuất. Nó giúp đảm bảo rằng các mô hình đáp ứng yêu cầu hiệu suất và có khả năng thích nghi với dữ liệu mới.

### Các cách kiểm tra phổ biến trong thử nghiệm sản xuất

Có một số cách kiểm tra phổ biến trong thử nghiệm sản xuất, bao gồm:

Shadow Deployment (Triển khai ẩn): Mô hình mới được triển khai song song với mô hình hiện có trên cơ sở hạ tầng sản xuất, nhưng các dự đoán của nó không được phục vụ trực tiếp cho người dùng.

A/B Testing (Kiểm Tra A/B): Người dùng được chia thành hai nhóm ngẫu nhiên: Nhóm A tiếp tục nhận dự đoán từ mô hình hiện có, trong khi Nhóm B nhận dự đoán từ mô hình mới.

Canary Release (Triển Khai Canary): Mô hình mới được triển khai cho một phần nhỏ và cụ thể của người dùng, gọi là "bản canary." Phần lớn người dùng vẫn tiếp tục sử dụng mô hình hiện có.

Interleaving Experiments (Thử Nghiệm Xen Kẽ): Một nhóm người dùng nhận được gợi ý từ mô hình hiện có, trong khi nhóm khác nhận gợi ý từ mô hình mới. Sự lựa chọn của người dùng được theo dõi để xác định mô hình nào hoạt động tốt hơn.

### Ý nghĩa của thử nghiệm sản xuất trong nghiên cứu học liên tục

Thử nghiệm sản xuất trong nghiên cứu học liên tục có một số ý nghĩa quan trọng, bao gồm:

Đảm bảo ổn định sản phẩm: Khi triển khai mô hình trong môi trường sản xuất, điều quan trọng là đảm bảo rằng mô hình hoạt động ổn định và không gây ra các vấn đề không mong muốn như lỗi hoặc hiệu suất kém.

Đánh giá học liên tục: Thử nghiệm sản xuất cung cấp cơ hội để đánh giá hiệu suất của mô hình máy học trong thời gian thực và xác định xem liệu mô hình có thể thích nghi và học từ dữ liệu mới không.

Phát hiện sớm sự cố: Bằng cách thực hiện thử nghiệm trong quá trình sản xuất, ta có thể phát hiện sớm các vấn đề hoặc biểu hiện của mô hình máy học và thực hiện các biện pháp sửa chữa hoặc điều chỉnh khi cần thiết.

### Quá trình thử nghiệm sản xuất

Quá trình thử nghiệm sản xuất trong nghiên cứu học liên tục thường bao gồm các bước sau:

* Triển khai mô hình: Bắt đầu bằng việc triển khai mô hình máy học trong môi trường sản xuất. Điều này bao gồm việc tích hợp mô hình vào hệ thống thực tế và bắt đầu sử dụng nó để đưa ra dự đoán.
* Sử dụng thực tế: Mô hình sẽ được sử dụng để xử lý dữ liệu thực tế và đưa ra dự đoán. Dữ liệu đầu vào mới liên tục được đưa vào hệ thống.
* Giám sát hiệu suất: Các hệ thống giám sát được sử dụng để theo dõi hiệu suất của mô hình trong môi trường sản xuất. Điều này bao gồm việc ghi lại các metric hiệu suất và theo dõi sự thay đổi của chúng theo thời gian.
* Thử nghiệm và kiểm tra: Thử nghiệm và kiểm tra định kỳ được thực hiện để đảm bảo rằng mô hình hoạt động đúng cách và đáp ứng các tiêu chuẩn hiệu suất.
* Cập nhật và điều chỉnh: Nếu mô hình hiển thị bất kỳ dấu hiệu không bình thường hoặc hiệu suất không đạt yêu cầu, quy trình cập nhật và điều chỉnh sẽ được thực hiện. Điều này có thể bao gồm việc điều chỉnh siêu tham số của mô hình hoặc cập nhật dữ liệu đào tạo.

### Những thách thức trong thử nghiệm sản xuất trong nghiên cứu học liên tục

Một số thách thức trong thử nghiệm sản xuất trong nghiên cứu học liên tục bao gồm:

* Tích hợp liên tục: Cần xem xét cách triển khai và tích hợp liên tục các phiên bản cập nhật của mô hình vào môi trường sản xuất mà không gây gián đoạn quá lớn.
* Sử dụng lượng dữ liệu nhỏ: Trong học liên tục, có thể xảy ra tình huống khi lượng dữ liệu mới có hạn. Việc thử nghiệm trong điều kiện này đòi hỏi các phương pháp đặc biệt để ước tính hiệu suất mô hình.
* Quản lý chi phí tính toán: Thực hiện thử nghiệm sản xuất có thể đòi hỏi nhiều tính toán, do đó, quản lý chi phí tính toán là một yếu tố quan trọng.

### Kết luận

Thử nghiệm sản xuất trong nghiên cứu học liên tục trong Machine Learning là một phần quan trọng của quá trình triển khai và duy trì các mô hình máy học trong môi trường sản xuất. Nó giúp đảm bảo rằng các mô hình đáp ứng yêu cầu hiệu suất và có khả năng thích nghi với dữ liệu mới. Tuy có thách thức, nhưng việc thực hiện thử nghiệm sản xuất là cần thiết để đảm bảo tính ổn định và hiệu suất của các mô hình máy học trong thực tế.

# TÀI LIỆU THAM KHẢO

Tiếng Việt

[Trần Trung Trực (2020) Optimizer- Viblo](https://viblo.asia/p/optimizer-hieu-sau-ve-cac-thuat-toan-toi-uu-gdsgdadam-Qbq5QQ9E5D8)

[Tiep Vu Huu (2017) - MachineLearningCoBan](https://machinelearningcoban.com/2017/01/12/gradientdescent/)

Tiếng Anh

[Designing Machine Learning Cheatsheet( 2017) PUBLISHED BY: Brendan Fortuner](https://ml-cheatsheet.readthedocs.io/en/latest/optimizers.html#toc-entry-2)