TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ

MÔN NHẬP MÔN HỌC MÁY

BÁO CÁO CUỐI KỲ

Người hướng dẫn: **PGS.TS. LÊ ANH CƯỜNG**

Người thực hiện: **TRẦN TỐNG GIA VŨ - 52000733**

Lớp **: 20050201**

Khoá **: 24 - 25**

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

BÁO CÁO CUỐI KỲ

MÔN NHẬP MÔN HỌC MÁY

BÁO CÁO CUỐI KỲ

Người hướng dẫn: **PGS.TS. LÊ ANH CƯỜNG**

Người thực hiện: **TRẦN TỐNG GIA VŨ - 52000733**

Lớp **: 20050201**

Khoá **: 24 - 25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Để hoàn thành đồ án này trước tiên chúng em xin gửi đến các quý thầy, cô giảng viên trường Đại học Tôn Đức Thắng lời cảm ơn chân thành và sâu sắc nhất. Đặc biệt, em xin gửi đến thầy Lê Anh Cường – người đã tận tình hướng dẫn, giúp đỡ chúng em hoàn thành đề tài này lời cảm ơn sâu sắc nhất.

Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế của nhóm, bài báo cáo chưa được hoàn thiện và còn những thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các quý thầy cô để em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công việc thực tế sau này.

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 và được sự hướng dẫn của PGS.TS. 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 chúng tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 14 tháng 12 năm 2023

Tác giả

(ký tên và ghi rõ họ tên)

Trần Tống Gia Vũ

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)

MỤC LỤC

MỤC LỤC HÌNH ẢNH, BẢNG BIỂU

MỤC LỤC BẢNG BIỂU:

MỤC LỤC HÌNH ẢNH:

CHƯƠNG 1 - TÌM HIỂU LÝ THUYẾT

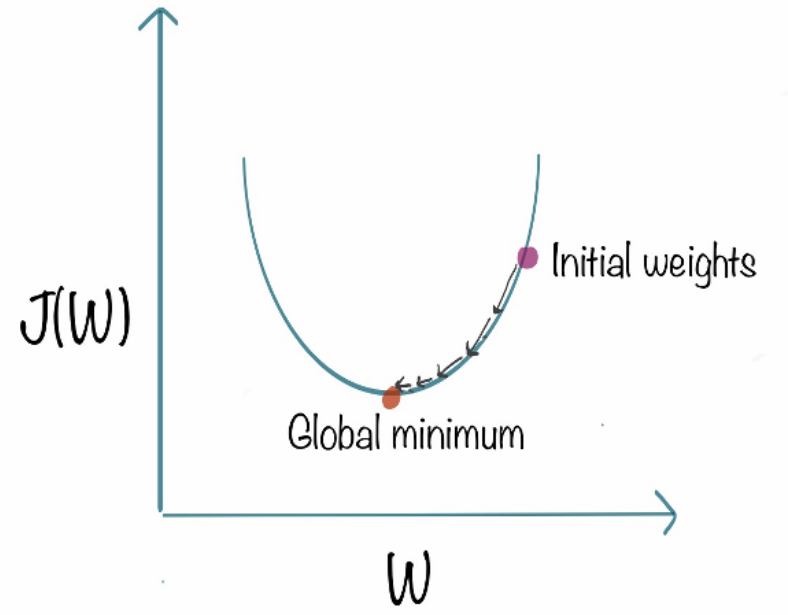
1.1 Phương pháp Optimizer trong huấn luyện mô hình học máy

Phương pháp Optimize *(thuật toán tối ưu)* là cơ sở để xây dựng mô hình neural network với mục đích "học " được các features (hay pattern) của dữ liệu đầu vào, từ đó có thể tìm 1 cặp weights và bias phù hợp để tối ưu hóa model, giúp cải thiện hiệu suất của mô hình học sâu.

Những thuật toán này sẽ cho phép các neural networks có thể "học hỏi" từ dữ liệu, bằng cách cập nhật lặp đi lặp lại các cặp weights và bias. Mỗi thuật toán tối ưu đều có những quy tắc, learning rates, và momentum để tìm ra các tham số mô hình tối ưu nhằm cải thiện hiệu suất.

1.1.1 Gradient Descent

- **Khái niệm:** Gradient Descent là một thuật toán tối ưu hóa lặp lại nhằm cố gắng tìm giá trị tối ưu (Tối thiểu/Tối đa) của hàm mục tiêu. Đây là một trong những kỹ thuật tối ưu hóa được sử dụng nhiều nhất trong các dự án học máy để cập nhật các tham số của mô hình nhằm giảm thiểu hàm chi phí.
- **Mục đích:** là tìm ra các tham số tốt nhất của mô hình mang lại độ chính xác cao nhất cho các tập dữ liệu huấn luyện cũng như kiểm tra. Trong Gradient Descent, gradient là một vectơ chỉ hướng tăng mạnh nhất của hàm tại một điểm cụ thể. Di chuyển theo hướng ngược lại của gradient cho phép thuật toán giảm dần về các giá trị thấp hơn của hàm và cuối cùng đạt đến mức tối thiểu của hàm.
- **Các bước thực hiện:**
    - **Bước 1:** đầu tiên chúng ta khởi tạo các tham số *(trọng số, learning rate, số vòng lặp)* của mô hình một cách ngẫu nhiên
    - **Bước 2:** Tính gradient của lost function đối với từng tham số, và cập nhật trọng số của mô hình.
    - **Bước 3:** Quá trình cập nhật trọng số và giảm giá trị hàm mất mát được lặp lại cho đến khi đạt được điều kiện dừng *(có thể là khi đạt được Global minimum)*



- **Mã giả:**

```
t ← 0
max_iterations ← 1000
w, b ← initialize randomly

while t < max_iterations do
    t ← t + 1
    w_t+1 ← w_t - η ∇w_t
    b_t+1 ← b_t - η ∇b_t
end
```

Trong đó:

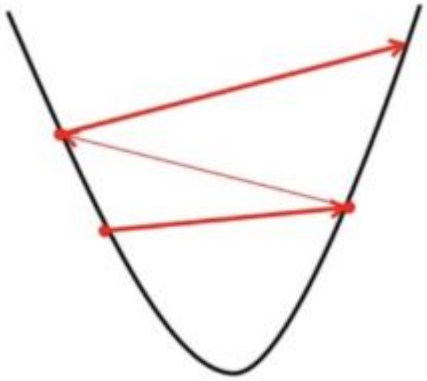
- max_iterations là số lần lặp chúng ta muốn thực hiện để cập nhật tham số của mình

- W,b là các weight và bias
- η là tham số học cũng được ký hiệu là alpha

★ **Chú ý khi chọn Learning rate:**

Việc lựa chọn tốc độ học chính xác là rất quan trọng vì nó đảm bảo rằng gradient Descent hội tụ trong một thời gian hợp lý. :

- **Nếu chúng ta chọn α rất lớn**: Gradient Descent có thể vượt quá mức tối thiểu. Nó có thể không hội tụ hoặc thậm chí phân kỳ.



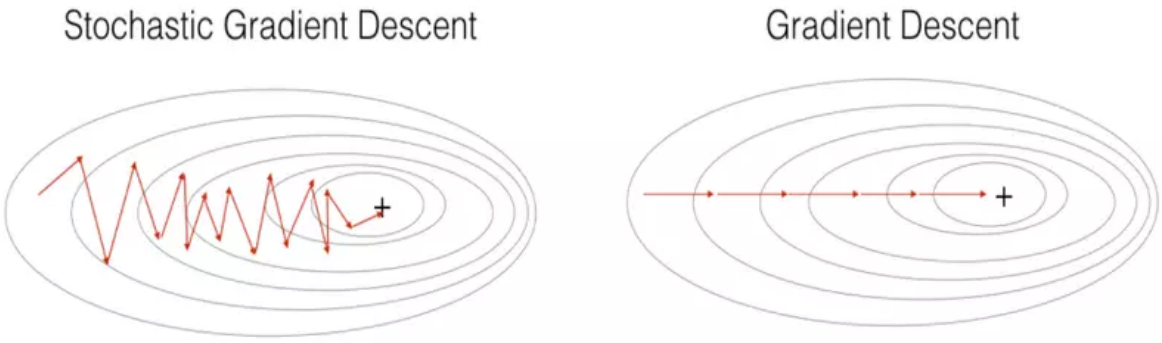
- **Nếu chúng ta chọn α rất nhỏ**: Gradient Descent sẽ thực hiện các bước nhỏ để đạt cực tiểu cục bộ và sẽ mất nhiều thời gian hơn để đạt cực tiểu.

- **Ưu điểm:**
  - **Tính linh hoạt:** Gradient Descent có thể được sử dụng với các hàm chi phí khác nhau và có thể xử lý các vấn đề hồi quy phi tuyến tính.
  - **Khả năng mở rộng:** Gradient Descent có thể mở rộng thành các tập dữ liệu lớn vì nó cập nhật các tham số cho từng ví dụ huấn luyện cùng một lúc.
  - **Hội tụ:** Gradient Descent có thể hội tụ đến mức tối thiểu toàn cầu của hàm chi phí, miễn là tốc độ học được đặt phù hợp.

- **Nhược điểm:**
    - **Độ nhạy đối với tốc độ học: Việc** lựa chọn tốc độ học có thể rất quan trọng trong Gradient Descent vì việc sử dụng tốc độ học cao có thể khiến thuật toán vượt quá mức tối thiểu, trong khi tốc độ học thấp có thể khiến thuật toán hội tụ chậm.
    - **Hội tụ chậm:** Gradient Descent c có thể yêu cầu nhiều lần lặp hơn để hội tụ đến mức tối thiểu vì nó cập nhật lần lượt các tham số cho từng ví dụ huấn luyện.
    - **Dễ bị gây nhiễu:** Các bản cập nhật trong gradient Descent gây nhiễu và có độ chênh lệch cao, điều này có thể khiến quá trình tối ưu hóa kém ổn định hơn và dẫn đến dao động quanh mức tối thiểu.
    - Phụ thuộc vào nghiệm khởi tạo ban đầu và learning rate.
    - Ví dụ 1 hàm số có 2 global minimum, thì tùy thuộc vào 2 điểm khởi tạo ban đầu sẽ cho ra 2 nghiệm cuối cùng khác nhau.
    - Tốc độ học quá lớn sẽ khiến cho thuật toán không hội tụ, quanh quẩn bên đích vì bước nhảy quá lớn; hoặc tốc độ học nhỏ ảnh hưởng đến tốc độ training

1.1.2 Stochastic Gradient Descent

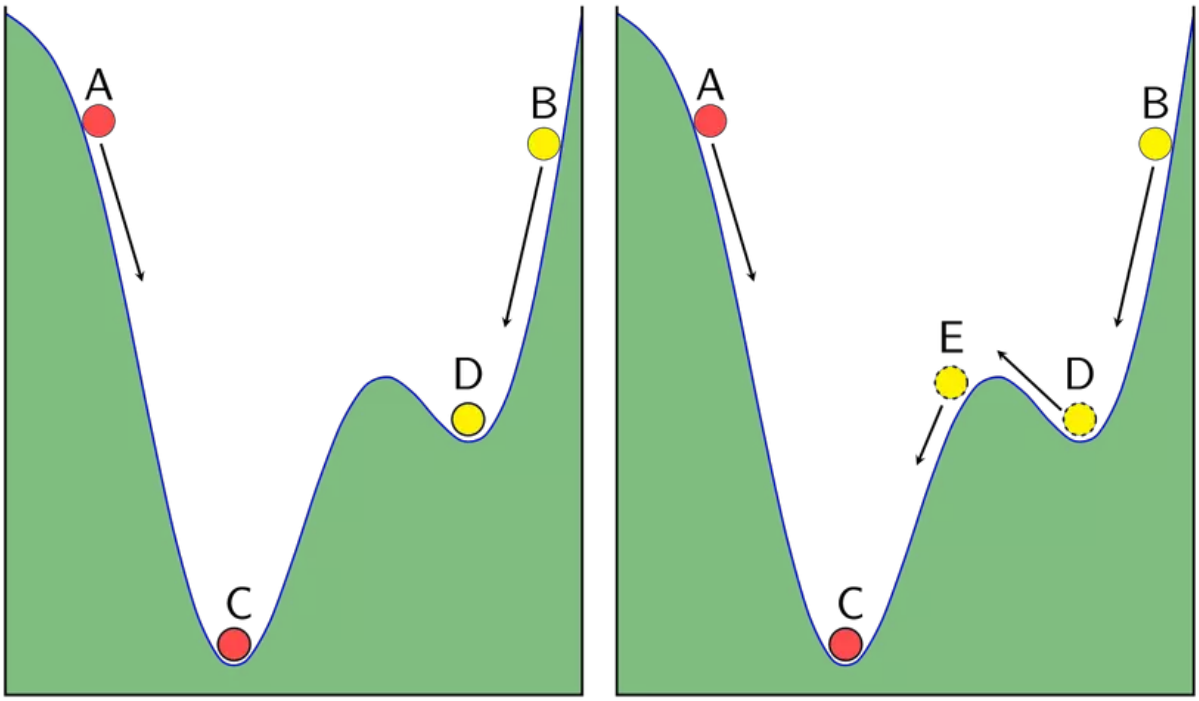
- **Khái niệm:** Stochastic Gradient Descent (SGD) là một thuật toán tối ưu hóa lặp thường được sử dụng trong học máy và học sâu. Đây là một biến thể của Gradient Descent, thực hiện cập nhật các tham số mô hình (trọng số) dựa trên độ dốc của hàm mất mát, được tính toán trên một tập hợp con được chọn ngẫu nhiên của dữ liệu huấn luyện, thay vì trên toàn bộ tập dữ liệu. *(giúp xử lý các tập dữ liệu lớn trong các dự án học máy)*
- **Ý tưởng cơ bản của SGD:** là lấy mẫu một tập hợp con ngẫu nhiên nhỏ của dữ liệu huấn luyện, được gọi là lô nhỏ và tính toán độ dốc của hàm mất đối với các tham số mô hình chỉ sử dụng tập hợp con đó. Độ dốc này sau đó được sử dụng để cập nhật các tham số. Quá trình này được lặp lại với một lô nhỏ ngẫu nhiên mới cho đến khi thuật toán hội tụ hoặc đạt đến tiêu chí dừng được xác định trước.
- **Cách hoạt động:** Thay vì sau mỗi epoch chúng ta sẽ cập nhật trọng số (Weight) 1 lần thì trong mỗi epoch có N điểm dữ liệu chúng ta sẽ cập nhật trọng số N lần. Nhìn vào 1 mặt , SGD sẽ làm giảm đi tốc độ của 1 epoch. Tuy nhiên nhìn theo 1 hướng khác,SGD sẽ hội tụ rất nhanh chỉ sau vài epoch. Công thức SGD cũng tương tự như GD nhưng thực hiện trên từng điểm dữ liệu



- **Ưu điểm:**
  - **Tốc độ:** SGD nhanh hơn các biến thể khác của Gradient Descent hàng loạt, và Gradient Descent hàng loạt vì nó chỉ sử dụng một ví dụ để cập nhật các tham số.
  - **Hiệu quả bộ nhớ:** Vì SGD cập nhật từng tham số cho từng ví dụ huấn luyện nên nó tiết kiệm bộ nhớ và có thể xử lý các tập dữ liệu lớn không vừa bộ nhớ.
  - **Tránh cực tiểu cục bộ:** Do các cập nhật ồn ào trong SGD, nó có khả năng thoát khỏi cực tiểu cục bộ và hội tụ đến mức tối thiểu toàn cầu.
- **Nhược điểm:**
  - **Hội tụ chậm**: SGD có thể yêu cầu nhiều lần lặp hơn để hội tụ đến mức tối thiểu vì nó cập nhật từng tham số cho từng ví dụ huấn luyện tại một thời điểm.
  - **Độ nhạy đối với tốc độ học:** Việc lựa chọn tốc độ học có thể rất quan trọng trong SGD vì việc sử dụng tốc độ học cao có thể khiến thuật toán vượt quá mức tối thiểu, trong khi tốc độ học thấp có thể khiến thuật toán hội tụ chậm.

1.1.3 Momentum

- **Khái niệm:** Momentum-based Gradient Descent là một biến thể của phương pháp tối ưu hóa Gradient Descent được thiết kế để giảm tốc độ hội tụ của thuật toán và tránh được mô hình "mắc kẹt" trong các vùng địa phương của không gian tham số. *(bằng cách truyền thêm một động lực để vượt qua global minimum)*



b) GD c) GD with momentum

- **Ưu điểm:** Thuật toán tối ưu giải quyết được vấn đề: Gradient Descent không tiến được tới điểm global minimum mà chỉ dừng lại ở local minimum. Ngoài ra, nó có thể giúp trình tối ưu hóa hội tụ nhanh hơn và đạt mức tối thiểu cục bộ tốt hơn
- **Nhược điểm:** nó vẫn mất khá nhiều thời gian giao động qua lại trước khi dừng hẳn *(Hội tụ chậm)*

1.1.4 Mini-Batch Gradient Descent

- **Cách hoạt động:** thay vì lấy tất cả dữ liệu huấn luyện, chỉ một tập hợp con của tập dữ liệu được sử dụng để tính hàm mất mát.
- **Ưu điểm:**
  - **Nhanh hơn:** Vì chúng ta đang sử dụng một loạt dữ liệu thay vì lấy toàn bộ tập dữ liệu nên cần ít lần lặp hơn. Đó là lý do tại sao thuật toán giảm độ dốc hàng loạt nhỏ nhanh hơn cả thuật toán giảm độ dốc ngẫu nhiên và thuật toán giảm độ dốc hàng loạt.
  - **Hiệu năng tốt hơn:** Vì thuật toán sử dụng tính năng bó, tất cả dữ liệu huấn luyện không cần phải được tải vào bộ nhớ, do đó giúp quá trình thực hiện hiệu quả hơn.

- **Nhược điểm:**
    - Tùy thuộc vào batch size *(batch size 32 thường được coi là phù hợp với hầu hết mọi trường hợp)*, các bản cập nhật có thể được thực hiện ít nhiễu hơn.

1.1.5 Adagrad (Adaptive Gradient Descent)

- **Cách hoạt động:** Adagrad sử dụng các learning rate khác nhau cho mỗi lần lặp. Sự thay learning rate phụ thuộc vào sự khác biệt của các tham số trong quá trình huấn luyện. Các tham số càng thay đổi thì learning rate càng thay đổi nhỏ. Việc sửa đổi này rất có lợi vì các bộ dataset thực tế, lúc thì chứa nhiều feature, lúc thì lại chứa ít. Vì vậy, thật không công bằng khi có cùng một giá trị tốc độ học cho tất cả các đặc tính.

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

Trong đó :

- n : hằng số
- gt : gradient tại thời điểm t
- ϵ : hệ số tránh lỗi (chia cho mẫu bằng 0)
- G : 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.

- **Ưu điểm:** loại bỏ nhu cầu sửa đổi learning rate theo cách thủ công. Nó đáng tin cậy hơn các thuật toán giảm độ dốc và các biến thể của chúng, đồng thời đạt được sự hội tụ ở tốc độ cao hơn.
- **Nhược điểm:** nó làm giảm tốc độ học tập một cách mạnh mẽ và đơn điệu. Có thể có một thời điểm khi tốc độ học tập trở nên cực kỳ nhỏ. Điều này là do gradient bình phương ở mẫu số tiếp tục tích lũy và do đó phần mẫu số tiếp tục tăng. Do tốc độ học tập nhỏ, mô hình cuối cùng không thể thu được nhiều kiến thức hơn và do đó độ chính xác của mô hình bị ảnh hưởng. *(tổng bình phương biến thiên sẽ lớn dần theo thời gian cho đến khi nó làm tốc độ học cực kì nhỏ, làm việc training trở nên đóng băng.)*

1.1.6 RMS Prop (Root Mean Square)

- Giống như Adagrad, RMSProp điều chỉnh tốc độ học của từng tham số trong quá trình đào tạo. Tuy nhiên, thay vì tích lũy tất cả các gradient trong quá khứ như Adagrad, RMSProp tính toán mức trung bình động của các gradient bình phương.

- Thuật toán chủ yếu tập trung vào việc đẩy nhanh quá trình tối ưu hóa bằng cách giảm số lượng đánh giá hàm để đạt mức tối thiểu cục bộ. Thuật toán giữ giá trị trung bình động của các gradient bình phương cho mọi trọng số và chia gradient cho căn bậc hai của bình phương trung bình. Điều này cho phép thuật toán điều chỉnh tốc độ học trơn tru hơn và ngăn tốc độ học giảm quá nhanh.
- **Ưu điểm:** có thể xử lý các mục tiêu không cố định, trong đó chức năng cơ bản mà mạng nơ-ron đang cố gắng ước tính những thay đổi theo thời gian. Trong những trường hợp này, Adagrad có thể hội tụ quá nhanh, nhưng RMSProp có thể điều chỉnh tốc độ học cho phù hợp với hàm mục tiêu đang thay đổi.
- **Nhược điểm:** tốc độ học phải được xác định theo cách thủ công

1.1.7 Adam Optimizer

- Không giống như SGD duy trì một tốc độ học tập duy nhất trong suốt quá trình đào tạo, trình tối ưu hóa Adam tự động tính toán tốc độ học tập của từng cá nhân dựa trên độ dốc trong quá khứ và khoảnh khắc thứ hai của chúng
- Bằng cách kết hợp cả khoảnh khắc đầu tiên (trung bình) và khoảnh khắc thứ hai (phương sai không tập trung) của độ dốc, trình tối ưu hóa Adam đạt được tốc độ học thích ứng có thể điều hướng bối cảnh tối ưu hóa một cách hiệu quả trong quá trình đào tạo. Khả năng thích ứng này giúp hội tụ nhanh hơn và cải thiện hiệu suất của mạng lưới thần kinh.
- **Ưu điểm:** dễ thực hiện, có thời gian chạy nhanh hơn, yêu cầu bộ nhớ thấp và yêu cầu điều chỉnh ít hơn bất kỳ thuật toán tối ưu hóa nào khác. Khả năng hội tụ nhanh và xử lý các gradient nhiễu hoặc thưa thớt. Ngoài ra, nó không yêu cầu điều chỉnh thủ công các siêu tham số như suy giảm tốc độ học tập hoặc hệ số động lượng, khiến nó dễ sử dụng hơn các thuật toán tối ưu hóa khác.

1.1.8 Tổng kết:

Bảng 1.1 Ưu nhược điểm của một số thuật toán trên

Thuật toán	Ưu điểm	Nhược điểm
Stochastic Gradient Descent (SGD)	- Thực hiện đơn giản và hiệu quả về mặt tính toán. - Hiệu quả đối với các tập dữ liệu lớn với không gian đặc trưng nhiều chiều.	- SGD có thể bị kẹt ở mức cực tiểu địa phương. - Độ nhạy cao với tốc độ học tập ban đầu.
Momentum	- Giảm dao động trong quá trình tập luyện. - Hội tụ nhanh hơn cho các bài toán không điều kiện.	- Tăng độ phức tạp của thuật toán.
Adagrad	- Tỷ lệ học tập thích ứng cho mỗi tham số. - Hiệu quả đối với dữ liệu thưa thớt.	- Việc tích lũy gradient bình phương trong mẫu số có thể khiến tốc độ học giảm xuống quá nhanh. - Có thể dừng việc học quá sớm.
RMSProp	- Tốc độ học thích ứng trên mỗi tham số giúp hạn chế sự tích lũy độ dốc. - Hiệu quả đối với các mục tiêu không cố định.	- Có thể có tốc độ hội tụ chậm trong một số trường hợp.
Adam	- Hiệu quả và dễ thực hiện. - Áp dụng cho các tập dữ	- Yêu cầu điều chỉnh cẩn thận các siêu tham số.

	liệu lớn và mô hình nhiều chiều. - Khả năng khái quát hóa tốt.	

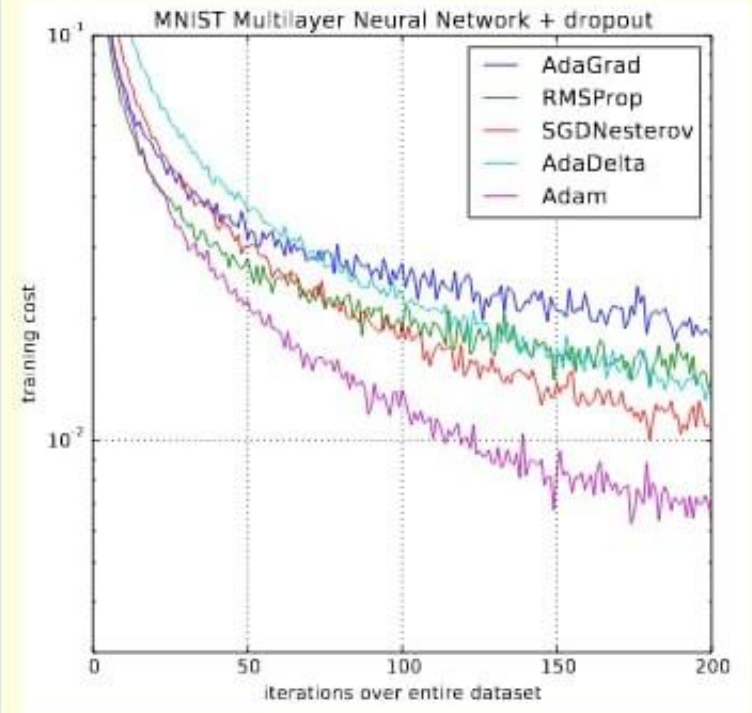


Fig. Optimizer comparison

1.2 Tìm hiểu về Continual Learning và Test Production

1.2.1 Continual Learning

- **Khái niệm:** là mô hình cho một số lượng lớn nhiệm vụ một cách tuần tự, mà không quên kiến thức thu được từ các nhiệm vụ trước, trong đó dữ liệu trong các nhiệm vụ cũ không còn nữa, trong quá trình đào tạo những cái mới.
- **Lý do:** giúp mô hình có thể theo kịp sự thay đổi phân phối dữ liệu. Có một số trường hợp sử dụng trong đó việc thích ứng nhanh chóng với việc thay đổi phân phối là rất quan trọng.
- **Ví dụ 1:** Để theo kịp nguồn cung cấp chương trình mới không ngừng nghỉ, cũng như sự thay đổi sở thích và xu hướng của khách hàng Netflix, dữ liệu đến phải được nhập liên tục. Và mô hình phải cập nhật liên tục để có khả năng đề xuất các chương trình hoặc phim phù hợp.

- **Ví dụ 2:** Khi mô hình của ta, phải đưa ra dự đoán cho người dùng mới (hoặc đã đăng xuất) không có dữ liệu lịch sử (hoặc dữ liệu đã lỗi thời). Nếu ta không điều chỉnh mô hình của mình, ngay khi nhận được một số dữ liệu từ người dùng đó, thì ta sẽ không thể đề xuất những điều liên quan cho người dùng đó.
- **Phân loại:** gồm 2 loại
    - **Stateless retraining:** Đào tạo lại mô hình từ đầu, nhưng sử dụng trọng số được khởi tạo ngẫu nhiên và dữ liệu mới. Tuy nhiên, có thể có một số trùng lặp với dữ liệu đã được sử dụng để huấn luyện phiên bản mô hình trước đó.
    - **Stateful training:** Đào tạo mô hình, với các trọng số từ lần training trước và dữ liệu mới. Nó cũng mang lại một số lợi ích:
        - Cho phép mô hình của ta cập nhật với lượng dữ liệu ít hơn đáng kể.
        - Cho phép mô hình của ta hội tụ nhanh hơn và sử dụng ít năng lượng tính toán hơn.
        - Về mặt lý thuyết, nó có thể tránh việc lưu trữ dữ liệu hoàn toàn sau khi dữ liệu đã được sử dụng để đào tạo (và để lại một khoảng thời gian an toàn). Về mặt lý thuyết, điều này giúp loại bỏ những lo ngại về quyền riêng tư dữ liệu.
        - Trên thực tế, hầu hết các công ty đều có thói quen theo dõi mọi thứ và không muốn vứt bỏ dữ liệu ngay cả khi nó không còn cần thiết nữa.
        - Thỉnh thoảng, ta sẽ cần phải chạy stateless retraining với một lượng lớn dữ liệu để hiệu chỉnh lại mô hình.
        - Sau khi cơ sở hạ tầng của ta được ổn định (tối ưu), việc thay đổi từ stateless retraining sang stateful training chỉ bằng một nút nhấn.

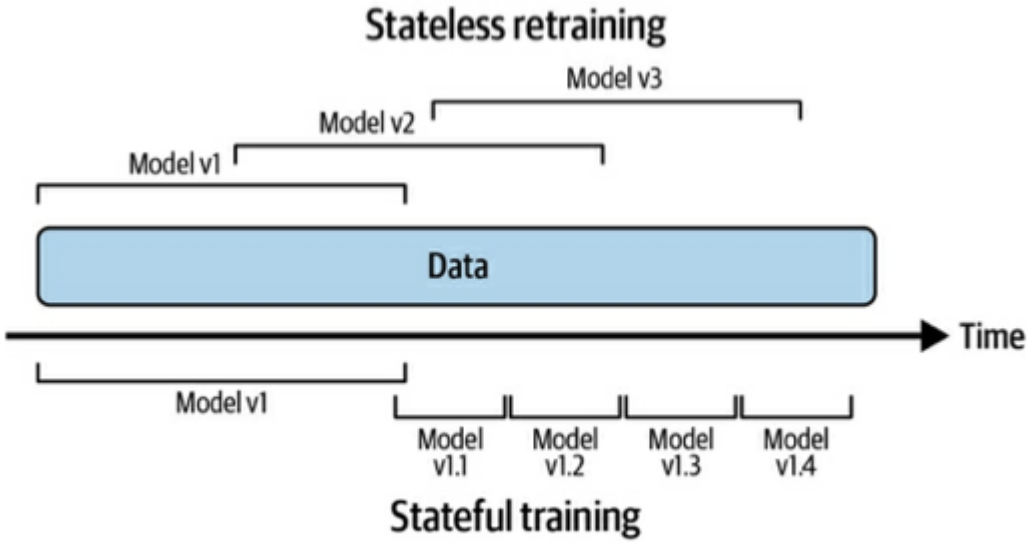


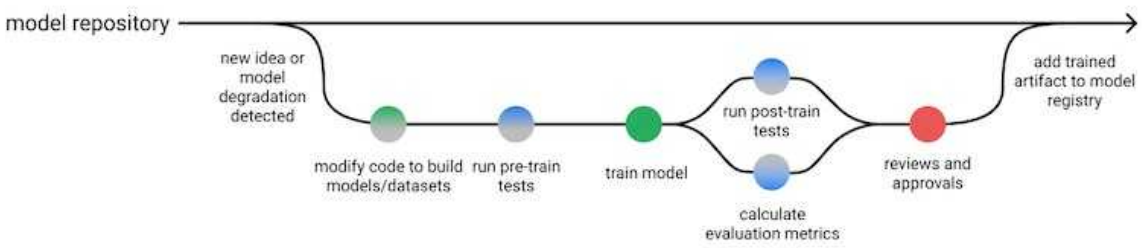
Figure 9-2. Stateless retraining versus stateful training

- **Cách hoạt động:** gồm 2 giai đoạn
  - Giai đoạn 1: stateless retraining

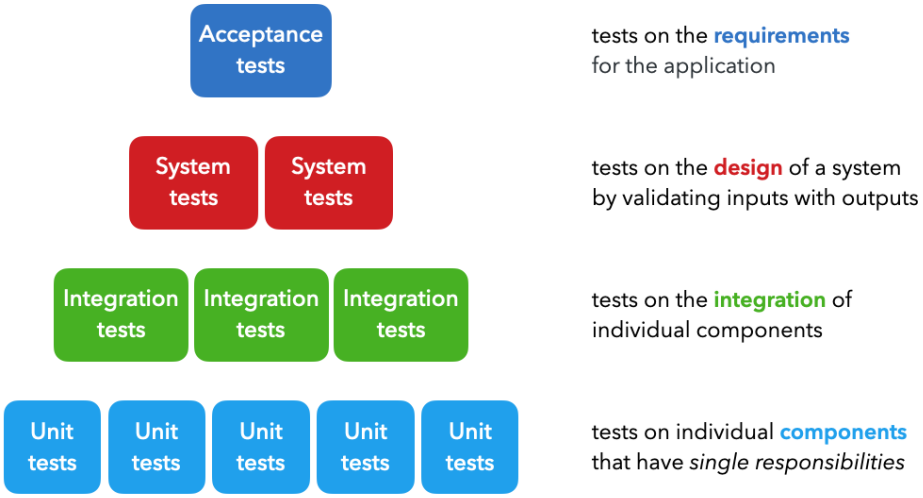
    Các mô hình chỉ được đào tạo lại khi đáp ứng hai điều kiện:
    - (1) hiệu suất của mô hình đã suy giảm đến mức hiện tại nó gây hại nhiều hơn là có lợi
    - (2) công ty có thời gian để cập nhật mô hình.
  - **Giai đoạn 2:** stateless retraining tự động theo lịch trình
    - Giai đoạn này thường xảy ra khi các mô hình chính của đã được phát triển, và do đó ưu tiên của ta không còn là tạo các mô hình mới, mà là duy trì và cải thiện các mô hình hiện có. Nhiệm vụ chính trong giai đoạn này, là lập lịch trình tự động training lại model theo định kỳ.

1.2.2 Test Production

- Khác với testing trong phần mềm (kiểm tra logic bằng văn bản), thì testing trong học máy, kiểm tra logic đã học.
- Các thử nghiệm Machine Learning (ML), có thể được chia thành testing và evaluation. Tiêu chí để kiểm thử mô hình, là chúng ta sẽ huấn luyện một mô hình, rồi đánh giá hiệu suất của nó trên bộ validation (có thể qua accuracy, hay qua biểu đồ trực quan,...)
- Kiểm tra là một cách để chúng ta đảm bảo rằng một cái gì đó hoạt động như dự định. Chúng ta được khuyến khích triển khai thử nghiệm và phát hiện nguồn lỗi càng sớm càng tốt trong chu kỳ phát triển, để có thể giảm chi phí hạ nguồn và thời gian lãng phí. Sau khi thiết kế xong các thử nghiệm của mình, chúng ta có thể tự động thực thi chúng mỗi khi thay đổi hoặc thêm vào cơ sở mã của mình.



- Các 4 loại testing chính được sử dụng ở các thời điểm khác nhau trong chu kỳ phát triển:
- **Unit tests**: kiểm tra trên các thành phần riêng lẻ mà mỗi thành phần có một trách nhiệm duy nhất (ví dụ: chức năng lọc danh sách).
- **Integration tests:** kiểm tra chức năng kết hợp của các thành phần riêng lẻ (ví dụ: xử lý dữ liệu).
- **System tests**: kiểm tra thiết kế của hệ thống về kết quả đầu ra dự kiến dựa trên đầu vào (ví dụ: đào tạo, suy luận, v.v.).
- **Acceptance tests:** kiểm tra để xác minh rằng các yêu cầu đã được đáp ứng, thường được gọi là Kiểm tra chấp nhận của người dùng (UAT).
- **Regression tests:** kiểm tra dựa trên các lỗi mà chúng tôi đã thấy trước đây để đảm bảo các thay đổi mới không tái hiện các lỗi đó



- **Công cụ sử dụng để test:** Trong Python, có nhiều công cụ, chẳng hạn như unittest, pytest, v.v. cho phép chúng ta dễ dàng thực hiện các thử nghiệm của mình. Những công cụ này đi kèm với chức năng tích hợp mạnh mẽ như tham số hóa, bộ lọc, v.v. để kiểm tra nhiều điều kiện trên quy mô lớn.

CHƯƠNG 2 – GIẢI QUYẾT BÀI TOÁN HỌC MÁY

Mô tả về bài toán

- **Mục tiêu:** Dự đoán một người có bị rối loạn giấc ngủ hay không?

  (Không có, Mất ngủ, Ngưng thở khi ngủ).)

- **Các thuộc tính trong data:**

Bảng 2.1 Các thuộc tính trong dataset

Số thứ tự	Thuộc tính	Mô tả	Giá trị
1	Person ID	Mã định danh cho mỗi cá nhân	
2	Gender	Giới tính của người đó	Male/Female
3	Age	Tuổi của người (tính bằng năm)	
4	Occupation:	Nghề nghiệp của một người.	
5	Sleep Duration	Số giờ người đó ngủ mỗi ngày. (tính bằng giờ)	Từ 1 đến 10
6	Quality of Sleep	Đánh giá chủ quan về chất lượng giấc ngủ.	
7	Physical Activity Level	Số phút một người tham gia hoạt động thể chất hằng ngày. (phút/ngày)	

8	Stress Level	Đánh giá chủ quan về mức độ căng thẳng mà một người phải trải qua	Từ 1 đến 10
9	BMI Category	Loại BMI của một người	(Thiếu cân, Bình thường, Thừa cân)
10	Blood Pressure	Số đo huyết áp của một người, được biểu thị bằng huyết áp tâm thu so với huyết áp tâm trương.	
11	Heart Rate	Nhịp tim lúc nghỉ ngơi của một người tính bằng nhịp mỗi phút.	
12	Daily Steps	Số bước mà người đó thực hiện mỗi ngày	
13	Sleep Disorder	Sự hiện diện hay vắng mặt của chứng rối loạn giấc ngủ ở người	(Không có, Mất ngủ, Ngưng thở khi ngủ)

TÀI LIỆU THAM KHẢO

1) Bai 8: Tất tần tật về feature selection - Lựa Chọn đặc trưng trong machine learning - Thiết bị giáo dục STEM. (2022, 1),. https://ohstem.vn/tat-tan-tat-ve-feature-selection-trong-machine-learning