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

LÊ MINH TRIẾT - 521H0173

BÀI 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

LÊ MINH TRIẾT - 521H0173

BÀI CUỐI KỲ
NHẬP MÔN HỌC MÁY

Người hướng dẫn
TS. Lê Anh Cường

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Thầy Lê Anh Cường, người đã hướng dẫn và hỗ trợ tận tình cho nhóm chúng em trong quá trình thực hiện báo cáo này. Sự kiên nhẫn, kiến thức chuyên môn, và kinh nghiệm quý báu của Thầy đã giúp nhóm chúng em nắm bắt được những kiến thức cần thiết và áp dụng chúng một cách hiệu quả để hoàn thiện báo cáo. Sự giảng dạy và sự hỗ trợ của Thầy không chỉ giúp chúng em vượt qua những khó khăn trong quá trình làm báo cáo, mà còn là nguồn cảm hứng để chúng em phấn đấu và học hỏi nhiều hơn nữa. Chúng em xin chân thành cảm ơn và mong rằng sẽ tiếp tục nhận được sự hỗ trợ và hướng dẫn của Thầy trong tương lai.

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

Tác giả

(Ký tên và ghi rõ họ tên)

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 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

(Ký tên và ghi rõ họ tên)

CHƯƠNG 1. CÁC PHƯƠNG PHÁP OPTIMIZER TRONG HUẤN LUYỆN MÔ HÌNH

1.1 Optimizer trong học máy

1.1.1 Khái niệm

Optimizer trong học máy là một khái niệm chỉ các thuật toán hoặc phương pháp được áp dụng để cải thiện hiệu suất của mô hình học máy trong quá trình huấn luyện. Optimizer có nhiệm vụ điều chỉnh các tham số của mô hình, thường là các trọng số liên kết giữa các tầng mạng, sao cho hàm mất mát đạt giá trị nhỏ nhất có thể. Hàm mất mát là một hàm số biểu diễn sự sai khác giữa kết quả dự đoán của mô hình và giá trị thực tế của dữ liệu. Bằng cách tối ưu hóa hàm mất mát, mô hình học máy có thể học được các đặc trưng (features) của dữ liệu và dự đoán hoặc phân loại dữ liệu một cách chính xác hơn.

1.1.2 Các loại Optimizer trong học máy

Có nhiều loại optimizer khác nhau trong học máy, ví dụ như Stochastic Gradient Descent, Momentum, Adagrad, RMSprop, Adam, v.v. Mỗi loại optimizer có những ưu và nhược điểm riêng, cũng như các tham số cần được điều chỉnh, như tốc độ học (learning rate), động lượng (momentum), trọng số suy giảm (weight decay), v.v. Việc lựa chọn và sử dụng optimizer phù hợp là một yếu tố quan trọng để nâng cao hiệu quả của mô hình học máy.

1.2 Stochastic Gradient Descent (SGD)

1.2.1 Phương pháp

- SGD là thuật toán tối ưu đơn giản nhất. Tại mỗi bước thời gian, các tham số (trọng số) của mô hình được cập nhật bằng cách di chuyển theo hướng âm của gradient của hàm mất mát. Cập nhật được điều chỉnh bởi một siêu tham số tốc độ học (ETA).

- Update rule:

  $\boldsymbol{\theta_{t+1} = \theta_t - \eta \nabla J(\theta_t)}$

  Trong đó:

  θ_t là vector tham số tại bước thời gian t,

  η là tốc độ học,

  $\nabla J(\theta_t)$ là độ dốc của hàm mất mát theo các tham số.

1.2.2 Ưu điểm

- Đơn giản: SGD dễ triển khai và hiểu.
- Hội tụ nhanh trên các tập dữ liệu lớn: SGD có thể hội tụ nhanh trên các tập dữ liệu lớn nhờ tính ngẫu nhiên của nó.

1.2.3 Nhược điểm

- Lựa chọn tốc độ học thủ công: Tốc độ học cần được thiết lập thủ công trước quá trình huấn luyện và việc chọn một giá trị tối ưu có thể khó khăn.
- Rơi vào điểm cực tiểu cục bộ: SGD dễ bị mắc kẹt trong các điểm cực tiểu cục bộ tối ưu hơn.

1.3 Momentum

1.3.1 Phương pháp

- Momentum là một sự mở rộng của SGD, giới thiệu một thuật ngữ vận tốc vào quy tắc cập nhật. Tại mỗi bước thời gian, các tham số của mô hình được cập nhật bằng cách thêm một phần của vận tốc trước đó và một phần của gradient hiện tại. Thuật ngữ vận tốc giúp làm mịn quỹ đạo và giảm thiểu sự ngẫu nhiên của chuyển động trong không gian tham số.
- Update rule:

  $\boldsymbol{v_{t+1} = \beta v_t + (1 - \beta)\, \nabla J(\theta_t)}$

  $\boldsymbol{\theta_{t+1} = \theta_t - \eta v_{t+1}}$

  Trong đó:

  v_t là thành phần vận tốc tại bước thời gian t,

β là hệ số đà.

1.3.2 Ưu điểm

- Hội tụ nhanh hơn: Momentum có thể tăng tốc quá trình hội tụ bằng cách cho phép mô hình đi qua nhiều không gian trong thời gian ngắn hơn.
- Giảm thiểu điểm cực tiểu cục bộ: Momentum có thể giúp mô hình thoát khỏi các điểm cực tiểu cục bộ hời hợt.

1.3.3 Nhược điểm

- Nhạy cảm với siêu tham số: Siêu tham số vận tốc cần được điều chỉnh cẩn thận để đạt hiệu suất tối ưu.
- Rủi ro vượt mục tiêu: Momentum có thể làm cho mô hình vượt quá giới hạn tối thiểu mục tiêu.

1.4 Adagrad

1.4.1 Phương pháp

- Adagrad điều chỉnh tốc độ học cho mỗi tham số cá nhân dựa trên gradient bình phương lịch sử. Các tham số có gradient tích lũy lớn sẽ có tốc độ học nhỏ hơn, trong khi các tham số có gradient tích lũy nhỏ sẽ có tốc độ học lớn hơn.
- Update rule:

$$\boldsymbol{G_{t+1} = G_t + (\nabla J(\theta_t))^2}$$

$$\boldsymbol{\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_{t+1}+\epsilon}} \nabla J(\theta_t)}$$

Trong đó:

G_t là tổng các độ dốc bình phương tích lũy,

ϵ là một hằng số nhỏ để đảm bảo tính ổn định số học.

1.4.2 Ưu điểm

- Tốc độ học thích nghi: Adagrad tự động điều chỉnh tốc độ học dựa trên gradient của mỗi tham số.

- Hiệu quả với dữ liệu thưa: Adagrad hoạt động tốt trên các vấn đề có dữ liệu thưa.

1.4.3 Nhược điểm

- Giảm tốc độ học theo thời gian: Adagrad có thể giảm tốc độ học quá nhiều, gây ra quá trình hội tụ chậm hoặc dừng học sớm.
- Khó khăn với các vấn đề phi lồi: Adagrad có thể gặp khó khăn trong việc tối ưu hóa các vấn đề phi lồi và bị mắc kẹt trong các giải pháp tối ưu hóa không tối ưu.

1.5 RMSprop

1.5.1 Phương pháp

- RMSprop giải quyết vấn đề tốc độ học giảm quá nhiều bằng cách giới thiệu một tham số suy giảm. Nó tính trung bình trọng số mũ của gradient bình phương và điều chỉnh tốc độ học tương ứng.
- Update rule:

$$\boldsymbol{G_{t+1} = \beta G_t + (1-\beta)(\nabla J(\theta_t))^2}$$

$$\boldsymbol{\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_{t+1}+\epsilon}} \nabla J(\theta_t)}$$

Trong đó:

G_t là trung bình trọng số của độ dốc bình phương,

β là tham số giảm,

ϵ là một hằng số nhỏ để đảm bảo tính ổn định số học.

1.5.2 Ưu điểm

- Tốc độ học thích nghi: RMSprop điều chỉnh tốc độ học dựa trên độ lớn của gradient gần đây, cho phép nó hội tụ nhanh hơn.
- Hiệu quả cho các mục tiêu không ổn định: RMSprop hoạt động tốt trên các vấn đề có mục tiêu không ổn định.

1.5.3 Nhược điểm

- Khó khăn với các điểm sô cô la: RMSprop có thể gặp khó khăn trong việc thoát khỏi các điểm sô cô la tối ưu hóa trong không gian.
- Nhạy cảm với siêu tham số: Tham số suy giảm cần được điều chỉnh đúng để đạt được kết quả tốt.

1.6 Adam

1.6.1 Phương pháp

- Adam kết hợp lợi ích của RMSprop và Momentum. Nó sử dụng các moment đầu và moment hai của gradient để điều chỉnh tốc độ học thích nghi và làm mịn quỹ đạo. Adam cũng bao gồm các thuật ngữ sửa lỗi độ lệch để tính toán sai số khởi tạo.
- Update rule:

$$\boldsymbol{m_{t+1} = \beta_1 m_t + (1 - \beta_1)\nabla J(\theta_t)}$$

$$\boldsymbol{v_{t+1} = \beta_2 v_t + (1 - \beta_2)(\nabla J(\theta_t))^2}$$

$$\boldsymbol{\hat{m}_{t+1} = \frac{m_{t+1}}{(1-\beta_1^t)}}$$

$$\boldsymbol{\hat{v}_{t+1} = \frac{v_{t+1}}{(1-\beta_2^t)}}$$

$$\boldsymbol{\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_{t+1}+\epsilon}}\hat{m}_{t+1}}$$

Trong đó:

$m_t$và v_t là các khoản đầu tiên và thứ hai của độ dốc,

β_1 và $\beta_2$là tỷ lệ giảm moment cho ước lượng moment,

ϵ là một hằng số nhỏ để đảm bảo tính ổn định số học.

1.6.2 Ưu điểm

- Tốc độ học thích nghi: Adam tự động điều chỉnh tốc độ học dựa trên các moment đầu và moment hai của gradient.
- Hội tụ nhanh: Adam thường hội tụ nhanh hơn các thuật toán tối ưu hóa khác.

1.6.3 Nhược điểm

- Nhạy cảm với siêu tham số: Adam yêu cầu điều chỉnh cẩn thận các siêu tham số để đạt hiệu suất tối ưu.
- Yêu cầu bộ nhớ: Adam yêu cầu nhiều bộ nhớ hơn để lưu các ước tính moment đầu và moment hai.

CHƯƠNG 2. CONTINUAL LEARNING VÀ TEST PRODUCTION

2.1 Continual Learning

2.1.1 Khái niệm

Học liên tục (Continuous Learning) là một khái niệm quan trọng trong lĩnh vực Trí tuệ Nhân tạo và Học máy. Nó liên quan đến việc học từ nhiều nhiệm vụ mà không quên những kiến thức đã học trước đó. Điều này cho phép các mô hình AI có khả năng thích ứng với sự thay đổi trong dữ liệu và môi trường, đồng thời giảm thiểu hiện tượng quên kiến thức khi học nhiều nhiệm vụ.

2.1.2 Ứng dụng

Học liên tục có ứng dụng rộng rãi trong nhiều lĩnh vực thực tế. Ví dụ, trong lĩnh vực nhận diện khuôn mặt, học liên tục cho phép mô hình AI nâng cao khả năng nhận diện và phân loại các khuôn mặt mới dựa trên những kiến thức đã học từ các khuôn mặt trước đó. Trong xử lý ngôn ngữ tự nhiên, học liên tục giúp mô hình AI hiểu và tương tác với ngôn ngữ một cách tự nhiên và linh hoạt hơn. Ngoài ra, học liên tục còn được áp dụng trong việc phát hiện bất thường, học tăng cường và nhiều lĩnh vực khác.

2.1.3 Các thuật toán phổ biến

Có nhiều thuật toán phổ biến được sử dụng trong học liên tục. Một trong số đó là Progressive Neural Networks, một thuật toán cho phép mô hình AI mở rộng kiến thức của mình khi đối mặt với các nhiệm vụ mới. Thuật toán Learning without Forgetting giúp mô hình AI học từ nhiều nhiệm vụ mà không quên những kiến thức đã học trước đó. Variational Continual Learning là một thuật toán khác trong học liên tục, nó tối ưu hóa quá trình học liên tục bằng cách sử dụng phương pháp biến phân. Học liên tục cũng có thể kết hợp với các phương pháp học máy khác như học có giám sát và học không giám sát. Việc kết hợp này giúp tăng cường khả năng học và thích ứng của mô hình AI trong các tình huống khác nhau.

2.2 Test Production

2.2.1 Khái niệm

Test Production là một kỹ thuật kiểm thử phần mềm tiên tiến dựa trên học máy. Phương pháp tiếp cận này được thiết kế để tạo ra các ca kiểm thử tự động, đóng góp lớn cho các nhóm phát triển phần mềm. Bằng cách tận dụng sức mạnh của các thuật toán học máy, Test Production có thể phân tích lượng dữ liệu lớn và xác định các mẫu và xu hướng có thể được sử dụng để tạo ra các ca kiểm thử hiệu quả.

2.2.2 Ưu điểm

Một trong những ưu điểm quan trọng của Test Production là khả năng tiết kiệm thời gian và chi phí trong quá trình kiểm thử phần mềm. Ngoài việc tiết kiệm thời gian và chi phí, Test Production cũng có tác động tích cực đến chất lượng tổng thể của các sản phẩm phần mềm. Bằng cách tự động hóa quá trình tạo ra các ca kiểm thử, Test Production đảm bảo rằng tất cả các kịch bản và trường hợp biên được kiểm tra kỹ lưỡng. Điều này giúp phát hiện các lỗi và vấn đề tiềm ẩn mà có thể không được phát hiện, dẫn đến sản phẩm cuối cùng có chất lượng cao hơn. Hơn nữa, việc sử dụng các thuật toán học máy cho phép Test Production liên tục học và thích nghi, cải thiện độ chính xác và hiệu quả theo thời gian.

2.2.3 Các bước trong Test production

- Bước 1: Thu thập và phân tích dữ liệu về yêu cầu, mục tiêu, chức năng và đặc điểm của phần mềm.
- Bước 2: Xây dựng mô hình học máy để học từ dữ liệu và tạo ra các ca kiểm thử phù hợp.
- Bước 3: Chạy các ca kiểm thử trên môi trường kiểm thử và thu thập kết quả.
- Bước 4: Phân tích kết quả kiểm thử và đánh giá hiệu quả của mô hình học máy.

- Bước 5: Cải tiến mô hình học máy và lặp lại các bước trên cho đến khi đạt được mức độ hài lòng.