**FINAL PROJECT MACHINE LEARNING**

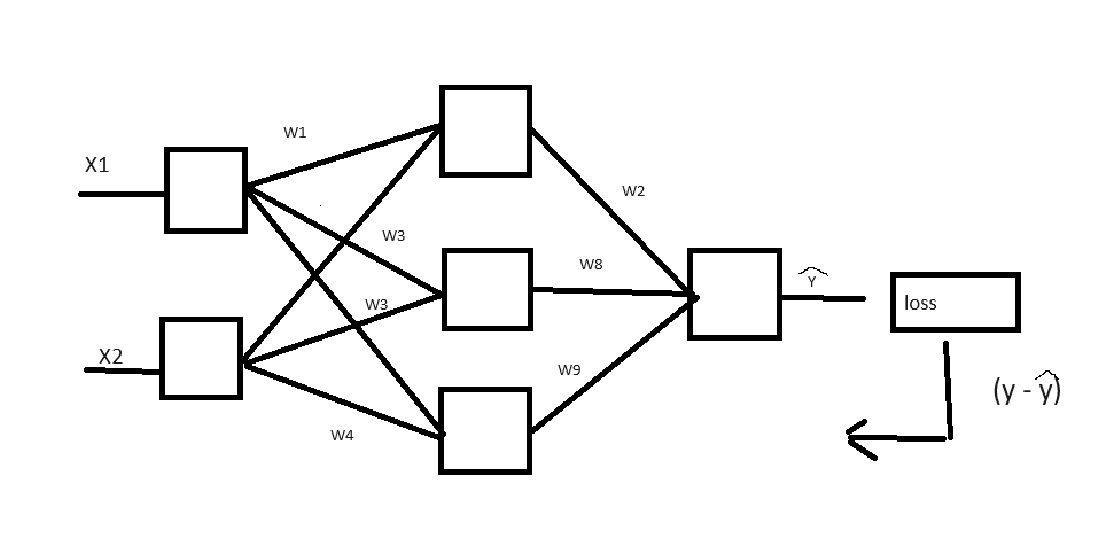
**521H0438-NGUYỄN TẤN BẢO**

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

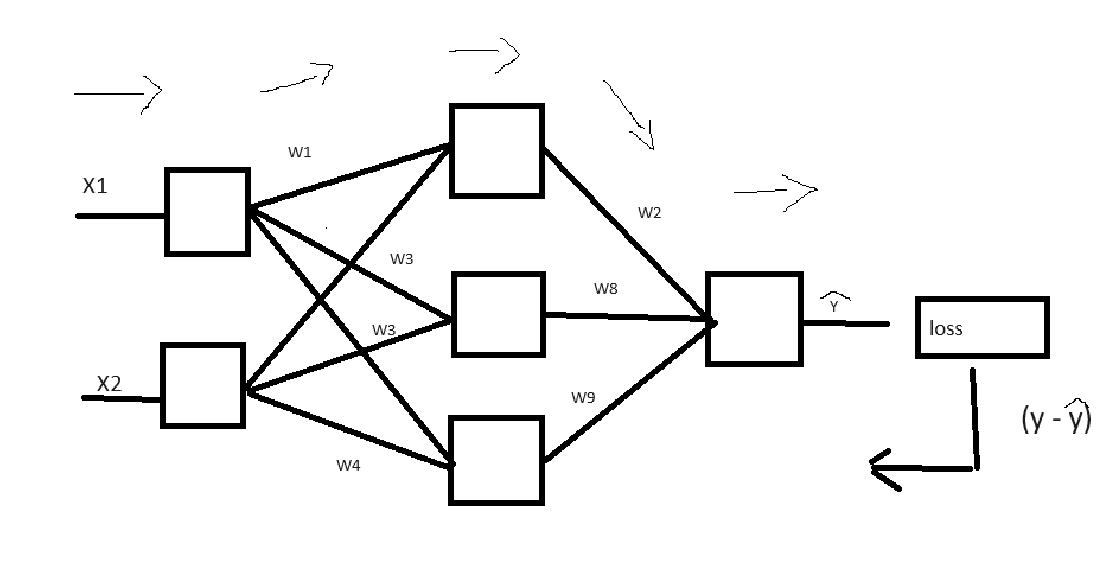
## Giới thiệu thuật toán

Optimizers là một thuật toán tối ưu, về cơ bản nó là cơ sở để xây dựng mô hình neural network với mục đích học được các feature của dữ liệu đầu vào.

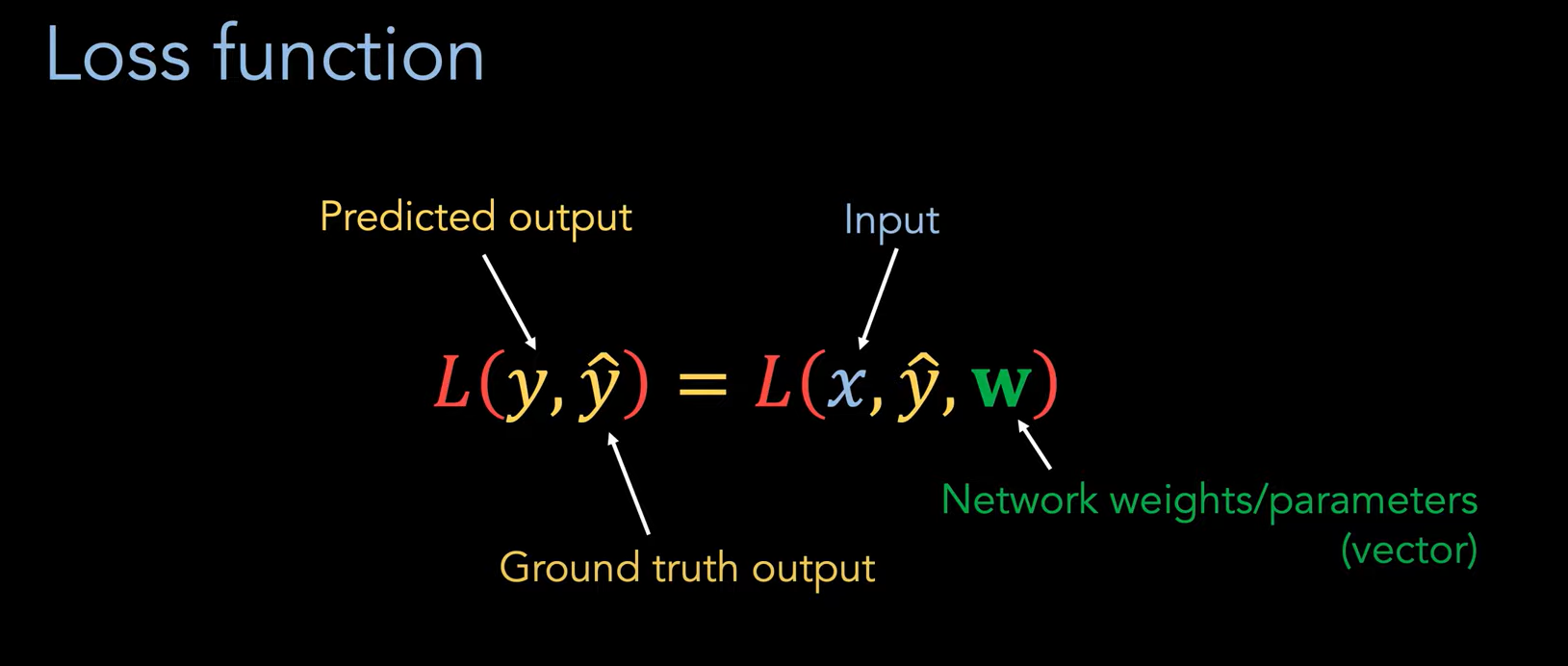
Optimizers cải thiện weight và bias theo từng bước hoặc là learning rate để giảm tổn thất và nâng cao độ chính xác cho mô hình , và đó là lý do thuật toán optimizers ra đời.



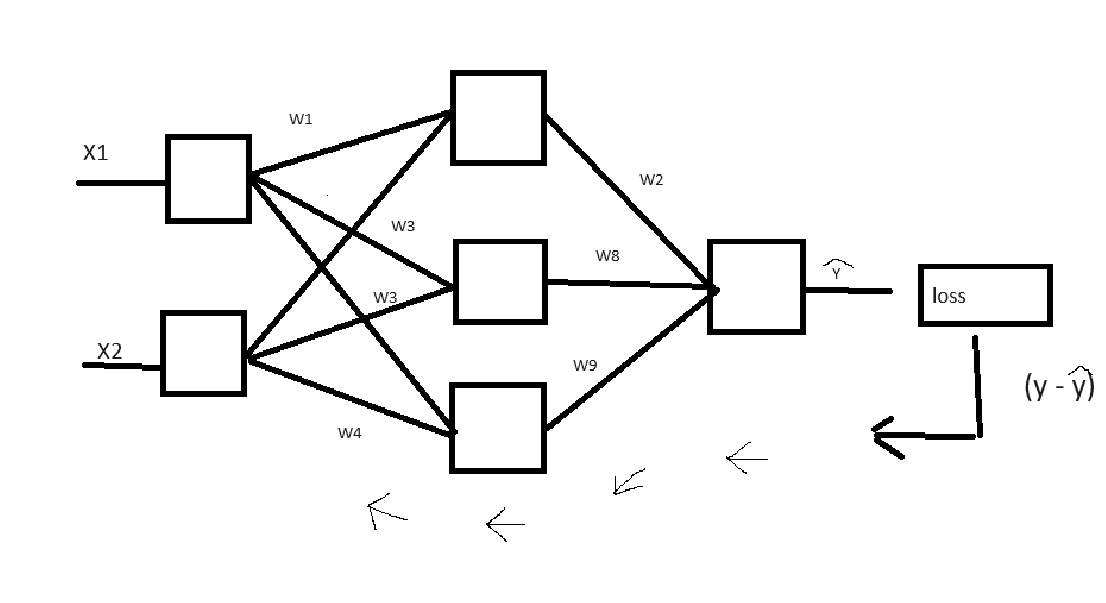
Ví dụ chúng ta có mạng lưới noron trên với 2 đầu vào input là X1 và X2 kết nối với các lớp ẩn và đầu ra của các lớp ẩn cũng được kết nối với lớp cuối cùng của mạng lưới noron. Theo cách chung thì đầu vào sẽ đi theo hướng này



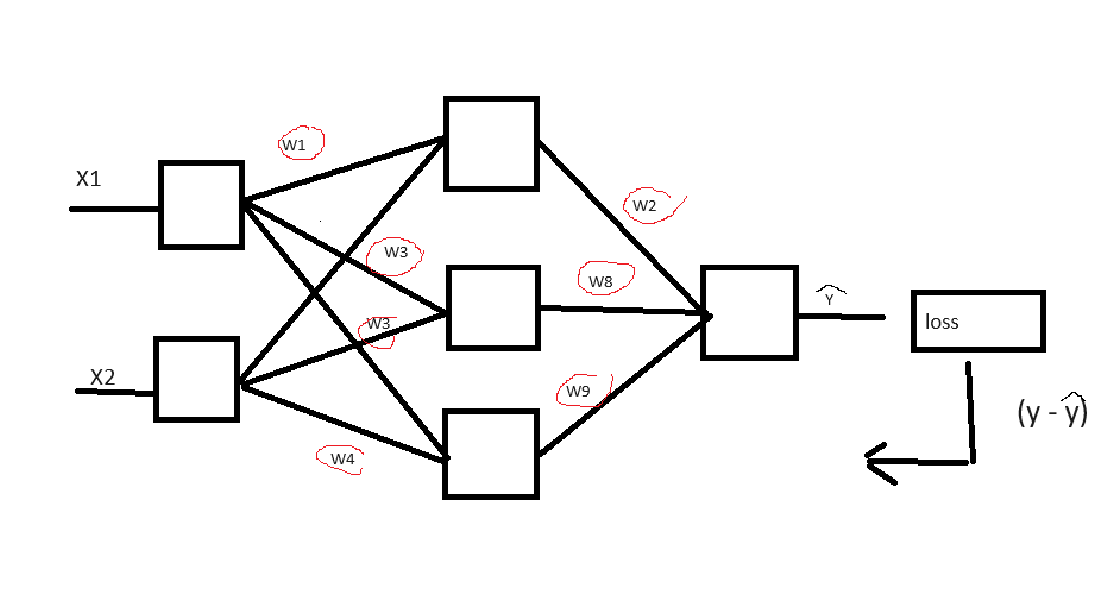
Và y^ sẽ được tính từ cuối cùng chúng ta sẽ tìm được hàm loss tính bằng (y - y^).



Nếu hàm loss này có giá trị cao thì chúng ta sẽ lan truyền ngược. Lan truyền ngược( backpropogate) giống như đi từ lớp đầu ra sang lớp đầu vào

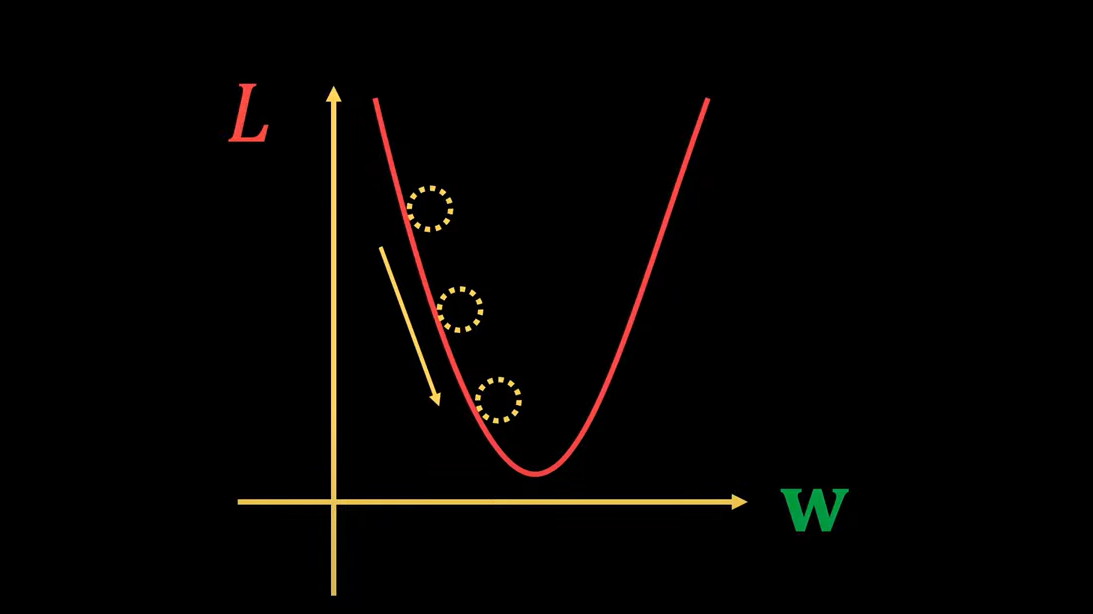


Lan truyền ngược( backpropogate) giống như đi từ lớp đầu ra sang lớp đầu vào. Tại đây chúng ta sẽ gặp được rất nhiều giá trị như w9,w8, w2 ,w3,w1,…



Vì vậy tất cả trọng số sẽ được cập nhật, vậy nên việc cập nhật các trọng số rất quan trọng, chúng ta không thể chỉ định hãy là gán các trọng số nên việc các trình tối ưu dễ liệu(optimizer) rất quan trọng trong quá trình này.Từ đó optimizer điều chỉnh trọng số, tốc độ học tập nhầm tăng cao độ chính xác của mô hình.

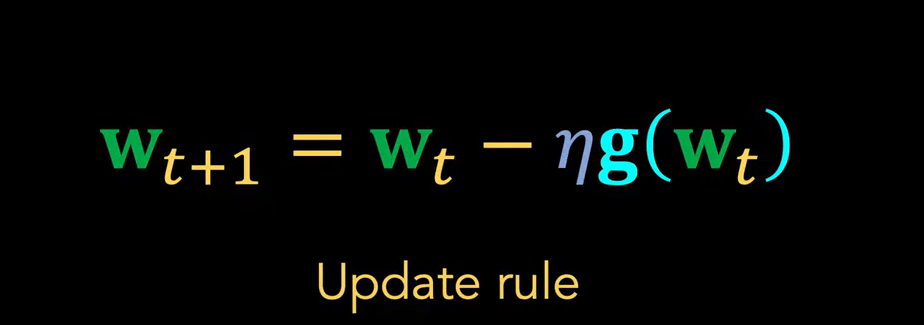
Việc này được thực hiện bởi việc giảm độ dốc trong không gian tham số trong đó mục tiêu là tìm mức tối thiểu toàn cục của hàm Loss hoặc ít nhất là ở làm hám loss ở mức tối thiểu cục bộ.



## Các thuật toán Optimizer

### 1.2.1 Gradient Descent

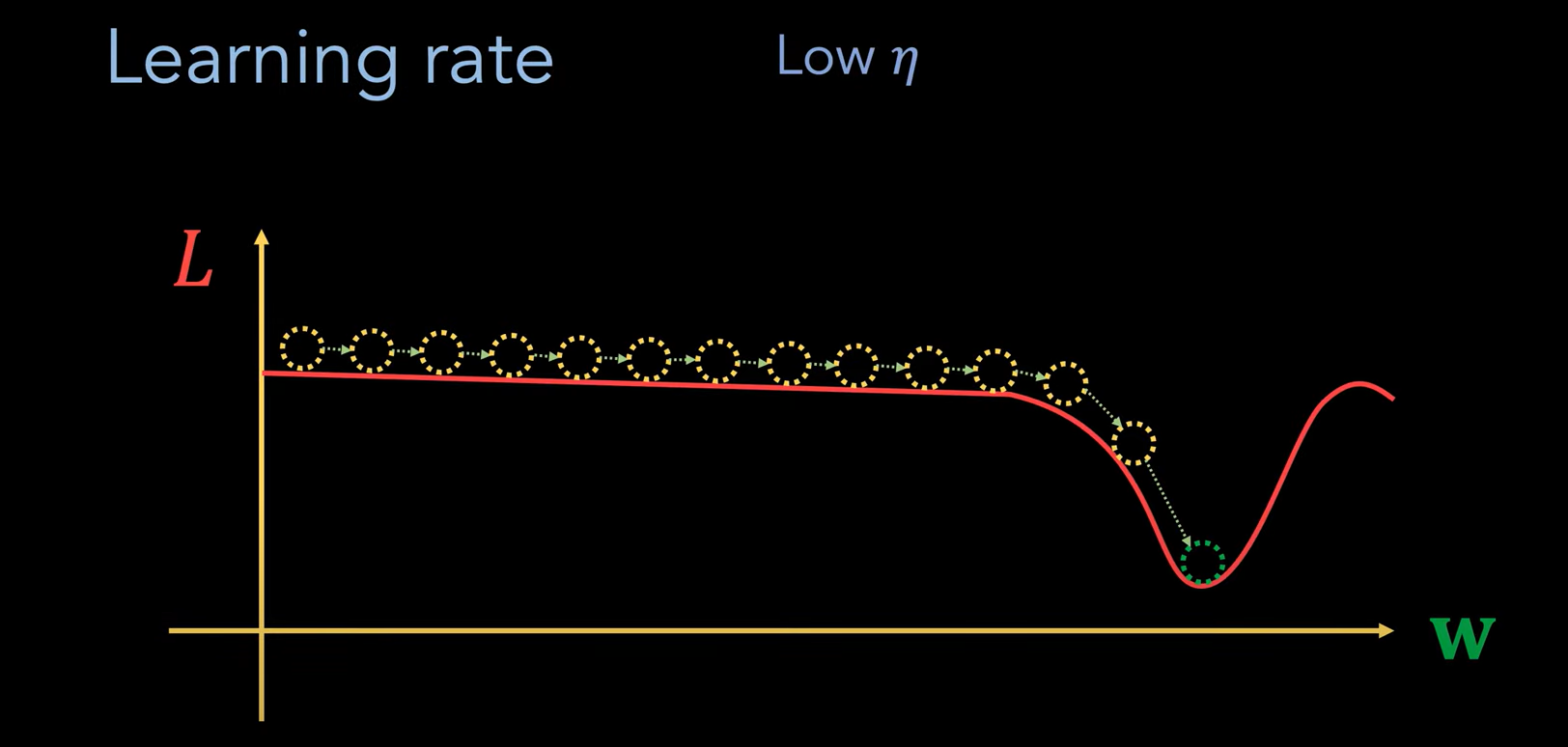
### Mục tiêu chính của phương pháp tối ưu hóa Gradient Descent là điều chỉnh các tham số (hoặc trọng số) của mô hình để giảm thiểu giá trị của hàm mất mát. Mô hình thường dự đoán một giá trị, và hàm mất mát đo lường sự chênh lệch giữa giá trị dự đoán và giá trị thực tế. Do đó, khi giảm thiểu giá trị của hàm mất mát, chúng ta đang cố gắng làm cho dự đoán của mô hình gần với thực tế nhất có thể.Gradient Descents (GD)



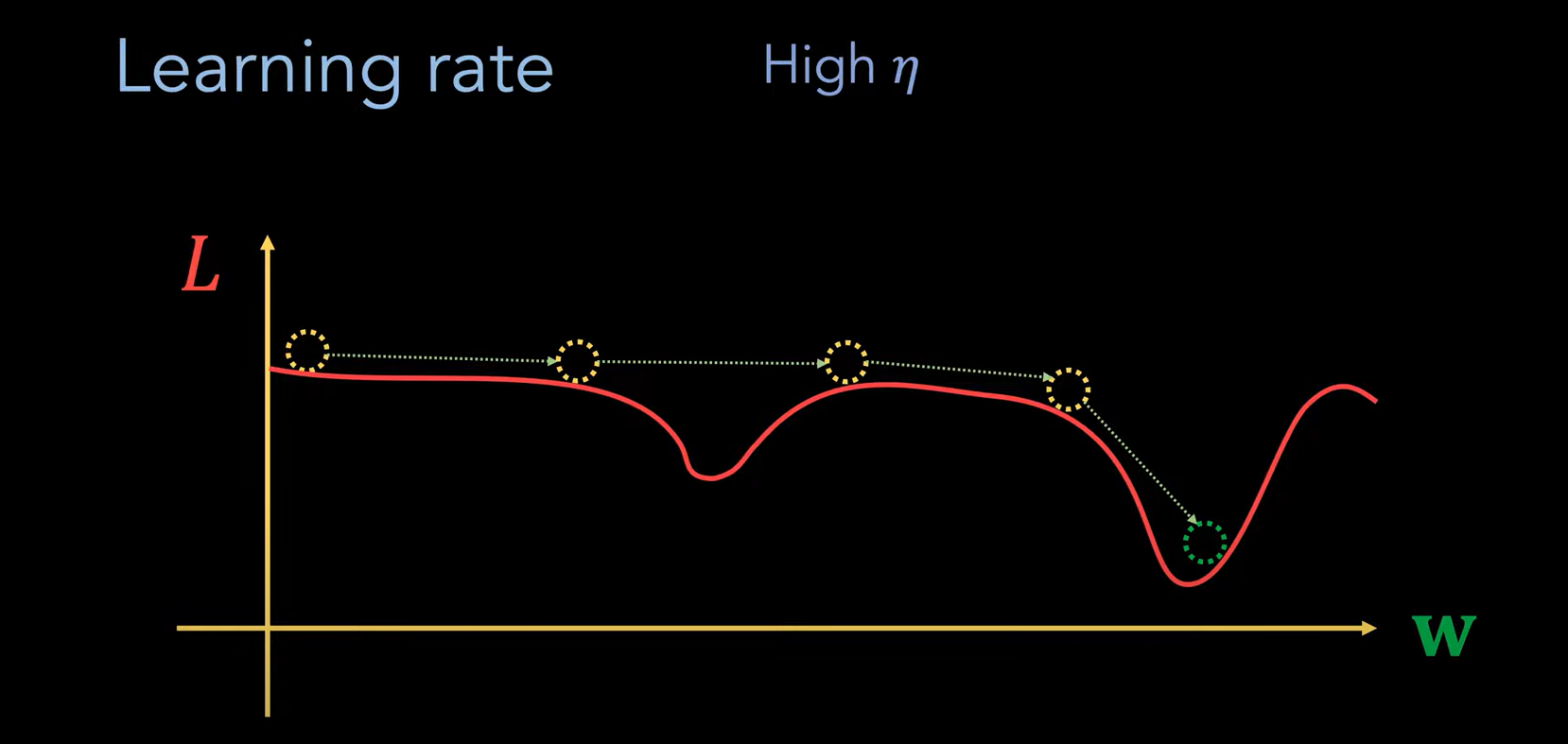
Tại một thời điểm nào đó, T được sửa đổi theo độ dốc được chia bởi tỷ lệ bởi một số siêu tham số ( n) đại diện cho tốc độc học tập(Learning rate)

Learning rate là một tham số cực kỳ quan trọng, trong quá trình giảm độ dốc ngẫu nhiên

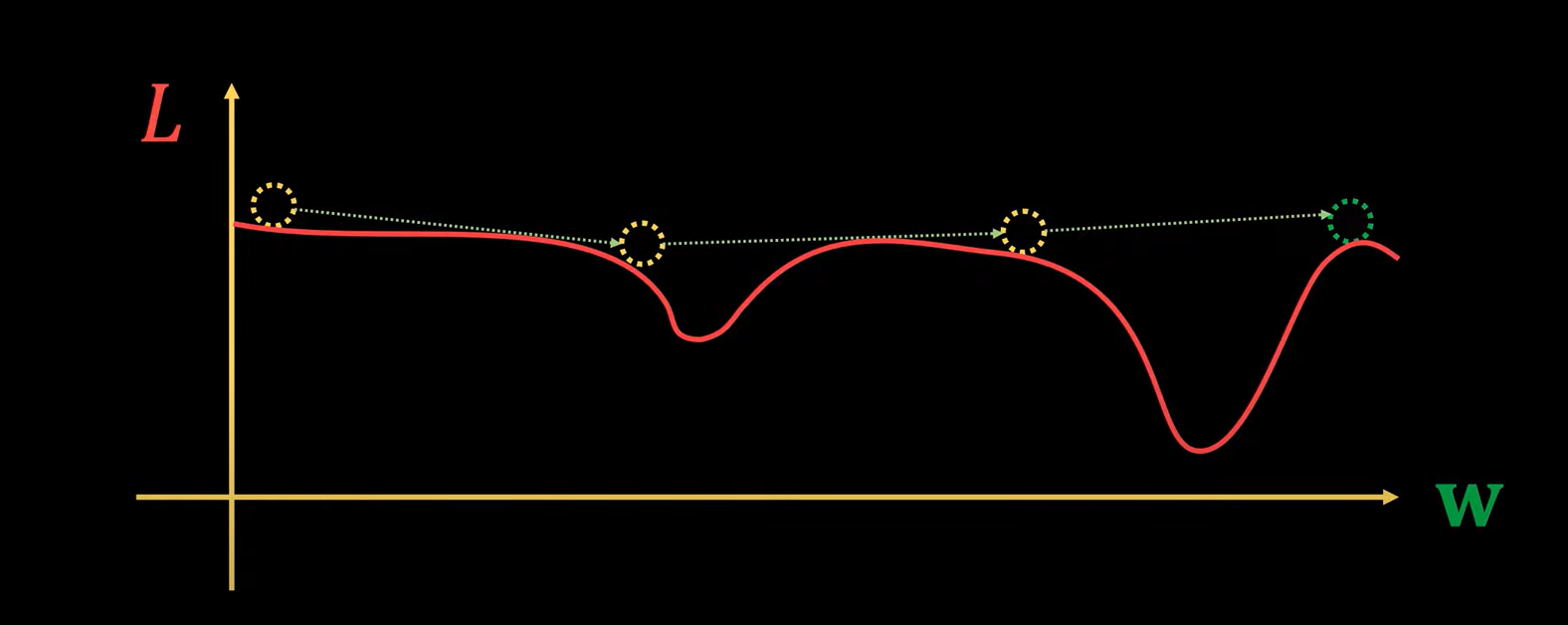
Example: Với một learning rate thấp, có thể mất rất nhiều bước thời gian để thay đổi trọng số w để có thể đưa hàm Loss về mức tối thiểu



Với một learning rate cao hơn thì tốc độ điều chỉnh trọng số để đưa ra độ chính hình chính xác càng nhanh hơn



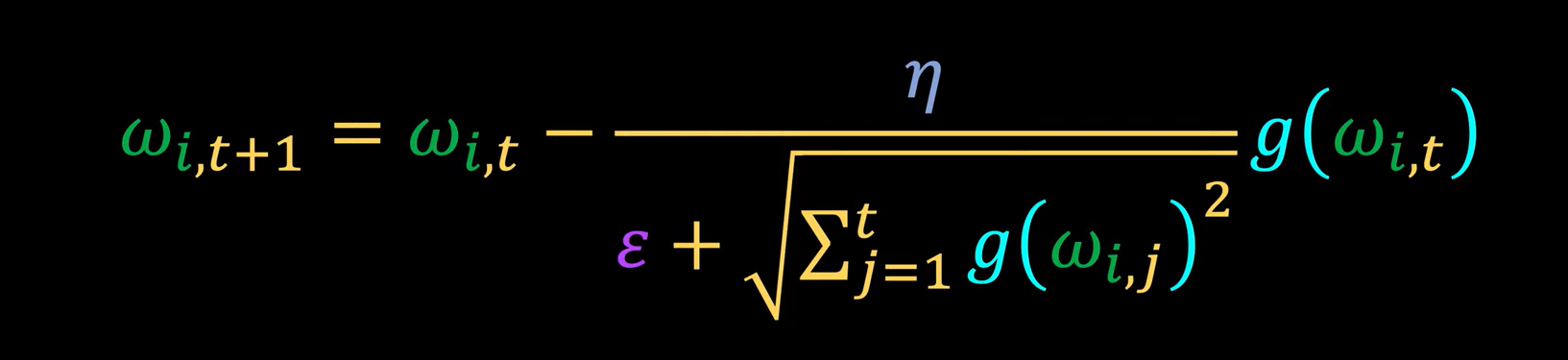
Nhưng cũng có vài trường hợp với một learning rate cao thì trọng số cuối cùng có thể vượt qua mức tối thiểu mục tiêu cho Hàm Loss dẫn đến ảnh hưởng độ chính xác của mô hình.



Vì vậy việc lựa chọn learning rate trọng việc điều chỉnh trọng số trong gradient descent rất quan trọng. Trong thực tế mất rất nhiều lần thử nghiệm để có thể đưa ra được giá trị tối ưu

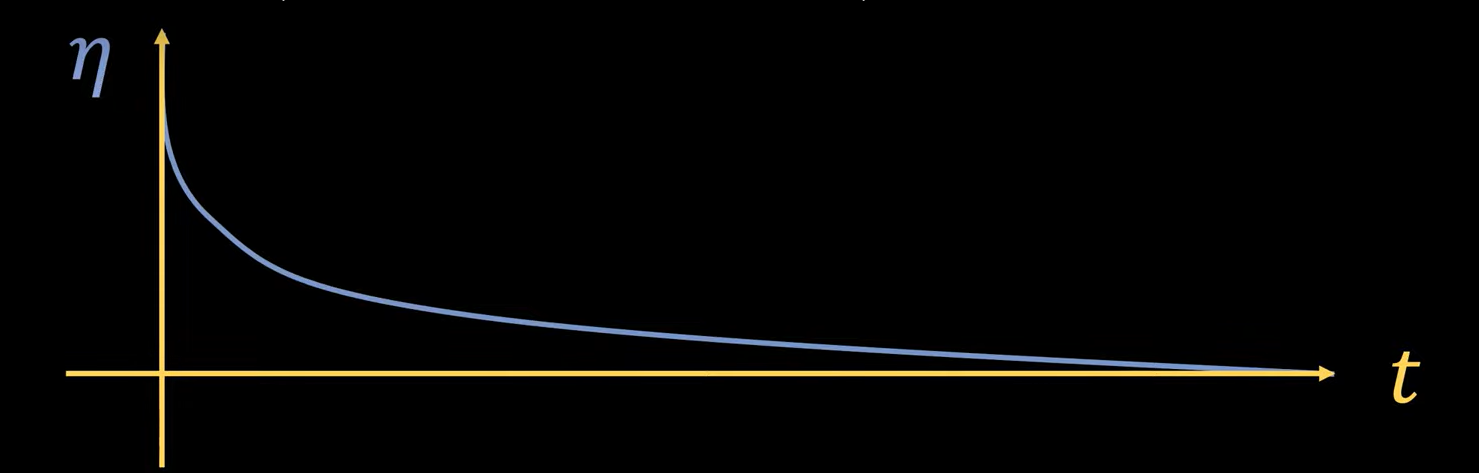
### 1.2.2. AdaGrad

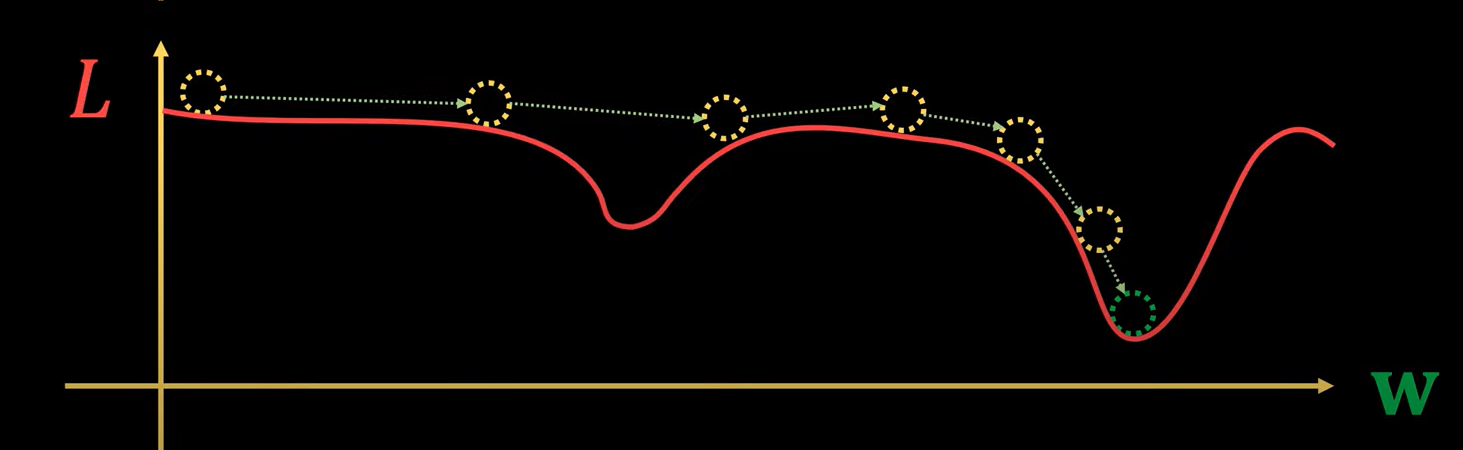
Đây là phương pháp đối với mỗi trọng số riêng lẻ được biểu thị bởi Omega, độ dốc, gradient scale được chia theo căng bậc hai của tổng các gradient đã gặp trước



Logic ở đây tương tự như logic của Learning rate trong đó (n : chỉ số của learning rate) được giảm theo một loại lịch trình nào đó trong hầu hết thời gian, nó giảm theo cấp số nhân khi tiến trình đào tạo và trọng số đạt đến gần với mục tiêu tối thiểu( hàm Loss minimum)

Example:





Như chúng ta có thể thấy khi trọng số w gần tới mục tiêu tối thiểu ( Loss minimum) thì Learning rate càng giảm.

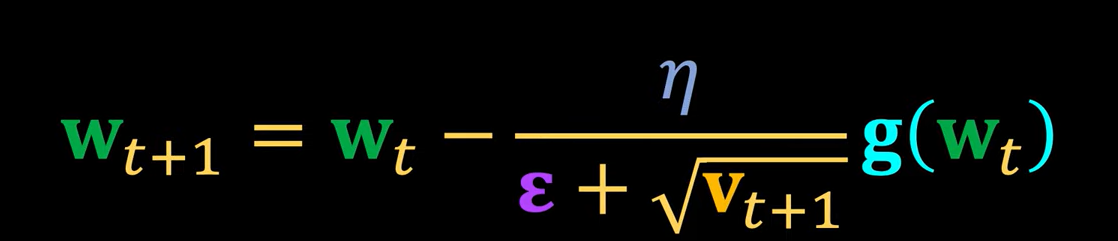
Nhưng nhược điểm của những thuật toán giảm Learning Rate này là khi n sẽ giảm khi tiến tới các mục tiêu tốt nhưng mà mục tiêu thực sự đang còn rất xa điều đó có thể ảnh hưởng đến thời gian training cho mô hình.

Tuy nhiên Ada đã tạo ra một sự thay đổi nhỏ đối với ý tưởng về tốc độ học tập giảm dần bằng cách chia tỷ lệ từng tham số khác nhau tùy theo số lượng rằng tham số cụ thể đó đã thay đổi trong quá trình đào tạo được đánh giá bằng tổng bình phương của các gradient trước đó.

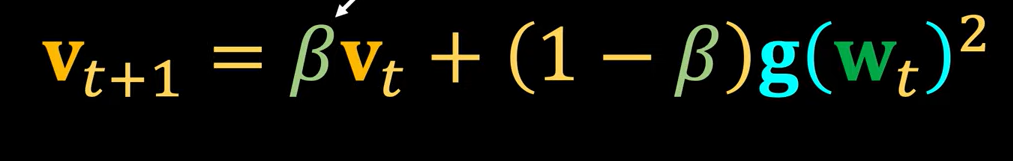
Ý tưởng là nếu một tham số đã thay đổi đang kể thì nó phải đạt được nhiều tiến bộ hướng tới mục tiêu, tuy nhiên nếu nó không thay đổi nhiều thì nên tiếp tục được cập nhật với sự tăng trưởng nhiều hơn và cần lưu ý ở đây là chúng ta cố bộ ổn định số ở đây để đảm bảo rằng chúng ta không chia cho các giá trị cực gần bằng 0:

### 1.2.3. RMSprop

Đây là một thuật toán sinh ra để khắc phụ vấn đề learning rate thấp khi chưa đạt tới mục tiêu tối thiểu bằng cách cho phép vừa tăng vừa giảm tốc độ học. Như với adagrad, thuật toán giữ một bộ nhớ của các gradient trước đó:



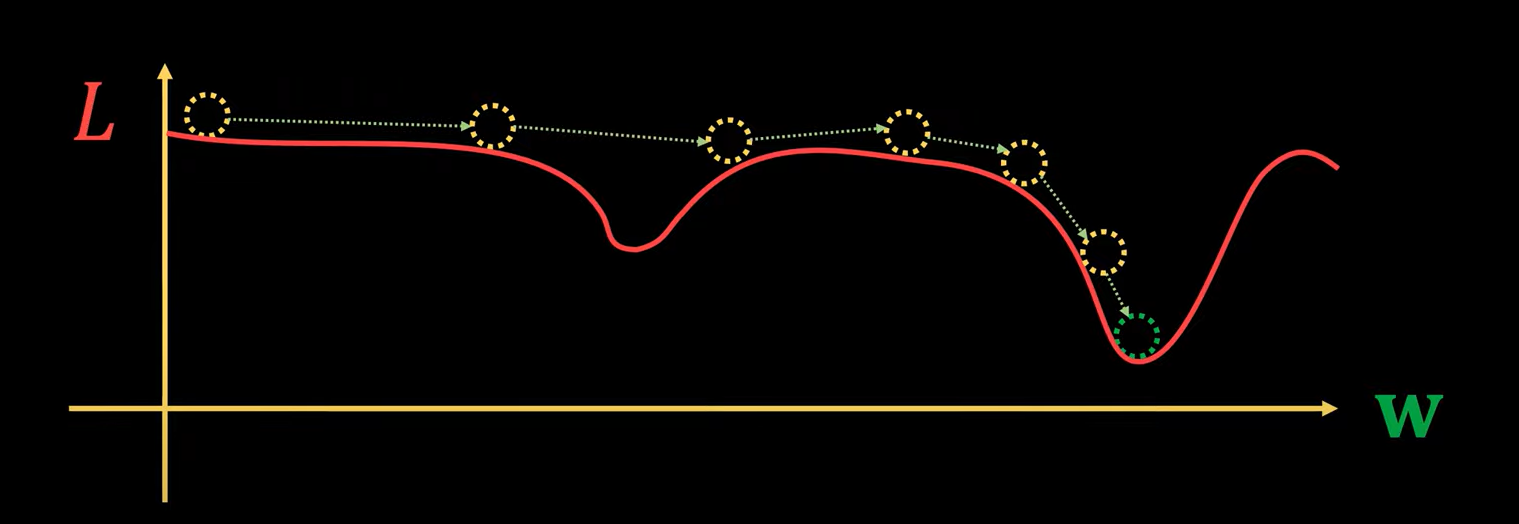
và lần này vận tốc học được cập nhật từ bước này sang bước tiếp theo tham số giảm beta để kiểm soát mức độ của tốc độ học trước đó được ghi nhớ.





Do đó khi gặp phải một gradient lớn tốc độ được sửa đổi chẳng hạn như learning rate giảm xuống và khi gặp một gradient nhỏ thì tốc độ học được tăng tỉ lệ.

Example:



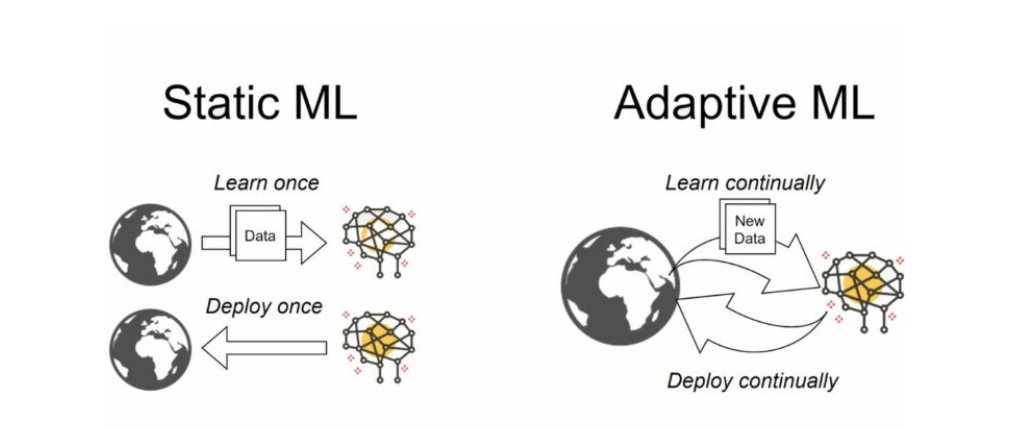
Chúng ta có thể thấy khi độ dốc nhỏ thì tốc độ học được tăng lên và khi gặp những dốc lớn thì tốc độ học giảm xuống và khi gặp những độ dốc nhỏ thì tốc độc học lại tăng lên điều này giúp cho mô hình chúng ta có thể học một cách chính xác mà không phải chịu tốc độc giảm dần vĩnh viễn như adagrad

### **Continual Learning and Test Production**

## 2.1 Continual Learning

**1. Định Nghĩa**

Học tập liên tục (CL) tập trung vào việc phát triển các mô hình để học các nhiệm vụ mới trong khi vẫn giữ lại thông tin từ các nhiệm vụ trước đó. CL là một lĩnh vực nghiên cứu quan trọng vì nó giải quyết tình huống thực tế trong đó dữ liệu và nhiệm vụ liên tục thay đổi và mô hình phải thích ứng với những thay đổi này mà không quên kiến thức trước đó.



Trong học máy truyền thống, một mô hình được đào tạo trên một tập dữ liệu cố định và dự kiến sẽ thực hiện một nhiệm vụ duy nhất. Tuy nhiên, cách tiếp cận này trở nên có vấn đề khi dữ liệu và nhiệm vụ thay đổi và linh hoạt, vì mô hình phải có khả năng thích ứng và học hỏi từ dữ liệu mới theo thời gian. Đây là lúc CL phát huy tác dụng, cho phép mô hình liên tục học hỏi và cải thiện mà không quên kiến thức trước đó.

## ****Why do we need continual learning?****

Câu trả lời khá đơn giản - dữ liệu đang thay đổi. Dữ liệu có thể thay đổi do xu hướng hoặc do các hành động khác nhau của người dùng. Ví dụ, cuốn sách bán chạy nhất của Amazon từ năm 2000 là cuốn Harry Potter. Hôm nay, bạn có thể ngạc nhiên khi biết cuốn sách bán chạy nhất lại thuộc một thể loại hoàn toàn khác: Lửa và cuồng nộ: Bên trong Nhà Trắng của Trump.



Vì vậy, Amazon sẽ phải đào tạo lại mô hình và giới thiệu sách mới cho khách hàng dựa trên dữ liệu và xu hướng mới. Một ví dụ cập nhật hơn một chút là giá bitcoin trước khi giảm mạnh. Năm 2017, bitcoin trị giá 19 nghìn đô la. Khoảng một tháng rưỡi sau, nó giảm xuống còn 6 nghìn đô la.

Các nhà nghiên cứu không chỉ thay đổi dữ liệu mà còn bày tỏ rằng “việc học tập suốt đời vẫn là một thách thức lâu dài đối với các mô hình mạng lưới thần kinh và học máy vì việc liên tục thu thập thông tin có sẵn dần dần từ các phân phối dữ liệu không cố định thường dẫn đến tình trạng lãng quên hoặc can thiệp nghiêm trọng”. Trường hợp học tập liên tục vẫn mạnh mẽ. Đối với các nhà khoa học dữ liệu, việc học tập liên tục cuối cùng sẽ tối ưu hóa độ chính xác của mô hình, cải thiện hiệu suất mô hình và tiết kiệm thời gian đào tạo lại bằng cách làm cho mô hình tự động thích ứng.

**2.Vấn Đề Cơ Bản:**

Các mô hình học máy truyền thống thường gặp vấn đề quên đi kiến thức khi học từ dữ liệu mới, đặc biệt là khi chuyển từ một miền dữ liệu sang miền dữ liệu khác. Continual Learning giúp giải quyết vấn đề này bằng cách duy trì và cập nhật kiến thức đã học mà không làm ảnh hưởng đến khả năng học từ dữ liệu mới.

Tóm lại, CL là một lĩnh vực quan trọng trong học máy nhằm giải quyết thách thức của các mô hình đào tạo có thể liên tục học hỏi và thích ứng với các nhiệm vụ và dữ liệu mới mà không quên kiến thức trước đó. Việc sử dụng các kỹ thuật như mạng **chính quy hóa và tăng cường bộ nhớ, cũng như thiết kế kiến trúc mô hình,** đóng một vai trò quan trọng trong khả năng thích ứng của các mô hình CL.

**3. Thách Thức:**

Lượng Dữ Liệu Giới Hạn: Trong các tình huống thực tế, có thể không có đủ dữ liệu để huấn luyện lại toàn bộ mô hình trên tất cả các dữ liệu cũ và mới.

Quản lý Interference (Can thiệp): Các tri thức mới có thể làm ảnh hưởng đến tri thức đã học trước đó, gây ra hiện tượng can thiệp và ảnh hưởng đến hiệu suất.

**4. Phương Pháp:**

**Regularization Techniques** (Kỹ Thuật Chính Quy): Sử dụng các phương pháp chính quy để giảm thiểu việc quên kiến thức cũ, ví dụ như **Elastic Weight Consolidation** (EWC).

Việc lưu giữ kiến thức đã học trước đó trong học tập tuần tự được thực hiện theo phương pháp **Elastic Weight Consolidation** bằng cách thêm một bộ quy tắc đến chức năng mất mát ngăn chặn các trọng lượng quan trọng nhất bị lệch

Khác xa với các giá trị hợp nhất trong quá trình học các nhiệm vụ tiếp theo:



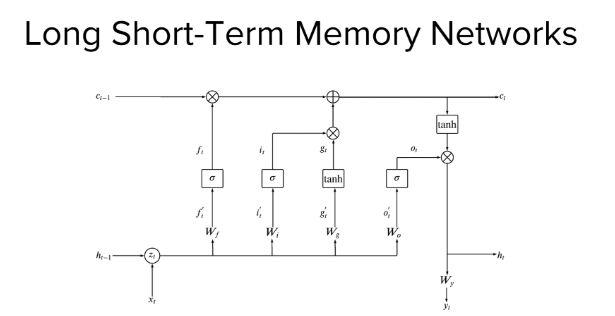
Memory-Augmented Networks (Mạng Tăng Cường Bộ Nhớ): Sử dụng các kiến trúc mạng có khả năng lưu giữ thông tin quan trọng từ quá khứ, như Neural Turing Machines hoặc Memory Networks.

trong đó LA là chức năng mất của đào tạo A,  Trọng lượng (thông số) của

mạng lưới thần kinh sau khi đào tạo cho các nhiệm vụ trước đó, Ωi – tầm quan trọng của trọng lượng thứ I của mạng nơ-ron sau khi đào tạo đến các nhiệm vụ trước đó. Từ công thức chúng ta có thể thấy rằng sự đóng góp của thành phần chính quy hóa vào ý chí chống gradient và do đó, khi sử dụng các phương pháp gradient để đào tạo, các khả năng chống thay đổi từng trọng lượng sẽ tỷ lệ thuận với tầm quan trọng của nó và siêu tham số λ.

**Memory-Augmented Networks** (Mạng Tăng Cường Bộ Nhớ): Sử dụng các kiến trúc mạng có khả năng lưu giữ thông tin quan trọng từ quá khứ, như Neural Turing Machines hoặc Memory Networks.

Một ví dụ cho thuật toán Memmory Networks mà chúng ta đã học là



5. **Ứng Dụng:**

a. Robotics (Robot Tự Động):

Trong lĩnh vực robot tự động, **Continual Learning là quan trọng để robot có thể thích ứng với môi trường làm việc đa dạng và biến động.**

Robot có thể học liên tục từ các nhiệm vụ như điều hướng trong môi trường không gian hạn chế, nhận biết đối tượng, hoặc thậm chí là thực hiện các tác vụ phức tạp như lắp ráp.

b. Ngôn Ngữ Tự Nhiên:

**Continual Learning có thể được áp dụng để mở rộng khả năng hiểu và tạo ra ngôn ngữ tự nhiên của mô hình.**

Trong hệ thống trò chuyện tự động, mô hình có thể liên tục học từ các cuộc trò chuyện trước đó để cải thiện khả năng hiểu ngữ cảnh và tạo ra câu trả lời tự nhiên hơn.

c. Hệ Thống Quản Lý Thông Tin:

**Trong các hệ thống quản lý thông tin, Continual Learning giúp duy trì và cập nhật cơ sở dữ liệu với thông tin mới.**

Hệ thống có thể học từ dữ liệu mới như tin tức, bài báo, hoặc dữ liệu từ mạng xã hội để cập nhật thông tin và cung cấp dữ liệu chính xác và hiện đại.

d. Nhận Diện Hình Ảnh và Video:

**Trong lĩnh vực nhận diện hình ảnh và video, Continual Learning giúp mô hình nhận biết đối tượng và biến động trong thời gian thực.**

Mô hình có thể liên tục học từ các loại đối tượng mới xuất hiện trong môi trường và thích ứng với các biến động, nhưng vẫn giữ được khả năng nhận diện các đối tượng đã học từ trước.

## 2.2 Test Production

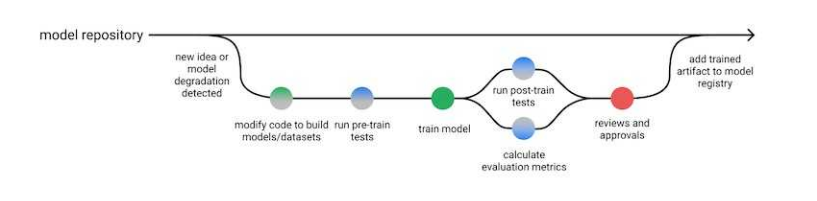
Test Production là một khía cạnh quan trọng trong quá trình xây dựng giải pháp học máy. Nó đề cập đến việc tạo ra một tập dữ liệu kiểm thử đa dạng và đại diện để đánh giá hiệu suất của mô hình học máy trước khi triển khai vào môi trường thực tế.

**Ý nghĩa**

Test Production đảm bảo rằng mô hình học máy không chỉ hoạt động tốt trên tập dữ liệu huấn luyện mà còn đối mặt tốt với dữ liệu mới, không gian dữ liệu mà mô hình có thể gặp trong tương lai. Điều này giúp đánh giá độ chính xác, độ tin cậy và khả năng tổng quát hóa của mô hình.

Mặt khác, kiểm thử ML liên quan đến việc kiểm tra hành vi của mô hình. Các bài kiểm tra trước khi đào tạo — có thể chạy mà không cần các tham số được đào tạo — kiểm tra xem logic viết của chúng tôi có chính xác không. Ví dụ, xác suất phân loại có nằm trong khoảng từ 0 đến 1 không? Các bài kiểm tra sau đào tạo kiểm tra xem logic đã học có được mong đợi hay không. Ví dụ, trên bộ dữ liệu Titanic, chúng ta nên mong đợi phụ nữa có xác suất sống sót cao hơn (so với đàn ông).

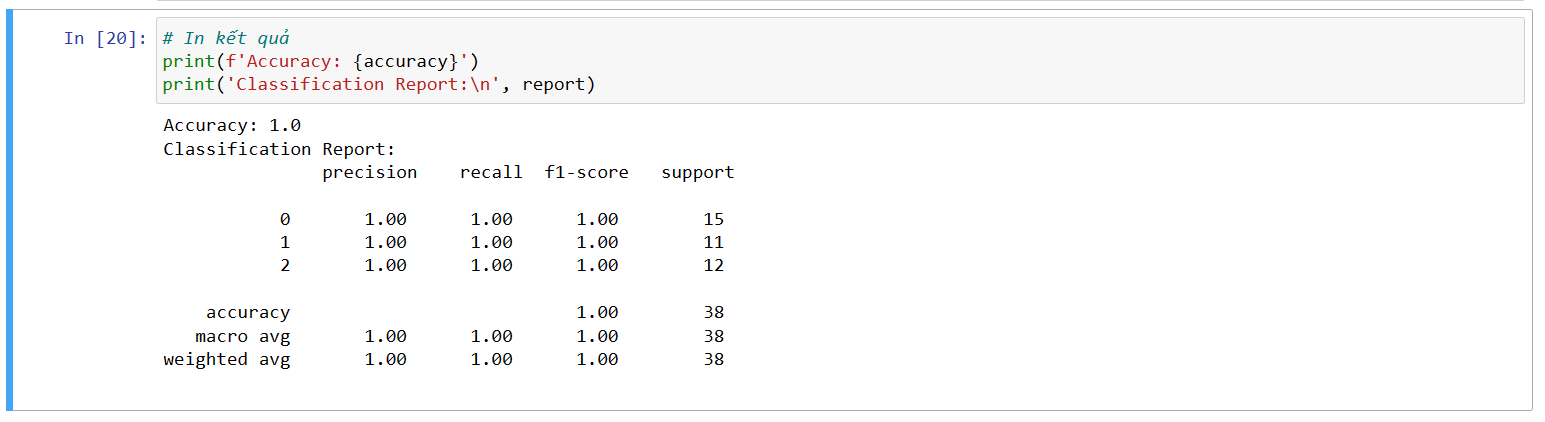
**Work flow diagram**



**Demo**

Dưới đây là một ví dụ với mô hình Logistic Regression và tập dữ liệu Iris từ thư viện scikit-learn





# Trích dẫn

Trần Trung Trực (2023, December 23). *Optimizer- Hiểu Sâu về Các Thuật Toán Tối ưu ( GD,SGD,adam,..)*. Viblo. https://viblo.asia/p/optimizer-hieu-sau-ve-cac-thuat-toan-toi-uu-gdsgdadam-Qbq5QQ9E5D8

Wang, L., Zhang, X., Su, H., & Zhu, J. (2023b, June 11). *A comprehensive survey of continual learning: Theory, method and application*. arXiv.org. https://arxiv.org/abs/2302.00487

*Machine learning in production - testing*. Machine Learning in Production - Testing. (n.d.). https://applyingml.com/resources/testing-ml/

Tercan, H., Deibert, P., & Meisen, T. (2021, June 5). *Continual learning of neural networks for quality prediction in production using memory aware synapses and weight transfer - journal of intelligent manufacturing*. SpringerLink. https://link.springer.com/article/10.1007/s10845-021-01793-0

Journal of La a comprehensive survey of continual learning: Theory ... (n.d.). https://arxiv.org/pdf/2302.00487.pdf