TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

KHOA CÔNG NGHỆ THÔNG TIN

TRẦN NGUYỄN THU TRANG - 52100938

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

TRẦN NGUYỄN THU TRANG - 52100938

BÁO CÁO CUỐI KỲ

NHẬP MÔN HỌC MÁY

Người hướng dẫn

PGS.TS. Lê Anh Cường

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Em xin chân thành cảm ơn Thầy Lê Anh Cường và Khoa Công Nghệ Thông Tin đã tạo điều kiện cho em làm bài báo cáo này. Qua quá trình học không chỉ truyền đạt kiến thức chuyên môn mà còn trang bị cho em những kĩ năng quan trọng cần thiết để phát triển sự nghiệp tương lai.

TP. Hồ Chí Minh, ngày 23 tháng 12 năm 2023

Tác giả

Trang

Trần Nguyễn Thu Trang

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 khoa học 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 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 23 tháng 12 năm 2023

Tác giả

Trang

Trần Nguyễn Thu Trang

TÓM TẮT

Bài báo cáo này, trong chương 1 tôi đã bước đầu đánh giá được tác động của các thuật toán tối ưu khác nhau đến quá trình huấn luyện và hiệu năng sau khi huấn luyện của mô hình học máy cho bài toán. Sau đó đánh giá, so sánh các thuật toán với nhau đối với những bài toán học máy cụ thể.

Chương 2 tìm hiểu về Continual Learning và Test Product trong Machine Laerning. Tìm hiểu về định nghĩa, phân loại, ưu và nhược điểm và các vấn đề của nó trong tương lai.

MỤC LỤC

CHƯƠNG 1. TÌM HIỂU, SO SÁNH CÁC PHƯƠNG PHÁP OPTIMIZER TRONG HUẤN LUYỆN MÔ HÌNH HỌC MÁY

1.1 Tìm hiểu các phương pháp Optimizer:

1.1.1 Gradient Descent:

Gradient Descent (GD) là cốt lõi của các thuật toán tối ưu trong Machine Laerning, mà cốt lõi của GD lại chính là đạo hàm. Đạo hàm của một hàm số cho biết sự biến thiên của hàm số đó trên tập xác định. Ta có thể tìm được giá trị nhỏ nhất hoặc giá trị lớn nhất của hàm số bằng đạo hàm tùy thuộc vào đặc điểm của hàm số. Đó cũng chính là mục tiêu của tất cả các thuật toán tối ưu, chính là tìm giá trị nhỏ nhất của hàm số. Nhưng không dễ dàng để tìm được giá trị nhỏ nhất của một hàm số, đối với hàm nhiều biến thì việc tìm giá trị nhỏ nhất cũng là điều bất khả thi vì đạo hàm càng phức tạp. Thay vào đó, ta sẽ tìm điểm gần với cực tiểu của hàm số nhất và xem đó là nghiệm của bài toán.

GD dịch ra tiếng việt là giảm dần độ dốc, vậy độ dốc ở đây tương ứng với gì trong đạo hàm ?

Đạo hàm của hàm số tại một điểm mô tả sự biến thiên của hàm số tại điểm đó, chính là độ dốc của phương trình tiếp tuyến tại điểm đó vì đạo hàm của một số tại một điểm là hệ số góc của phương trình tiếp tuyến. Khi tiến dần tới điểm cực tiểu, đạo hàm của hàm số sẽ tiến dần tới 0, tức là độ dốc sẽ giảm dần khi tiến dần tới điểm cực tiểu và đạt giá trị bằng 0 tại điểm cực tiểu.

Quá trình của thuật toán:

Bước 1:Bắt đầu khởi tạo với các hệ số nhỏ, hệ số ngẫu nhiên hoặc bằng 0

$$\theta_i \leftarrow 0; \qquad \forall i$$

Bước 2:Tính đạo hàm từng phần của loss function ở vị trí hiện tại đối với từng biến

$$\frac{\partial}{\partial x_i} f\left(\theta_1, \theta_2, \ldots, \theta_n\right); \quad \forall i$$

Bước 3:Cập nhật hệ số mới theo hàm cập nhật

$$\theta_i \leftarrow \left(\theta_i - \alpha \frac{\partial}{\partial x_i} f\left(\theta_1, \theta_2, \ldots, \theta_n\right)\right); \quad \forall i$$

Trong đó, α là tốc độ học (learning rate) là một hệ số dương nhằm xác định các kích thước của bước di chuyển đến giá trị cực tiểu (hoặc cực tiểu địa phương), thường được chọn là 0.1 hoặc 0.01.

Bước 4:Lặp lại bước 1 và 2 cho đến khi đạt được điều kiện dừng.

Nhận xét :

- Thuật toán hoạt động tốt trong trường hợp không thể tìm ra giá trị nhỏ nhất bằng đại số tuyến tính

- Việc quan trọng nhất của thuật toán là tính đạo hàm của hàm số theo từng biến sau đó lặp lại bước 2

- Việc chọn hệ số learning_rate cực kì quan trọng:

+ Nếu learning_rate quá nhỏ : tốc độ hội tụ chậm ảnh hưởng đến quá trình training

+ Nếu learning_rate quá lớn : không đạt được đến giá trị cực tiểu mà chỉ quanh quẩn xung quanh giá trị cực tiểu.

1.1.2 Stochastic Gradient Descent:

Có một số biến thể khác nhau của GD tùy thuộc vào số lượng dữ liệu được sử dụng để tính Gradient của hàm mất mát. Thuật toán Batch Gradient Descent (Batch GD) tính gradient của hàm mất mát tại trọng số trên toàn bộ tậo dữ liệu. Tất

cả các điểm dữ liệu đều được sử dụng để tính gradient trước khi cập nhật bộ trọng số. Điều này làm cho Batch GD có hạn chế là khi tập dữ liệu lớn, việc tính gradient sẽ tốn nhiều thời gian và chi phí tính toán. Để khắc phục hạn chế này, thuật toán Stochastic Gradient Descent (SGD) thực hiện việc cập nhật trọng số với mỗi mẫu dữ liệu $x^{(i)}$ có nhãn tương ứng $y^{(i)}$ như sau :

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta J\left(\theta_t; x^{(i)}; y^{(i)}\right)$$

Với cách cập nhật này, SGD thường nhanh hơn Batch GD và có thể sử dụng để học trực tuyến (online learning) khi tập dữ liệu huấn luyện được cập nhật liên tục.

Với SGD, bộ trọng số θ được cập nhật thường xuyên hơn so với Batch GD và vì vậy hàm mất mát cũng dao động nhiều hơn. Sự dao động này khiến SGD có vẻ không ổn định nhưng lại có điểm tích cực là nó giúp di chuyển đến những điểm cực tiểu (địa phương) mới có tiềm năng hơn. Với tốc độ học giảm, khả năng hội tụ của SGD cũng tương đương với Batch GD.

Cách tiếp cận thứ ba là thuật toán Mini-batch Gradient Descent (Mini-batch GD). Khác với hai thuật toán trước, Mini-batch GD sử dụng *k* điểm dữ liệu để cập nhật bộ trọng số ($1 < k < N$ với N là tổng số điểm dữ liệu)

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta J\left(\theta_t; x^{(i:i+k)}; y^{(i:i+k)}\right)$$

Mini-batch GD giảm sự dao động của hàm mất mát so với SGD và chi phí tính gradient với *k* điểm dữ liệu là chấp nhận được. Mini-batch GD thường được lựa chọn khi huấn luyện mạng nơron và vì vậy trong một số trường hợp, SGD được hiểu là Mini-batch GD. Riêng bản thân Mini-batch GD không đảm bảo tìm được điểm cực tiểu của hàm mất mát mà bên cạnh đó các yếu tố như tốc độ học, thuộc tính dữ liệu và tính chất của hàm mất mát cũng ảnh hưởng đến điều này.

1.1.3 Adagrad:

Khác với SGD, tốc độ học trong Adagrad thay đổi tùy thuộc vào trọng số: tốc độ học thấp đối với các trọng số tương ứng với các đặc trưng phổ biến, tốc độ học cao đối với các trọng số tương ứng với các đặc trưng ít phổ biến.

Ký hiệu g_t là gradient của hàm mất mát tại bước t , $g_{t,i}$ là đạo hàm riêng của hàm mất mát theo $\boldsymbol{\theta}_i$ tại bước t.

$$g_{t,i} = \nabla_\theta J\left(\theta_{t,i}\right)$$

Quy tắc cập nhật của Adagrad :

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \varepsilon}} \cdot g_{t,i}$$

Theo quy tắc cập nhật, Adagrad điều chỉnh tốc độ học η tại bước t tương ứng với trọng số $\boldsymbol{\theta}_i$ xác định dựa trên các gradient đã tính được theo $\boldsymbol{\theta}_i$. Mẫu số là chuẩn L2 (L2 norm) của ma trận đường chéo G_t trong đó phần tử i, i là tổng bình phương của các gradient tương ứng với $\boldsymbol{\theta}_i$ tính đến bước t . ε là một số dương khá nhỏ nhằm tránh trường hợp mẫu số bằng 0. Quy tắc cập nhật trên có thể viết dưới dạng tổng quát hơn như sau:

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

Trong đó, Θ là phép nhân ma trận-vectơ giữa G_t và g_t . Có thể nhận thấy rằng trong thuật toán Adagrad tốc độ học được tự động điều chỉnh. Adagrad thường khá hiệu quả đối với bài toán có dữ liệu phân mảnh. Tuy nhiên, hạn chế của Adagrad là các tổng bình phương ở mẫu số ngày càng lớn khiến tốc độ học ngày càng giảm và

có thể tiệm cận đến giá trị 0 khiến cho quá trình huấn luyện gần như đóng băng. Bên cạnh đó, giá trị tốc độ học η cũng phải được xác định một cách thủ công.

1.1.4 Adadelta:

Adadelta là một biến thể của Adagrad để khắc phục tình trạng giảm tốc độ học ở Adagrad. Thay vì lưu lại tất cả gradient như Adagrad, Adadelta giới hạn tích lũy gradient theo cửa sổ có kích thước *w* xác định. Bằng cách này, Adadelta vẫn tiếp tục học sau nhiều bước cập nhật.

Trong quá trình thực hiện, thay vì lưu trữ w bình phương của gradient theo cách thông thường, Adadelta thực hiện tích lũy dưới dạng mô-men bậc 2 của gradient:

$$E\left[g^2\right]_t = \gamma E\left[g^2\right]_{t-1} + (1-\gamma) g_t^2$$

Công thức trên thể hiện trung bình các gradient $E\ [\ g^2\ ]_t$ ở bước t phụ thuộc vào trung bình các gradient $E\ [\ g^2\ ]_{t-1}$ ở bước t −1 và gradient g_t ở bước t . Hệ số γ thường có giá trị 0.9 với ý nghĩa rằng gradient ở hiện tại sẽ phụ thuộc phần lớn vào gradient ở các bước trước đó. Với thuật toán Adadelta, tốc độ học hoàn toàn được thay thế bởi:

$$\frac{\sqrt{E\left[\Delta\theta^2\right]_{t-1} + \varepsilon}}{\sqrt{E\left[g^2\right]_t + \varepsilon}}$$

Trong đó,

$$E\left[\Delta\theta^2\right]_t = \gamma E\left[\Delta\theta^2\right]_{t-1} + (1-\gamma)\Delta\theta_t^2$$

Tương tự như trường hợp gradient, công thức trên thể hiện độ biến thiên $\Delta\theta$ của θ tại bước t phụ thuộc vào độ biến thiên của θ tại bước $t-1$.

Adadelta được cập nhật theo quy tắc:

$$\theta_{t+1} = \theta_t - \frac{\sqrt{E\left[\Delta\theta^2\right]_{t-1} + \varepsilon}}{\sqrt{E\left[g^2\right]_t + \varepsilon}} \cdot g_t$$

Adadelta không sử dụng tốc độ học. Thay vào đó, nó sử dụng tốc độ thay đổi của chính bản thân các trọng số để điều chỉnh tốc độ học.

1.1.5 RMSProp:

Vấn đề của Adagrad là tốc độ học thực tế được giảm theo một thời điểm được định nghĩa sẵn. Tuy nhiên, cách này thích hợp với các bài toán lồi nhưng có thể không phải là giải pháp lý tưởng chó những bài toán không lồi.

Adagrad cộng dồn tổng bình phương của gradient $\boldsymbol{g_t}$ vào vector trạng thái $\mathbf{S_t} = \mathbf{S_{t-1}} + \boldsymbol{g_t^2}$. Kết quả là do không có phép chuẩn hóa, $\mathbf{S_t}$ vẫn tiếp tục tăng tuyến tính không ngừng trong quá trình hội tụ của thuật toán.

Vấn đề này được giải quyết bằng cách sử dụng $\mathbf{S_t/t}$. Với phân phối $\mathbf{g_t}$ hợp lý, thuật toán sẽ hội tụ. Nhưng có thể sẽ mất nhiều thời gian cho đến khi các tính chất tại giới hạn bắt đầu có ảnh hưởng, bởi thuật toán này ghi nhớ toàn bộ quỹ đạo của các giá trị. Một cách khác là sử dụng trung bình rò rỉ tương tự như trong phương pháp động lượng, tức là $\mathbf{s_t \leftarrow \gamma s_{t-1} + (1-\gamma) g_t^2}$ cho các tham số $\boldsymbol{\gamma > 0}$. Giữ nguyên tất cả các phần khác và ta có thuật toán RMSProp.

RMSProp tự động điều chỉnh tốc độ học tập và chọn một tỉ lệ học tập khác nhau cho mỗi tham số. Phương pháp cập nhật các trọng số được thực hiện :

$$s_t \leftarrow \gamma s_{t-1} + (1-\gamma) g_t^2$$

$$\Delta x_t = -\frac{\eta}{\sqrt{s_t + \epsilon}} * g_t$$

$$x_{t+1} = x_t + \Delta x_t$$

Với S_t : tích lũy phương sai của các gradient trong quá khứ,

γ : tham số suy giảm

Δx_t : sự thay đổi các tham số trong mô hình

G_t : gradient của các tham số tại vòng lặp t

ϵ : tham số đảm bảo kết quả xấp xỉ có ý nghĩa

1.1.6 Adam:

- Chúng ta nhận thấy rằng, SGD hiệu quả hơn hạ gradient khi giải các bài tonas tối ưu, ít chịu anh hưởng xấu gây ra bởi dữ liệu thừa

- Mini Batch SGD mang lại hiệu quả đáng kể nhờ việc vector hóa, tức là xử lí nhiều mẫu quan sát hơn trong một mini Batch. Đây là chìa khóa để xử lí dữ liệu song song trên nhiều GPU và nhiều máy tính một cách hiệu quả.

- Phương pháp động lượng bổ sung cơ chế gộp các gradient quá khứ, giúp quá trình hội tụ diễn ra nhanh hơn

- Adagrad sử dụng phép biến đổi tỉ lệ theo từng tọa độ để tạo ra tiền điều kiện hiệu quả về mặt tính toán.

- RMSProp tách rời phép biến đổi tỉ lệ theo từng tọa độ khỏi phép điều chỉnh tốc độ học

***Adam** kết hợp tất cả các kỹ thuật trên thành một thuật toán học hiệu quả. Đây là một trong những thuật toán tối ưu mạnh mẽ và hiệu quả được sử dụng phổ biến. Tuy nhiên, có nhiều trường hợp mà Adam có thể phân kỳ do việc kiểm soát phương sai kém.

Một trong những thành phần chính của Adam là các trung bình động trong số mũ (hay còn được gọi là trung bình rò rỉ) để ước lượng cả động lượng và mô-men bậc hai của gradient. Cụ thể, nó sử dụng biến trạng thái :

$$\mathbf{v}_t \leftarrow \beta_1 \mathbf{v}_{t-1} + (1 - \beta_1)\mathbf{g}_t,$$
$$\mathbf{s}_t \leftarrow \beta_2 \mathbf{s}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2.$$

Ở đây, $\beta 1$, $\beta 2$ là các tham số trọng số không âm. Thường được chọn là $\beta 1$ = ***0.9 và*** $\beta 2$ = ***0.999.***

V_t là trung bình động của bình phương và S_t là trung bình động của gradient

Là một biến thể của Adam, ý tưởng của AdamW khá đơn giản đó là : thực hiện thuật toán Adam với L2 regularization (chuẩn hóa L2), tác giả loại bỏ phần tiêu biến của trọng số (weight decay) $w_t\theta_t$ khỏi công thức tính gradient hàm mất mát tại hời điểm *t :*

$$g_t = \nabla f(\theta_t) + w_t\theta_t$$

Và thay vào đó, đưa phần giá trị đã được phân tách này vào quá trình cập nhật trọng số :

$$\theta_{t+1,i} = \theta_{t,i} - \eta \left(\frac{1}{\sqrt{\hat{v}_t + \epsilon}} \cdot \hat{m}_t + w_{t,i}\theta_{t,i} \right), \forall t$$

1.1.7 AdamW:

Là một biến thể của Adam, ý tưởng của AdamW khá đơn giản đó là : thực hiện thuật toán Adam với L2 regularization (chuẩn hóa L2), tác giá loại bỏ phần tiêu biến của trọng số (weight decay) $w_t\theta_t$ khỏi công thức tính gradient hàm mất mát tạit hời điểm *t :*

$$g_t = \nabla f(\theta_t) + w_t\theta_t$$

Và thay vào đó, đưa phần giá trị đã được phân tách này vào quá trình cập nhật trọng số :

$$\theta_{t+1,i} = \theta_{t,i} - \eta \left(\frac{1}{\sqrt{\hat{v}_t + \epsilon}} \cdot \hat{m}_t + w_{t,i}\theta_{t,i} \right), \forall t$$

1.1.8 AMSGrad:

AMSGrad sử dụng giá trị lớn nhất của các bình phương gradient trước đó v_t để cập nhật các trọng số. Ở đây, v_t cũng được định nghĩa như trong thuật toán Adam:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Thay vì trực tiếp sử dụng v_t (hay giá trị ước lượng $\hat{v}_t$), thuật toán sẽ sử dụng giá trị trước đó v_{t-1} nếu giá trị này lớn hơn giá trị hiện tại :

$$\hat{v}_t = max\left(\hat{v}_{t-1}, v_t\right)$$

Tương tự như trong thuật toán Adam, các giá trị được ước lượng theo công thức dưới dây để khử lệnh cho các trọng số:

$$m_t = \beta_1 m_{t-1} + \left(1-\beta_1\right) g_t ; \qquad v_t = \beta_2 v_{t-1} + \left(1-\beta_2\right) g_t^2 ; \qquad \hat{v}_t = max\left(\hat{v}_{t-1}, v_t\right)$$

AMSGrad được cập nhật theo quy tắc:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} m_t$$

1.2 So sánh các phương pháp Optimizer:

1. Thuật toán SGD có thời gian huấn luyện thấp, tốc độ hội tụ chậm nhưng ổn định, và trong hai thực nghiệm đều đạt được ngưỡng độ chính xác đề ra. Tuy SGD luôn đem lại một trong các tần suất lỗi nhỏ nhất, khi quá trình huấn luyện tương đối dài, độ chính xác trên tập kiểm thử có xu hướng giảm. Để khắc phục hiện tượng này, có thể đưa thêm một số mốc chu kỳ lớn hơn vào phương án lập lịch tốc độ học. Điều này cũng chứng minh rằng hiệu quả của thuật toán SGD phụ thuộc rất lớn vào việc tinh chỉnh thủ công tốc độ học.
2. Thuật toán Adagrad và Adadelta yêu cầu thời gian huấn luyện lớn hơn SGD, trong đó thời gian huấn luyện khi sử dụng Adadelta là lớn nhất trong các thuộc toán được so sánh. Tuy nhiên, hai thuật toán này không đem lại kết quả tương xứng: đều không đạt được ngưỡng độ

chính xác đề ra. Do đó, chúng tôi không khuyến khích sử dụng hai thuật toán này.

3. Thuật toán Adam hội tụ nhanh nhưng không đạt được ngưỡng độ chính xác đề ra. Chỉ sử dụng Adam nếu thời gian huấn luyện hạn chế và không yêu cầu quá cao về tính chính xác của mô hình.
4. Biến thể của AdamW của Adam kết hợp cả ưu điểm của Adam và SGD. AdamW có thời gian huấn luyện không quá lớn so với SGD hay Adam, đem lại tốc độ hội tụ nhanh như Adam và tần suất lỗi thấp như SGD. Cũng như Adam, AdamW kế thừa khả năng tự động tinh chỉnh tốc độ học của mình trong quá trình huấn luyện. Với những ưu điểm trên, tôi đề xuất sử dụng AdamW như một thuật toán lý tưởng thay thế cho SGD.
5. Việc tích hợp AMSGrad vào Adam không giúp cải thiện khả năng hội tụ. Trong mọi trường hợp, AMSGrad khiến thời gian huấn luyện tăng lên đáng kể. Đối với AdamW, AMSGrad đem lại kết quả không đồng nhất. Do đó, tôi cho rằng nên thận trọng khi tích hợp AMSGrad vào Adam hoặc AdamW.
6. Việc lập lịch tốc độ học có ảnh hưởng tích cực không chỉ đến SGD mà cả các thuật toán với tốc độ học thích ứng như Adam: đa phần các thuật toán đều sớm đạt được ngưỡng độ chính xác đề ra ngay sau khi vượt qua các cột mốc chu kỳ trong lịch.

CHƯƠNG 2. TÌM HIỂU VỀ CONTINUAL LEARNING VÀ TEST PRODUCT KHI XÂY DỰNG MỘT GIẢI PHÁP HỌC MÁY

2.1 Continual Learning:

2.1.1 Định nghĩa:

Học liên tục, còn được gọi là học máy liên tục (CML), là một quá trình trong đó một mô hình học từ các luồng dữ liệu mới mà không cần đào tạo lại.

Trái ngược với các phương pháp tiếp cận truyền thống, trong đó các mô hình được đào tạo trên một tập dữ liệu tĩnh, được triển khai và đào tạo lại định kỳ, các mô hình học liên tục cập nhật lặp đi lặp lại các tham số của chúng để phản ánh các phân phối mới trong dữ liệu.

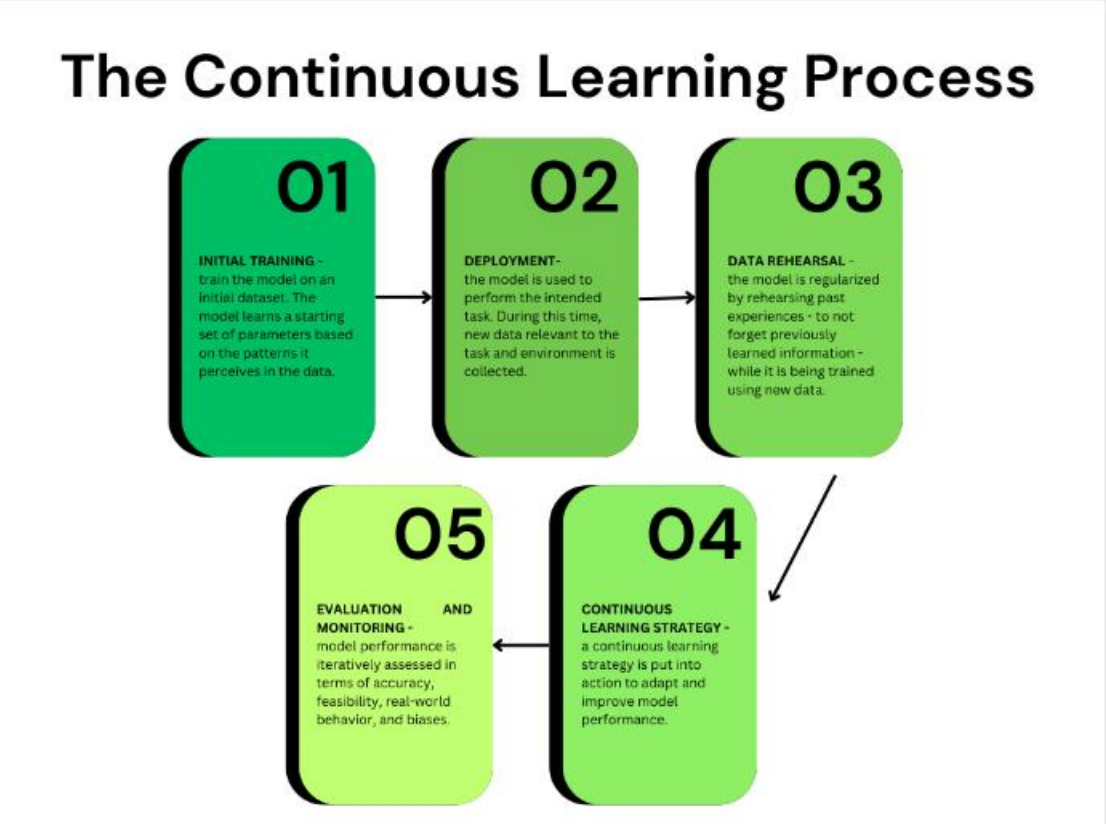
Trong quy trình sau, mô hình tự cải thiện bằng cách học hỏi từ lần lặp mới nhất và cập nhật kiến thức của nó khi có dữ liệu mới. Vòng đời của mô hình học tập liên tục cho phép các mô hình duy trì tính phù hợp theo thời gian nhờ chất lượng năng động vốn có của chúng.

2.1.2 Phân loại:

Có nhiều phương pháp học máy liên tục để lập mô hình. Các chiến lược phổ biến bao gồm học tăng dần, học chuyển tiếp và học tập suốt đời.

Học liên tục là một sự phát triển của mô hình học máy truyền thống. Do đó, nó liên quan đến nhiều nguyên tắc lập mô hình giống nhau: tiền xử lý, lựa chọn mô hình, điều chỉnh siêu tham số, đào tạo, triển khai và giám sát.

Hai bước bổ sung cần thiết trong quá trình học tập liên tục: diễn tập dữ liệu và thực hiện chiến lược học tập liên tục. Các bước này nhằm đảm bảo rằng mô hình đang học từ các luồng dữ liệu mới một cách hiệu quả dựa trên ứng dụng và bối cảnh của nhiệm vụ dữ liệu.



Hình 1 : Sơ đồ quá trình học tập liên tục đến học máy

2.1.3 Ưu điểm :

Lợi ích so với phương pháp học máy truyền thống bao gồm:

Sự khái quát : Việc học hỏi liên tục giúp mô hình trở nên mạnh mẽ và chính xác hơn khi đối mặt với dữ liệu mới.

Lưu giữ thông tin : Bằng cách sử dụng chiến lược học tập liên tục, mô hình sẽ xem xét kiến thức trước đó 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 sử dụng việc học tập liên tục sẽ thích ứng với kiến thức mới - chẳng hạn như sự thay đổi khái niệm và xu hướng mới - từ đó có khả năng dự đoán tốt hơn về lâu dài.

2.1.4 Hạn chế :

Một phương pháp học tập liên tục không có những hạn chế. Những điều này nên được xem xét trước khi lựa chọn cách tiếp cận kiểu này dựa trên tính chất và kết quả mong đợi của nhiệm vụ hiện tại.

Từ quan điểm hoạt động, một thách thức là:

Trị giá . Các phương pháp học liên tục, tuy hiệu quả nhưng cũng có xu hướng phức tạp hơn về mặt tính toán so với các phương pháp truyền thống vì mô hình cần phải thích ứng nhất quán với dữ liệu mới. Sự phức tạp nói trên thường dẫn đến chi phí kinh tế cao hơn vì nó đòi hỏi nhiều tài nguyên dữ liệu, con người và máy tính hơn.

Từ góc độ mô hình hóa, một số nhược điểm là:

Quản lý mô hình . Mỗi khi các tham số của mô hình cập nhật dựa trên dữ liệu mới, một mô hình mới sẽ được hình thành. Do đó, cách tiếp cận học tập liên tục có thể tạo ra một số lượng lớn các mô hình, làm phức tạp việc xác định những mô hình hoạt động tốt nhất.

Dữ liệu trôi dạt . Để phương pháp học tập liên tục có giá trị, chúng ta phải xử lý một khối lượng lớn dữ liệu mới. Tuy nhiên, mô hình như vậy có nguy cơ mất khả năng dự đoán nếu phân bổ tính năng thay đổi đột ngột. Tìm hiểu thêm về sự trôi dạt dữ liệu trong một bài viết riêng.

Mặc dù chi phí không phải là một hạn chế dễ vượt qua, nhưng những thách thức liên quan đến mô hình hóa có thể được giảm bớt bằng cách áp dụng một phương pháp phù hợp và thông qua sự can thiệp của con người. Các hoạt động như lập phiên bản mô hình, giám sát và đánh giá là chìa khóa để theo dõi hiệu suất của mô hình. Ngoài ra, sự can thiệp của con người rất quan trọng để thực thi các biện pháp được đề cập ở trên và đưa ra các lựa chọn ngẫu nhiên về dữ liệu, chẳng hạn như tần suất và quy mô cập nhật.

2.1.5 Ứng dụng :

Do chi phí bổ sung và độ phức tạp phát sinh từ việc học liên tục, phương pháp này phù hợp nhất cho các ứng dụng liên quan đến luồng dữ liệu mới liên tục. Điều này đòi hỏi môi trường của nhiệm vụ dữ liệu phải không ngừng phát triển.

Các ứng dụng hiện tại của học tập liên tục bao gồm:

Thị giác máy tính. Bản chất năng động của dữ liệu dựa trên hình ảnh làm cho các phương pháp học liên tục trở nên phổ biến đối với các thuật toán huấn luyện nhằm xác định và phân loại thông tin hình ảnh. Chúng thường được áp dụng trong công nghệ hình ảnh và nhận dạng khuôn mặt.

An ninh mạng . Các phương pháp học tập liên tục được triển khai để đảm bảo giám sát liên tục trong cơ sở hạ tầng bảo mật CNTT. Chúng là chìa khóa trong việc phát hiện hành vi lừa đảo, xâm nhập mạng và thư rác, cùng các hoạt động liên quan đến bảo mật khác.

Chăm sóc sức khỏe . Do bản chất ngày càng phát triển của bệnh tật, các phương pháp học tập liên tục được sử dụng trong các lĩnh vực chăm sóc sức khỏe khác nhau để tăng cường quy trình chẩn đoán xung quanh việc chẩn đoán bệnh. Các chuyên ngành như ung thư và X quang đã là những người đi đầu trong việc khám phá AI học tập liên tục để đạt được mục tiêu này.

Người máy: Học tập liên tục đã được sử dụng để nâng cao khả năng thích ứng và hiệu suất của robot trong các môi trường khác nhau, cho phép chúng tối ưu hóa hành động của mình trong các điều kiện thay đổi dựa trên kinh nghiệm trong quá khứ và mới.

2.2 Test Product:

Test Prodcut là quá trình đánh giá hiệu suất của mô hình máy học trên dữ liệu mà nó chưa từng thấy trước đó. Mục tiêu là đưa ra một ước lượng về khả năng tổng quát hóa của mô hình, tức là khả năng áp dụng kiến thức đã học từ dữ liệu huấn luyện vào dữ liệu mới

DANH MỤC TÀI LIỆU THAM KHẢO

[1] Lifelong Machine Learning, 2018, Morgan & Claypool Publishers. https://books.google.ca/books/about/Lifelong_Machine_Learning.html?id=JQ5pDwAAQBAJ&redir_esc=y

[2] Machine learning for absolute beginners, 2017, Oliver Theobald. bit.ly/MachineLearningForAbsoluteBeginners