Machine Learning

Project Final

52100872 – La Quốc Bảo

Mục lục

1. Các phương pháp Optimizer

Trong quá trình huấn luyện mô hình học máy, việc chọn các phương pháp tối ưu hóa là một phần quan trọng để cải thiện tốc độ và hiệu suất learning của một mô hình. Optimized được sử dụng trong mô hình theo hướng giảm thiểu hàm mất mát (loss function). Loss function là một hàm đo lường mức độ chêch lệch giữa kết quả dự đoán của mô hình và giá trị thực tế. Về cơ bản thì Optimized là cở sở để xây dựng mô hình NN với mục đích learning được các features của dataset, từ đó tìm cặp weights và bias phù hợp để tối ưu hóa model.

Có nhiều phương pháp Optimizer khác nhau có thể được sử dụng trong huấn luyện mô hình học máy. Mỗi phương pháp sẽ có các ưu và nhược điểm riêng phù hợp với các mô hình và các tập dữ liệu khác nhau:

1.1. Gradient Descent (GD)

Đây là phương pháp tối ưu hóa đơn giản nhất. Phương pháp này cập nhật tham số của mô hình theo hướng gradient của loss function tại thời điểm hiện tại với mục tiêu là di chuyển đến điểm cực tiểu của loss function. Là một thuật toán tối ưu lặp (iterative optimization algorithm) được sử dụng để tìm tập các biến nội tại (internal parameters).

$$loss = -(y \times \log(\hat{y}) + (1 - y) \times \log(1 - \hat{y}))$$

Có 3 loại Gradient Descent:

- Batch Gradient Descent: Cập nhật tham số sau khi tính toán đạo hàm trên toàn bộ tập huấn luyện.

- Stochastic Gradient Descent: Cập nhật tham số sau khi tính toán đạo hàm trên một điểm dữ liệu.
- Mini-batch Gradient Descent: Cập nhật tham số sau khi tính toán đạo hàm trên một batch dữ liệu.

Quá trình thực thi / hoạt động của phương pháp Gradient Descent:

- B1. Khởi tạo các biến nội tại
- B2. Đánh giá model dựa theo biến nội tại ở B1 và hàm mất mát
- B3. Cập nhật lại các biến nội tại theo hướng tối ưu hàm mất mát
- B4. Lặp lại B2 và B3 cho tới khi thỏa điều kiện dừng (đạt ngưỡng vòng lặp tối đa, đạt ngưỡng hội tụ, hội tụ đến điểm cực tiểu)

Tốc độ và hiệu suất của phương pháp GD sẽ phụ thuộc vào điểm khởi tạo ban đầu và learning rate. Đặc biệt là learning rate vì nếu lựa chọn learning rate quá nhỏ thì tốc độ hội tụ sẽ rất chậm, tệ nhất là không bao giờ tới đích nếu bài toán quá phức tạp. Với learning rate quá cao, mặc dù tộc độ hội tụ rất nhanh (có thể chỉ qua vài lần lặp) nhưng thuật toán có thể sẽ không hội tụ được vì bước nhảy quá lớn dẫn đến nó cứ quẩn quanh đích mà không thể chạm đích.

Ưu điểm	Nhược điểm
- Cơ bản, dễ hiểu - Giải quyết được vấn đề tối ưu model neural network bằng cách	- Còn hạn chế phụ thuộc vào điểm khởi tạo ban đầu và learning rate - Chậm hội tụ

cập nhật tham số sau mỗi vòng lặp	- Dễ bị mắc kẹt trong các điểm tối ưu cục bộ

1.2. Batch Gradient Descent

Batch Gradient Descent là phương pháp cập nhật tham số sau khi tính toán đạo hàm trên toàn bộ tập huấn luyện. Đây là phương pháp cập nhật tham số chậm nhất vì chúng ta phải tính toán đạo hàm trên toàn bộ tập huấn luyện trước khi cập nhật tham số.

Cách làm này có một vài hạn chế đối với cơ sở dữ liệu có vô cùng nhiều điểm.Việc phải tính toán lại đạo hàm với tất cả các điểm này sau mỗi vòng lặp trở nên cồng kềnh và không hiệu quả.

- Ứng dụng của Batch Gradient Descent:

+ Linear Regression: Batch Gradient Descent được sử dụng để tìm các tham số của mô hình hồi quy tuyến tính.

+ Logistic Regression: Batch Gradient Descent được sử dụng để tìm các tham số của mô hình hồi quy logistic.

+ Support Vector Machines: Batch Gradient Descent được sử dụng để tìm các tham số của mô hình Support Vector Machines.

+ Neural Networks: Batch Gradient Descent được sử dụng để tìm các tham số của các mạng nơ-ron nhân tạo

Công thức cập nhật tham số:

$$\theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} \left(\phi \left(z^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

1.3. Stochastic Gradient Descent (SGD)

Là một biến thể của Gradient Descent. Thay vì sử dụng toàn bộ tập huấn luyện thì phương pháp này sẽ chỉ cập nhật tham số của mô hình dựa trên một số dữ liệu mẫu ngẫu nhiên được chọn trong tập dữ liệu. Do tính chất ngẫu nhiên nên hàm mất mát sẽ lúc tăng lúc giảm nhưng sẽ giảm dần theo thời gian.

Mỗi lần duyệt một lượt qua tất cả các điểm trên toàn bộ dữ liệu sẽ được gọi là một epoch, sau mỗi lần epoch thì chúng ta cần phải suffle thứ tự của các dữ liệu để đảm bảo tính ngẫu nhiên. Phù hợp với các bài toán có tập dữ liệu lớn.

- Ứng dụng của Stochastic Gradient Descent:

+ Huấn luyện mạng nơ ron: SGD là một thuật toán phổ biến để huấn luyện mạng nơ ron, bao gồm cả mạng nơ ron tích chập (CNN) và mạng nơ ron tái phát (RNN).

+ Huấn luyện các mô hình hồi quy: SGD cũng có thể được sử dụng để huấn luyện các mô hình hồi quy, chẳng hạn như mô hình đường hồi quy và mô hình dự đoán.

+ Huấn luyện các mô hình phân loại: SGD cũng có thể được sử dụng để huấn luyện các mô hình phân loại, chẳng hạn như mô hình phân loại logistic và mô hình hỗ trợ vector machine (SVM).

Quá trình hoạt động :

- B1. Chia nhỏ dữ liệu : Chia thành các mini-batch nhỏ hơn
- B2. Lặp qua từng mini-batch
- B3. Cập nhật tham số
- B4. Lặp lại quá trình cho đến khi thỏa điều kiện dừng

Ưu điểm	Nhược điểm
- Hiểu quả cao cho các bài toán có dữ liệu lớn - Thích hợp cho bài toán online learning - Tránh được việc kẹt ở điểm cực tiểu cục bộ	- Không ổn dịnh do tính chất ngẫu nhiên có thể tạo ra dao động lớn - Cần đặt learning rate thích hợp - Khó điều chỉnh hệ số momentum và các siêu tham số khác

1.4. Mini-batch Gradient Descent

Mini-batch Gradient Descent là phương pháp cập nhật tham số sau khi tính toán đạo hàm trên một batch dữ liệu. Đây là phương pháp cập nhật tham số trung bình giữa Batch Gradient Descent và Stochastic Gradient Descent.

Khác với SGD, mini-batch sử dụng một số lượng n lớn hơn 1 (nhưng vẫn nhỏ hơn tổng số dữ liệu N rất nhiều). Giống với SGD, Mini-batch Gradient Descent bắt đầu mỗi epoch bằng việc xáo trộn ngẫu nhiên dữ liệu rồi chia toàn bộ dữ liệu thành các mini-batch, mỗi mini-batch có n điểm dữ liệu (trừ mini-batch cuối có thể có ít hơn nếu N không chia hết cho n). Mỗi lần cập nhật, thuật toán này lấy ra một mini-batch để tính toán đạo hàm rồi cập nhật. Công thức có thể viết dưới dạng:

$$\theta_j := \theta_j - \eta \frac{1}{n} \sum_{i=1}^{n} \left(\phi \left(z^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

1.5. Momentum

Đây là một phương pháp để cải thiện tốc độ hội tụ của Stochastic Gradient Descent bằng cách giảm thiểu sự dao động lớn của tham số và giúp thuật toán di chuyển nhanh hơn trong không gian tham số. Làm cho nghiệm vượt qua giá trị local minimum không mong muốn và tiếp tục tiến đến giá trị global minimum. Momentum là một vector hướng tính theo hướng mà tham số của mô hình đã được cập nhật trước đó.

- Ứng dụng của Momentum:

+ Huấn luyện mạng nơ ron: Momentum là một kỹ thuật phổ biến để huấn luyện mạng nơ ron, bao gồm cả mạng nơ ron tích chập (CNN) và mạng nơ ron tái phát (RNN).

+ Huấn luyện các mô hình hồi quy: Momentum cũng có thể được sử dụng để huấn luyện các mô hình hồi quy, chẳng hạn như mô hình đường hồi quy và mô hình dự đoán.

+ Huấn luyện các mô hình phân loại: Momentum cũng có thể được sử dụng để huấn luyện các mô hình phân loại, chẳng hạn như mô hình phân loại logistic và mô hình hỗ trợ vector machine (SVM).

Quá trình hoạt động:

- Sử dụng một vector momentum để lưu trữ hướng của các cập nhật tham số đã được thực hiện trước đó. Khi cập nhật tham số

tại thời điểm hiện tại, momentum sẽ được sử dụng để tăng tốc độ, giúp mô hình nhanh chóng tìm ra tập tham số tối ưu

$$v_t = \gamma v_{t-1} + \eta \nabla \theta J(\theta) \theta = \theta - v_t$$

Ưu điểm	Nhược điểm
- Tối ưu hóa được vấn đề của Gradient Descent không tiến tới được điểm global minimum mà chỉ dừng lại ở local minimum	- Tuy giải quyết đc vấn đề đến global minimum nhưng nó vẫn mất khá nhiều thời gian giao động qua lại trước khi đừng lại

1.6. AdaGrad (Adaptive Gradient Algorithm)

Đây là một phương pháp điều chỉnh learning rate của Stochastic Gradient Descent dựa trên gradient của hàm mất mát (loss function) để cải thiện hiệu suất thuật toán. AdaGrad xem learning rate là 1 tham số, do đó learning rate sẽ biến thiên sau mỗi thời điểm t.

$$G_{t,i} = G_{t-1,i} + \left(\nabla J\left(W_{t,i}\right)\right)^2$$

$$W_{t+1,i} = W_{t,i} - \frac{\propto}{\sqrt{G_{t,i} + \varepsilon}} \nabla J\left(W_{t,i}\right)$$

Quá trình hoạt động:

- Adagrad điều chỉnh learning rate của mỗi tham số dựa trên lịch sử của gradient của tham số đó. Cơ bản là cập nhật learning rate của mỗi tham số sao cho tham số có gradient lớn hơn thì learning rate sẽ giảm, và ngược lại.

Ưu điểm	Nhược điểm
- Tính hiệu quả cho các tham số ít xuất hiện trong tập dữ liệu huấn luyện - Tự điều chỉnh learning rate phù hợp	- Làm giảm learning rate quá nhanh dẫn đến quá trình học chậm lại - Khó khăn trong việc điều chỉnh hệ số epsilon - Có thể làm dẫn đến tốc độ học quá nhanh

1.7. RMSprop (Root Mean Square Propagation)

Phương pháp này cũng tương tự như AdaGrad dùng để điều chỉnh learning rate của Stochastic Gradient Descent dựa trên loss function nhưng phương pháp này sử dụng bộ nhớ ngắn hạn hơn so với AdaGrad. Giải quyết được vấn đề tỷ lệ giảm dần của Adagrad bằng việc giảm độ dao động của learning rate để mang lại hiệu suất được cải thiện hơn. RMSProp sử dụng đường trung bình động của gradient bình phương để chia learning rate cho từng tham số.

$$E[g^2]_t = 0.9[g^2]_{t-1} + 0.1g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Quá trình hoạt động:

- Tương tự như Adagrad, RMSprop cũng duy trì một ma trận đường chéo (diagonal matrix) để điều chỉnh learning rate của mỗi

tham số dựa trên lịch sử của gradient của tham số đó. Tuy nhiên, khác với Adagrad, RMSprop sử dụng trung bình có trọng số của các bình phương gradient.

Ưu điểm	Nhược điểm
- Tương đối dễ điều chỉnh - Giảm độ dao động của learning rate - Giảm thiểu được nguy cơ mắc kẹt trong các điểm tối ưu cục bộ	- Không hoạt động tốt cho tất cả các bài toán - Khó khăn trong việc điều chỉnh hệ số beta - Có thể làm dẫn đến tốc độ học quá nhanh

1.8. Adam (Adaptive Moment Estimation)

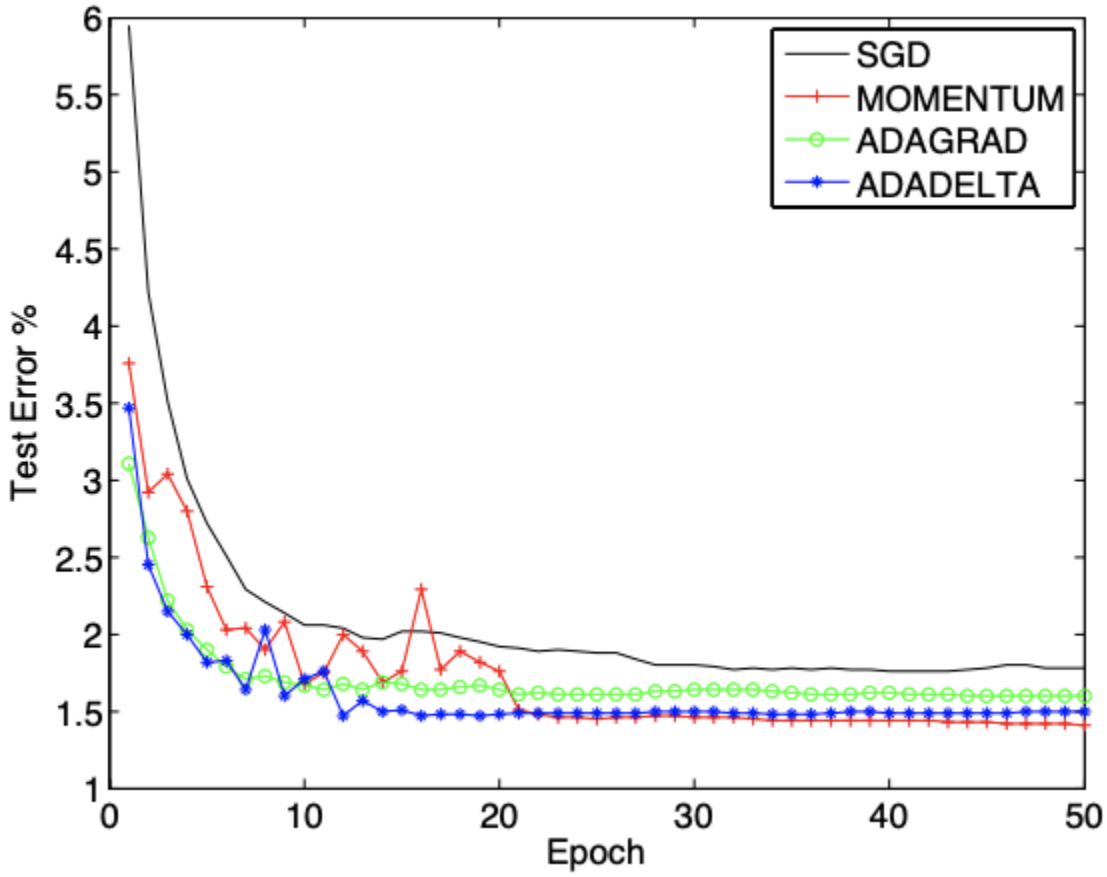
Phương pháp này là sự kết hợp của phương pháp Momentum và RMSprop để cải thiện tốc độ hội tụ của Stochastic Gradient Descent. Trong phương pháp Adam thường được ví như bài toán một quả cầu nặng có ma sát khi lao xuống dốc nó sẽ dễ dàng vượt qua điểm local minimum tới global minimum và khoảng thời gian dao động xung quanh đích sẽ rút ngắn hơn phương pháp Momentum vì nó ma sát.

Quá trình hoạt động:

- Adam kết hợp cả Momentum và RMSprop để điều chỉnh learning rate của mỗi tham số dựa trên lịch sử gradient và tốc độ di chuyển.

Ưu điểm	Nhược điểm

- Giảm thiểu được nguy cơ mắc kẹt trong các điểm tối ưu cục bộ - Hiệu suất tốt - Ít siêu tham số cần điều chỉnh	- Khó điều chỉnh hệ số learning rate theo từng bước để tối ưu - Không phù hợp với mọi loại bài toán



1.9. Nadam (Nesterov-accelerated Adaptive Moment Estimation)

Phương pháp này là một biến thể của phương pháp Momentum. Phương pháp này cập nhật tham số của mô hình theo hướng gradient của loss function tại thời điểm hiện tại cộng với hệ số momentum.

Ưu điểm	Nhược điểm
- Giảm thiểu được nguy cơ mắc kẹt trong các điểm tối ưu cục bộ - Hiệu suất tốt - Kết hợp tốt giữa Adam và NAG	- Có thể làm dẫn đến tốc độ học quá nhanh - Đòi hỏi tính toán phức tạp hơn và lưu trữ số lượng lớn giá trị trung bình

2. Continual Learning

2.1. Khái niệm

Continual Learning (CL hoặc còn được gọi là Lifelong Learning) là một kỹ thuật trong học máy cho phép mô hình có thể learning liên tục từ dữ liệu mới mà không mất đi kiến thức đã learning từ dữ liệu cũ hoặc phải khởi tạo lại quá trình huấn luyện. Nó là một phần quan trọng của học máy mang tri thức và có khả năng tiếp tục học từ dữ liệu mới. Nhưng nó cần phải được thử nghiệm để đảm bảo rằng nó an toàn và tốt hơn so với mô hình hiện tại đang được sử dụng.

Continual Learning thường bị hiểu sai là một loại thuật toán trong Machine Learning bởi vì do một số thuật toán trong học máy như Naïve cập nhật tuần tự hay thuật toán phân loại KNN có thể cập nhật dần dần theo thời gian khi có dữ liệu mới. Tuy nhiên, Continual Learning có thể được áp dụng cho bất kỳ thuật toán nào trong học máy được giám sát.

Continula Learning thường bị hiểu sai là bắt đầu huấn luyện lại từ đầu mỗi khi có dữ liệu mới truyền vào, điều này sẽ dẫn đến một hậu quả là

bởi vì huấn luyện lại từ đầu khi có dữ liệu mới sẽ khiến cho mô hình reset lại từ đầu và quên đi các dữ liệu cũ trước đó đã được sử dụng. Nếu như một mô hình quên đi những gì đã học từ các tập dữ liệu cũ thì sẽ dẫn đến giảm hiệu suất của mô hình đó. Vì thế mà Continual Learning thường được chia nhỏ ra để cập nhật mô hình theo từng đợt nhỏ lẻ.

Continual Learning đòi hỏi cả kỹ năng của người sử dụng và công nghệ cơ sở hạ tầng phát triển. Các nhà khoa học dữ liệu phải thiết kết và triển khai giải pháp học tập liên tục cho mô hình, còn công nghệ cơ sở hạ tầng cần thiết để thu thập và lưu trữ dữ liệu mới sau đó là để cập nhật mô hình đang được sử dụng.

CL được sử dụng để giải quyết các bài toán trong đó dữ liệu liên tục được cập nhật hoặc thay đổi.

- Incremental Learning: Phương pháp này cập nhật tham số của mô hình dựa trên từng mẫu dữ liệu mới.
- Ensemble Learning: Phương pháp này kết hợp nhiều mô hình CL khác nhau để cải thiện hiệu quả.
- Lifelong Learning: Phương pháp này cập nhật tham số của mô hình dựa trên cả tập dữ liệu cũ và mới.
- Stateless Retraining: Phương pháp này sẽ huấn luyện lại mô hình từ đầu trên toàn bộ dữ liệu hiện có (cũ và mới). Không ghi nhớ trạng thái hoặc kiến thức đã học trước đó.
- Stateful Training: Phương pháp này cập nhật mô hình dần dần với dữ liệu mới, dựa trên trạng thái của mô hình hiện tại. Sử dụng kỹ thuật regularizers, replaying, meta learning.

2.2. Thách thức

Continual learning đã được sử dụng nhiều cùng với các thành công lớn nhưng Continual learning cũng có nhiều thách thức mà nó gặp phải

Thuật toán:

- Gặp phải khi sử dụng các thuật toán đòi hỏi quyền truy cập vào tập dữ liêu đầy đủ như các mô hình dựa trên ma trận, dựa trên giảm kích thước,… Những loại mô hình này không được huấn luyện tăng dần bằng dữ liệu mới như mạng nơ-ron.
- Gặp phải khi cần cập nhật nhanh vì không thể chờ thuật toán quét qua hết tập dữ liệu

Truy cập dữ liệu mới:

- Tốc độ lưu trữ dữ liệu vào kho dữ liệu: Kho dữ liệu là nơi chứa tất cả dữ liệu xuất phát từ các nguồn khác nhau bằng các cách thức và tốc độ lưu trữ khác nhau. Bài toán sẽ phát sinh thách thức khi lấy dự liệu trực tiếp từ quá trình vận chuyển theo thời gian thực để huấn luyện trước khi đưa vào kho dữ liệu và các thách thức đó là không thể bơm tất cả dữ liệu của mình qua các sự kiện. Điều này đặc biệt phổ biến với các dữ liệu nằm trong hệ thống của nhà cùng cấp. Cần phải tìm ra các thực hiện quá trình xử lý hiệu quả trên một luồng dữ liệu
- Tốc độ ghi nhãn: Gán nhãn dữ liệu mới yêu cầu nguồn tài nguyên lớn đặc biệt là ki dữ liệu được truyền đến từ nhiều nguồn. Bởi vì dữ liệu có thể đến từ nhiều nguồn khác nhau với tính đa dạng cao

sẽ càng làm cho bài toán thêm độ phức tạp trong việc gán nhãn chính xác.

Dữ liệu không cân bằng:

- Khi dữ liệu mới có được phân phối không đồng đều hoặc không cân bằng, mô hình có thể dễ dàng bị "thiên vị" và làm giảm độ chính xác trên lớp dữ liệu thiểu số.
- Chia dữ liệu thường yêu cầu quyền truy cập vào số liệu , nếu sử dụng phương pháp đào tạo số liệu thống kê phải xem xét cả dữ liệu cũ đã được sử dụng cộng với dữ liệu mới để làm mới mô hình nên phải ước tính các thống kê này tăng dần khi quan sát dữ liệu mới.

Khả năng có thể "quên" dữ liệu: Mô hình khi học từ dữ liệu mới có thể sẽ có khả năng quên đi kiến thức đã học từ dữ liệu cũ, xảy ra hiện tượng quên mất các thông tin quan trọng có trong các dữ liệu cũ dẫn đến suy giảm độ chính xác trên dữ liệu cũ.

Bảo mật:

- Quản lý các dữ liệu có chứa thông tin nhạy cảm hoặc cá nhân, việc lưu trữ và xử lý những dữ liệu này có thẻ gây ra rủi ro về bảo mật nếu không được quản lý cẩn thận
- Rò rỉ thông tin từ mô hình ra bên ngoài sẽ phát sinh thông qua việc phản hồi từ mô hình đã được cập nhật với dữ liệu mới.

- Mô hình học máy liên tục có thể trở thành mục tiêu cho các cuộc kỹ thuật tấn công nhằm đầu độc các mô hình như chèn nhiễu vào dữ liệu, tấn công đường dẫn, tấn công trực tiếp lên mô hình.

2.3. Các giai đoạn chính của Continual Learning

Giai đoạn 1: Huấn luyện lại một cách thủ công, không trạng thái (manual, stateless retraining)

- Các mô hình chỉ được huấn luyện 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) nhóm của bạn có thời gian để cập nhật mô hình.

Giai đoạn 2: Huấn luyện lại trạng thái tự động theo lịch trình, không trạng thái (fixed schedule stateless retraining)

- Giai đoạn này thường xảy ra khi các mô hình chính của một miền đã được phát triển và do đó ưu tiên không còn là tạo các mô hình mới mà là duy trì và cải thiện các mô hình hiện có.
- Tần suất đào tạo lại ở giai đoạn này thường dựa trên ”trực giác”.
- Giúp giảm thiểu công sức thủ công nhưng chưa linh hoạt.
- Cần script tự động chạy huấn luyện lại, triển khai và đánh giá mô hình.
- Các bước cấp cao của tập lệnh này là:
    - Kéo dữ liệu
    - Xuống mẫu hoặc lấy mẫu dữ liệu nếu cần thiết
    - Trích xuất các tính năng

- Xử lý và chú thích nhãn để tạo dữ liệu huấn luyện
- Bắt đầu quá trình huấn luyện
- Đánh giá mô hình mới
- Triển khai sử dụng

Giai đoạn 3: Huấn luyện có trạng thái theo lịch trình cố định (fixed schedule automated stateful training)

- Cập nhật mô hình dựa trên các phiên bản trước đó, theo dõi lịch sử phiên bản cập nhật. Hiệu quả hơn, tiết kiệm tài nguyên hơn. Hầu hết các hệ thông lưu trữ mô hình hiện tại chưa hỗ trợ tính năng này.

Giai đoạn 4: Học tập liên tục

- Cập nhật mô hình tự động dựa trên các yếu tố: thời gian, hiệu suất, lượng dữ liệu, sự thay đổi dữ liệu, khối lượng dữ liệu. Linh hoạt nhất, nhưng cần thiết kế hệ thống kích hoạt và theo dõi hiệu quả phức tạp.

2.4. Các ứng dụng của Continual Learning

- Nhận diện giọng nói: cải thiện độ chính xác của nhận diện giọng nói như gọi điện thoại hoặc AI giọng nói (siri, google assistant,…)
- Đề xuất sản phẩm.

2.5. Ưu điểm của Continual Learning

- Tổng quát hóa. Việc học hỏi liên tục giúp mô hình trở nên mạnh mẽ và chính xác hơn khi xử lý dữ liệu mới.

- Lưu trữ thông tin. Bằng cách sử dụng chiến lược học hỏi liên tục, mô hình sẽ xem xét kiến thức trước đây thu được trong các lần lặp lại trước đây, cho phép mô hình tích lũy thông tin theo thời gian.
- Khả năng thích ứng. Một mô hình áp dụng phương pháp học tập liên tục sẽ thích ứng với kiến thức mới từ đó có khả năng dự đoán tốt hơn về lâu dài.

3. Test Production

Test Production là một kỹ thuật kiểm tra và triển khai liên tục các mô hình học máy. Để kiểm tra đầy đủ các mô hình trước khi phổ biến rỗng rãi, cần đưa ra một quy trình rõ ràng về các đánh giá các mô hình bao gồm thử nghiệm, tác nhân, ngưỡng áp dụng để thúc đẩy mô hình lên giai đoạn tiếp theo. Test Production được sử dụng để đảm bảo rằng các mô hình học máy luôn hoạt động tốt trong môi trường sản xuất.

Test Production thường bao gồm các bước sau:

- Thu thập dữ liệu từ môi trường sản xuất: Dữ liệu này được sử dụng để kiểm tra hiệu suất của mô hình.
- Đánh giá hiệu suất của mô hình: Evalutation metrics được sử dụng để đánh giá hiệu suất của mô hình.
- Triển khai mô hình nếu hiệu suất đạt yêu cầu: Nếu hiệu suất của mô hình đạt yêu cầu, thì mô hình sẽ được triển khai lên môi trường sản xuất.

Chiến lược Testing in Production Strategies

- **Kiểm tra thủ công**
- **Kiểm tra tự động**
- **Shadow Deployment:** Triển khia mô hình thách đấu song song với mô hình hiện đang sử dụng. Gửi mọi yêu cầu đến cả hai mô hình nhưng chỉ phục vụ suy luận của mô hình tốt nhất. Ghi lại các dự đoán của cả hai mô hình và so sánh chúng.
    - Đây là các an toàn nhất, đơn giản về mặt khái niệm, thử nghiệm sẽ thu thập đủ dữ liệu để đạt được ý nghĩa thống kê nhanh hơn tất cả các chiến lược khác vì tất cả các mô hình đều nhận được lưu lượng truy cập đầy đủ.
    - Đồng thời cách này không thể sử dụng kỹ thuật để đo lường hiệu suất của mô hình phụ thuộc vào việc quan sát cách người dùng tương tác với các dữ đoán. Tốn kém khi chạy vì nó tăng gập đôi số lượng dự đoán do sử hai mô hình cùng một lúc
- **A/B Testing:** triển khai mô hình thách thức cùng với mô hình quán quân (mô hình A) và định tuyến phần trăm lưu lượng truy cập đến người thách thức (mô hình B). So sánh hiệu suất của chúng trong môi trường thực tế.
    - Vì dữ đoán được cung cấp cho người dùng nên kỹ thuật này cho phép nắm bắt đầy đủ cách người dùng phản ứng với các mô hình khác nhau. Dễ hiểu và nhiều tài liệu hỗ trợ.
    - Kém an toàn, quyền lựa chọn cố hữu giữa việc giả định nhiều rủi ro và lấy đủ mẫu để phân tích nhanh hơn.

Các thách thức

- Môi trường sản xuất phức tạp: Khác biệt so với môi trường phát triển, môi trường sản xuất có thể gặp nhiều sự kiện bất ngờ và dữ liệu hác biệt
- Độ trôi dữ liệu: Dữ liệu theo thời gian có thể thay đổi, khiến mô hình dần kém chính xác
- Tính sẵn sàng của hệ thống: Kiểm tra không nên ảnh hưởng đến hoạt động của mô hình trong sản xuất.