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

ĐỒ ÁN CUỐI KÌ

NHẬP MÔN HỌC MÁY

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

Người thực hiện: **TRẦN QUỐC BẢO – 521H0494**

Nhóm : **04**

Khoá : **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

ĐỒ ÁN CUỐI KÌ

NHẬP MÔN HỌC MÁY

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

Người thực hiện: **TRẦN QUỐC BẢO – 521H0494**

Nhóm : **04**

Khoá : **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Lời cảm ơn chân thành đầu tiên em muốn gửi tới PGS. TS. Lê Anh Cường, người đã nhiệt tình giảng dạy môn học Nhập môn Học máy trong học kỳ I (2023-2024), để em nắm bắt được những kiến thức nền tảng cũng như kỹ năng để hoàn thành báo cáo này. Trong quá trình thực hiện đồ án, vì còn tồn đọng nhiều mặt hạn chế về kiến thức, em không tránh khỏi những sai sót nhất định. Tuy nhiên, em xem những sai sót này như cơ hội để học hỏi và cải thiện kỹ năng của mình. Bên cạnh đó, em rất mong nhận được sự cảm thông cũng như những đóng góp, ý kiến của thầy. Một lần nữa, em xin chân thành cảm ơn và kính chúc Thầy sức khỏe, hạnh phúc và thành công trong công việc giảng dạy tiếp theo.

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 / chúng 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 tôi gây ra trong quá trình thực hiện (nếu có).

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

Tác giả

Trần Quốc Bảo

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)

TÓM TẮT

Nội dung của bài báo cáo này bao gồm những yêu cầu của bài làm cá nhân trong đồ án cuối kỳ môn Machine Learning, bao gồm:

- Các phương pháp tối ưu hóa mô hình học máy.
- Tìm hiểu về Contrinual Learning và Test Production.

MỤC LỤC

DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

CÁC KÝ HIỆU

CÁC CHỮ VIẾT TẮT

GD	Gradient Descent
SGD	Stochastic Gadient Descent
BGD	Batch Gradient Descent
RMSprop	Root Mean Square Propagation

CHƯƠNG 1 – TÌM HIỂU VỀ PHƯƠNG PHÁP OPTIMIZER TRONG HUẤN LUYỆN MÔ HÌNH HỌC MÁY

1.1 Khái niệm

Trong học máy, Optimazer là một thuật toán hoặc phương pháp được sử dụng để giảm thiểu hoặc tối đa hóa hiệu suất hoạt động của mô hình huấn luyện.

Trong quá trình huấn luyện mô hình học máy, việc lựa chọn thuật toán tối ưu hóa phù hợp sẽ quyết định tốc độ huấn luyện và hiệu suất dự đoán cuối cùng của mô hình.

Các thuật toán tối ưu hóa rất quan trọng vì chúng ảnh hưởng rất lớn đến độ chính xác và tốc độ huấn luyện của mô hình học sâu.

1.2 Một số phương pháp tối ưu hóa phổ biến

1.2.1 Gradient Descent

GD là một phương pháp tối ưu hóa cơ bản được sử dụng để tìm ra giá trị tối ưu của hàm mất mát. Ý tưởng cơ bản của GD là cập nhật các tham số bằng cách di chuyển ngược theo đạo hàm của hàm mất mát. GD cập nhật tham số bằng cách sử dụng công thức: $\theta\_new = \theta\_old - learning\_rate * gradient$

Trong đó θ là tham số, learning_rate là tốc độ học, và gradient là đạo hàm của hàm mất mát theo tham số. GD có thể áp dụng cho toàn bộ tập dữ liệu (Batch Gradient Descent) hoặc một mẫu dữ liệu (Stochastic Gradient Descent).

1.2.2 Stochastic Gradient Descent

SGD là một biến thể của GD, mỗi epoch có N điểm dữ liệu thì nó sẽ cập nhật trọng số N lần. Vì thế nên SGD sẽ làm giảm đi tốc độ của 1 epoch nhưng nhìn theo một hướng khác thì 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.

SGD có thể gây ra dao động mạnh hơn xung quanh điểm tối ưu, vì gradient được tính toán trên một mẫu dữ liệu duy nhất.

1.2.3 Adagard

Adagard là một phương pháp tối ưu hóa có learning rate thích ứng (adaptive learning- rate). Nó điều chỉnh learning rate của từng tham số dựa trên lịch sử gradient của chúng. Adagard cập nhật learning rate theo công thức:

learning_rate_new = learning_rate / sqrt(sum_of_squared_gradients),

Trong đó sum_of_squared_gradients là tổng bình phương các gradient đã tính toán trước đó. Adagard giúp giảm learning rate cho các tham số có gradient lớn và tăng learning rate cho các tham số có gradient nhỏ, nó ưu tiên cập nhật các tham số hiếm xuất hiện.

1.2.4 RMSprop

RMSprop là một phương pháp tối ưu hóa khác với learning rate thích ứng. Nó cũng sử dụng lịch sử gradient, nhưng khác với Adagrad, RMSprop chỉ lưu trữ một phạm vi độ dài cố định của lịch sử gradient. RMSprop cập nhật learning rate theo công thức:

learning_rate_new=learning_rate/sqrt(moving_average_of_squared_gradients)

Trong đó moving_average_of_squared_gradients là trung bình di động của bình phương gradient. RMSprop giúp ổn định learning rate và giảm độ dao động xung quanh điểm tối ưu.

1.2.5 Adam

Adam (Adaptive Moment Estimation) kết hợp cả learning rate thích ứng và mômentum. Adam tích hợp cả thông tin về gradient và gradient bậc hai. Các bước cập nhật của Adam được tính toán bằng cách kết hợp công thức của RMSprop và momentum (động lượng).

Adam cung cấp khả năng tối ưu mạnh mẽ và linh hoạt trong nhiều tình huống khác nhau và thường được sử dụng phổ biến trong các bài toán học máy.

1.3 So sánh giữa các phương pháp tối ưu hóa

1.3.1 Gradient Descent

- Ưu điểm:
    - Đơn giản và dễ hiểu.
    - Thuật toán giải quyết được vấn đề tối ưu model neural network bằng cách cập nhật trọng số sau mỗi vòng lặp.
    - Có thể đạt được kết quả tốt nếu learning rate được điều chỉnh đúng.
- Nhược điểm:
    - Phụ thuộc vào nghiệm khởi tạo ban đầu và learning rate.
    - Learning rate không thay đổi theo thời gian và không phù hợp cho các bài toán có độ cong khác nhau hoặc bề mặt hàm mất mát phức tạp.
    - Learning rate quá lớn sẽ ảnh hưởng tới sự không hội tụ, learning rate nhỏ ảnh hưởng đến tốc độ training.

1.3.2 Stochastic Gradient Descent

- Ưu điểm:
    - Giải quyết được đối với cơ sở dữ liệu lớn mà GD không làm được và thuật toán này vẫn đang được sử dụng hiện nay.
    - Hiệu quả tính toán vì chỉ tính toán gradient cho một mẫu dữ liệu.
    - Có thể hội tụ nhanh hơn khi bề mặt hàm mất mát không đồng nhất.
- Nhược điểm:
    - Có thể gây ra dao động mạnh và khó hội tụ đến điểm tối ưu chính xác do đạo hàm ước lượng không chính xác.
    - Cần tinh chỉnh learning rate cẩn thận để đạt được kết quả tốt.

1.3.3 Adagard

- Ưu điểm:
    - Learning rate thích ứng giúp điều chỉnh tỷ lệ học tùy thuộc vào lịch sử gradient.
    - Tránh việc điều chỉnh learning rate bằng tay, chỉ cần để learning rate default là 0.01 thì thuật toán sẽ tự động điều chỉnh.
    - Tốt cho các bài toán có dữ liệu thưa (sparse data) và các biến thể của SGD.
- Nhược điểm:
    - Có thể dẫn đến learning rate giảm quá nhanh và dừng lại sớm.
    - Không phù hợp cho bài toán có bề mặt hàm mất mát không đồng nhất.

1.3.4 RMSprop

- Ưu điểm:
    - Giúp ổn định learning rate và giảm độ dao động.
    - Hiệu quả cho các bài toán có bề mặt hàm mất mát không đồng nhất.
- Nhược điểm:
    - Vẫn có thể gặp vấn đề với learning rate quá nhanh hoặc quá chậm.
    - Không phù hợp cho các bài toán có dữ liệu thưa.

1.3.5 Adam

- Ưu điểm:
    - Kết hợp learning rate thích ứng, cung cấp tính linh hoạt và hiệu quả cao.
    - Phù hợp với nhiều loại bài toán và thường cho kết quả tốt.
    - Không đòi hỏi nhiều siêu tham số.
- Nhược điểm:

- Có thể yêu cầu tinh chỉnh nhiều siêu tham số (hyperparameter) và cần sử dụng kỹ thuật regularization để tránh overfitting.

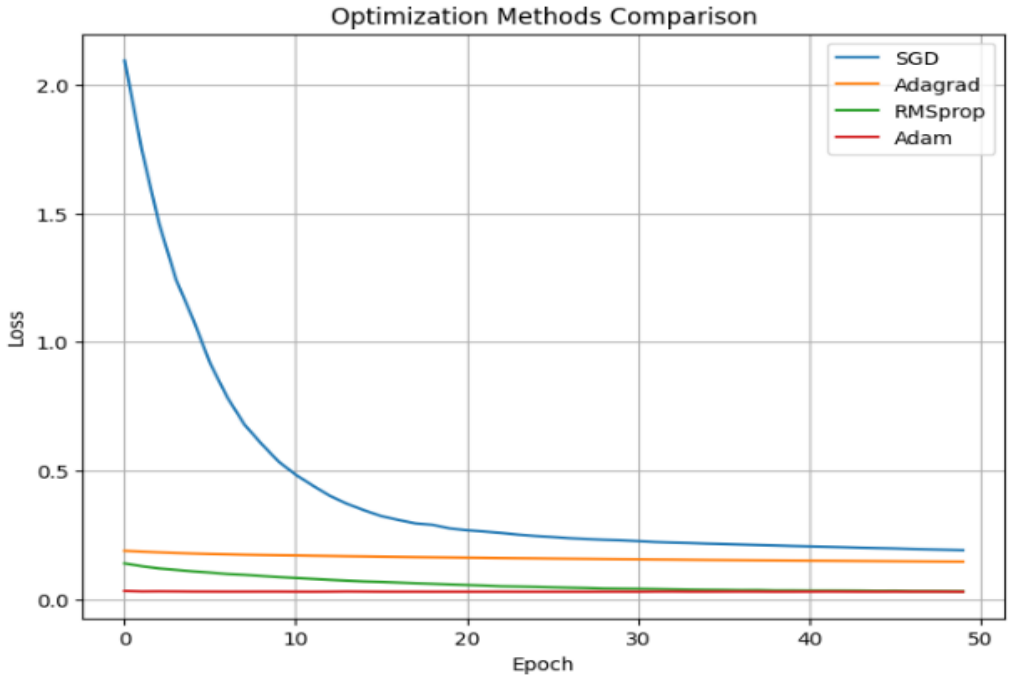
Tổng quan, GD là phương pháp cơ bản và đơn giản, trong khi các phương pháp tối ưu hóa khác như SGD, Adagard, RMSprop và Adam cung cấp những cải tiến để tối ưu hóa quá trình huấn luyện mô hình. Tuy nhiên mỗi phương pháp đều có ưu điểm và nhược điểm nhất định, việc chọn phương pháp phù hợp phụ thuộc vào đặc điểm của bài toán và dữ liệu cụ thể. Thông thường, việc thử nghiệm và tinh chỉnh các phương pháp tối ưu hóa là cần thiết để đạt được kết quả tốt nhất.

1.4 Minh họa sự so sánh giữa các phương pháp tối ưu hóa

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

# Tạo dữ liệu mô phỏng
X = np.random.rand(100, 1)
y = 2 * X + np.random.randn(100, 1) * 0.2

# Khởi tạo mô hình
model = tf.keras.Sequential([
   tf.keras.layers.Dense(1, input_shape=(1,))
])

# Khởi tạo các tối ưu hóa
optimizers = {
```

```
    'SGD': tf.keras.optimizers.SGD(learning_rate=0.01),
    'Adagrad': tf.keras.optimizers.Adagrad(learning_rate=0.01),
    'RMSprop': tf.keras.optimizers.RMSprop(learning_rate=0.01),
    'Adam': tf.keras.optimizers.Adam(learning_rate=0.01)
}

# Huấn luyện mô hình với các tối ưu hóa khác nhau
histories = {}
for optimizer_name, optimizer in optimizers.items():
    model.compile(optimizer=optimizer, loss='mse')
    history = model.fit(X, y, epochs=50, verbose=0)
    histories[optimizer_name] = history

# Vẽ biểu đồ các giá trị loss
plt.figure(figsize=(8, 6))
for optimizer_name, history in histories.items():
    plt.plot(history.history['loss'], label=optimizer_name)
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Optimization Methods Comparison')
plt.legend()
plt.grid(True)
plt.show()
```

Kết quả từ code minh họa trên:



Dựa vào biểu đồ, chúng ta có thể phân tích được kết quả:

- SGD cho thấy sự hội tụ chậm và dao động mạnh. Trong quá trình huấn luyện, loss giảm dần theo từng epoch nhưng vẫn có sự biến động lớn. Cần một số lượng lớn các epochs để đạt được loss nhỏ hơn.
- Adagard có sự hội tụ nhanh và ổn định. Loss giảm nhanh trong vài epochs đầu tiên và sau đó giảm dần đều. Adagard tự điều chỉnh learning rate cho mỗi tham số theo lịch sử của gradient, giúp giảm loss hiệu quả.
- RMSprop cho thấy sự hội tụ nhanh và ổn định. Loss giảm đáng kể trong vài epochs đầu tiên và sau đó giảm chậm hơn. RMSprop sử dụng một phép chuẩn hóa giá trị gradient để điều chỉnh learning rate, giúp mô hình hội tụ tốt hơn.
- Adam cho thấy sự hội tụ rất nhanh và ổn định. Loss giảm rất nhanh trong vài epochs đầu tiên và tiếp tục giảm một cách ổn định.

Từ kết quả của ví dụ minh họa, ta có thể nhận thấy rằng phương pháp tối ưu hóa Adam có hiệu suất tốt nhất. Điều này tương thích với quan sát rằng Adam thường là một phương pháp tối ưu hóa hiệu quả cho nhiều bài toán và mạng neural.

Tuy nhiên, trong các bài toán khác nhau, khả năng của các phương pháp tối ưu hóa có thể khác nhau, và việc lựa chọn phương pháp tối ưu hóa tối nhất vẫn cần được xác định dựa trên thực nghiệm trên dữ liệu thực tế và các thử nghiệm khác.

CHƯƠNG 2 – TÌM HIỂU VỀ CONTRINUAL LEARNING VÀ TEST PRODUCTION KHI XÂY DỰNG GIẢI PHÁP HỌC MÁY

1.1 Contrinual Learning

Contrinual Learning (hoặc còn được gọi là Lifelong Learning hoặc Incremental Learning) là một lĩnh vực quan trọng trong Machine Learning mà mô hình học máy được huấn luyện và cải thiện theo thời gian khi chúng tiếp tục nhận thêm dữ liệu mới mà không quên đi kiến thức đã học trước đó. Điều này đặc biệt hữu ích trong các tình huống khi dữ liệu thay đổi theo thời gian hoặc khi ta muốn mở rộng và cải thiện mô hình mà không phải huấn luyện lại từ đầu.

Trong Continual Learning, có nhiều thách thức phải đối mặt, bao gồm:

- **Catastrophic Forgetting**: Đây là hiện tượng khi model quên dữ liệu đã học trước đó khi nó được huấn luyện với dữ liệu mới. Điều này xảy ra vì các thông tin cũ bị ghi đè bởi thông tin mới trong quá trình huấn luyện.
- **Interference**: Hiện tượng này xảy ra khi dữ liệu đã học trước đó ảnh hưởng đến khả năng học mới. Các dữ liệu trước đó có thể gây nhiễu hoặc xung đột với dữ liệu mới, làm giảm hiệu suất học.
- **Scalability**: Khi số lượng tác vụ hoặc dữ liệu ngày càng tăng, khả năng mở rộng (scalability) của mô hình để xử lý các tác vụ mới và dữ liệu lớn là một thách thức quan trọng.

Để xây dựng giải pháp học máy cho Continual Learning, có một số phương pháp và kỹ thuật quan trọng:

- **Replay-Based Approaches**: Phương pháp này bao gồm việc lưu trữ lại các mẫu dữ liệu đã học trước đó và sử dụng chúng để đào tạo lại mô hình

khi có dữ liệu mới. Việc kết hợp dữ liệu cũ và mới giúp giảm thiểu việc quên đi kiến thức cũ.

- **Regularization Techniques**: Các kỹ thuật regularization như Elastic Weight Consolidation (EWC) và Synaptic Intelligence (SI) được sử dụng để bảo vệ kiến thức đã học trước đó bằng cách áp đặt các ràng buộc trên các trọng số của mô hình.
- **Architectural Approaches**: Các phương pháp kiến trúc như Progressive Neural Networks (PNN) và Memory Aware Synapses (MAS) sử dụng cấu trúc mô hình đặc biệt để giữ và sử dụng kiến thức đã học trước đó một cách hiệu quả.

1.2 Test Production

Test Production liên quan đến việc đảm bảo rằng mô hình học máy hoạt động tốt trên dữ liệu mới và không quen thuộc. Khi triển khai một giải pháp học máy vào môi trường thực tế, việc kiểm tra sản phẩm (test production) đảm bảo rằng mô hình hoạt động như mong đợi và đáp ứng các yêu cầu chất lượng và hiệu suất.

Các hoạt động liên quan đến Test Production bao gồm:

- **Kiểm tra chất lượng**: Đảm bảo rằng mô hình học máy đáp ứng các tiêu chí chất lượng như độ chính xác, độ phân loại đúng, độ tin cậy, và các yêu cầu khác tùy thuộc vào bài toán cụ thể. Điều này thường bao gồm việc sử dụng tập dữ liệu kiểm tra độc lập để đánh giá hiệu suất của mô hình.

- **Kiểm tra hiệu suất**: Đo lường hiệu suất và độ tin cậy của mô hình trên dữ liệu thực tế. Điều này có thể bao gồm việc thu thập dữ liệu từ hệ thống thực tế, đánh giá mô hình trên dữ liệu này và theo dõi các chỉ số hiệu suất theo thời gian.

- **Kiểm tra tính ổn định**: Đảm bảo rằng mô hình hoạt động ổn định trên thực tế và không gặp các vấn đề không mong muốn như lỗi, sự cố, hoặc giảm hiệu suất sau một thời gian sử dụng.

- **Kiểm tra về tốc độ và khả năng mở rộng**: Đánh giá tốc độ và khả năng mở rộng của mô hình khi áp dụng cho dữ liệu lớn hoặc trong môi trường có yêu cầu đáp ứng thời gian thực.

Để đạt được test production thành công, quy trình kiểm tra và đánh giá phải được thiết kế và thực hiện một cách cẩn thận. Đồng thời, việc theo dõi và đánh giá liên tục của mô hình trong môi trường thực tế là quan trọng để phát hiện và giải quyết các vấn đề kịp thời, bảo đảm hiệu suất và chất lượng của giải pháp học máy.

TÀI LIỆU THAM KHẢO

1. Optimizer- Hiểu sâu về các thuật toán tối ưu (GD,SGD,Adam,..) (viblo.asia)
2. Quy's blog (ndquy.github.io)
3. Continual Learning | Papers With Code