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*: **GV. LÊ ANH CƯỜNG**

*Người thực hiện*: **Nguyễn Vĩnh Thi - 51800933**

Lớp **: 18050303**

Khoá **: 22**


**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*: **GV. LÊ ANH CƯỜNG**

*Người thực hiện*: **Nguyễn Vĩnh Thi - 51800933**

Lớp **: 18050303**

Khoá **: 22**


**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

# LỜI CẢM ƠN

Trước tiên, nhóm chúng em xin gửi lời cảm ơn đến các thầy cô Phòng Khoa Công nghệ thông tin đã tạo điều kiện để chúng em được học hỏi, tiếp cận với một môn học mới. Đặc biệt là thầy Lê Anh Cường, cảm ơn thầy vẫn luôn nhiệt tình giúp đỡ, truyền đạt kiến thức môn học thông qua từng buổi học, từng bài tập, từng lời giảng. Nhờ vậy chúng em mới có đủ cơ sở để hoàn thành bài báo cáo cuối kỳ môn  nhập môn học máy hôm nay.

Cảm ơn thầy đã đồng hành và bên cạnh chúng em trong quá trình học tập vừa qua, chúc thầy có nhiều sức khỏe và đạt được nhiều thành công trong cuộc sống.

*TP. Hồ Chí Minh, ngày  24  tháng  12  năm 2023*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*(Đã ký)*

*Nguyễn Vĩnh Thi*

# ĐỒ ÁN / BÁO CÁO ĐƯỢC HOÀN THÀNH
# TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Em xin cam đoan đây là công trình nghiên cứu của riêng em và được sự hướng dẫn khoa học của thầy 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 cuối kỳ 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 cuối kỳ 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 24 tháng 12 năm 2023*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*(Đã ký)*

*Nguyễn Vĩnh Thi*

# TÓM TẮT

Bài báo cáo này trình bày những nội dung mà em sẽ tìm hiểu được cũng như quá trình hiện thực đồ án qua từng giai đoạn khác nhau. Đầu tiên là phần làm cá nhân về một bài nghiên cứu, đánh giá của em về vấn đề sau :

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

2. Tìm hiểu về Continual Learning và Test Production khi xây dựng một giải pháp học máy để giải quyết một bài toán nào đó.

Ngoài ra trong đồ án cuối kì này còn phần làm bài tập nhóm để đưa ra bài toán dự đoán có thể giải quyết bằng học máy (machine leanring) với các yêu cầu:

- Số Feature/Attribute gồm nhiều kiểu: categorial và numerical;
- Dữ liệu phải chưa được học, thực tập trên lớp và trong bài tập về nhà

1. Phân tích thống kê trên dữ liệu, vẽ các đồ thị để hiểu bài toán, hiểu dữ liệu. Tìm hiểu các đặc trưng và đánh gía vai trò của các đặc trưng đối với mục tiêu bài toán.
2. Ứng dụng các mô hình học máy cơ bản để giải quyết bài toán, bao gồm cả các mô hình thuộc Ensemble Learing.
3. Sử dụng Feed Forward Neural Network và Reccurent Neural Network (hoặc mô thuộc loại này) để giải quyết bài toán.
4. Áp dụng các kỹ thuật tránh Overfiting trên các mô hình của câu (2) và câu (3) để giải quyết bài toán.
5. Sau khi huấn luyện xong mô hình thì muốn cải thiện độ chính xác, ta sẽ làm gì để giải quyết nó? Phân tích các trường hợp sai, đề ra giải pháp và thực hiện nó, sau đó đánh giá xem có cải tiến so với trước không.

Phần làm nhóm thì sẽ thực hiện code, data và sẽ được nộp cùng với bài báo cáo này. Vì vậy trong bài báo cáo này chỉ thực hiện phần làm cá nhân.

## PHẦN I. TÌM HIỂU, SO SÁNH CÁC PHƯƠNG PHÁP OPTIMIZER TRONG HUẤN LUYỆN MÔ HÌNH HỌC MÁY

** 10 phương pháp Optimizer trong huấn luyện mô hình học máy:*

### 1. Gradient Descent (GD):

- Phương pháp cơ bản nhất và đơn giản nhất trong tối ưu hóa.

- Cập nhật trọng số bằng cách đi ngược chiều của gradient với một learning rate cố định.

- Không giải quyết được các vấn đề như tốc độ hội tụ chậm hoặc rơi vào các điểm tối ưu cục bộ.

### 2. Stochastic Gradient Descent (SGD):

- SGD là một thuật toán tối ưu hoá lặp được sử dụng phổ biến trong học máy. Nó là một biến thể của gradient descent, thực hiện cập nhật tham số mô hình (trọng số) dựa trên gradient của hàm loss được tính toán trên một tập hợp con được chọn ngẫu nhiên từ dữ liệu huấn luyện, thay vì toàn bộ dữ liệu.

- Ý tưởng cơ bản của SGD là lấy mẫu một tập hợp con nhỏ ngẫu nhiên từ dữ liệu huấn luyện, được gọi là mini-batch, và tính toán gradient của hàm loss đối với các tham số mô hình chỉ sử dụng tập con đó. Gradient này sau đó được sử dụng để cập nhật các tham số.

- Quá trình này được lặp lại với một mini-batch ngẫu nhiên mới cho đến khi thuật toán hội tụ hoặc đạt đến tiêu chí dừng được xác định trước.

- Tăng tốc độ hội tụ và giảm thời gian huấn luyện so với GD.

- Gặp khó khăn khi mô hình có các đặc trưng không đồng nhất hoặc dữ liệu nhiễu.

- Yêu cầu nhiều lần lặp hơn GD: SGD thường yêu cầu nhiều lần lặp hơn gradient descent để hội tụ.

- Nhạy cảm với tốc độ học: Tốc độ học cần được điều chỉnh cẩn thận để đảm bảo hội tụ của SGD.

### 3. Mini-batch Gradient Descent:

- Kết hợp giữa GD và SGD.

- Cập nhật trọng số dựa trên gradient tính toán trên một số lượng nhỏ các mẫu dữ liệu (mini-batch).

- Kompromiss giữa tốc độ hội tụ và hiệu quả tính toán.

**4. Adam (Adaptive Moment Estimation):**

- Adam là một thuật toán tối ưu hóa phổ biến trong huấn luyện mạng nơ-ron để cải thiện hiệu quả và tốc độ hội tụ.

- Thuật toán kết hợp 2 ý tưởng chính: động lượng (momentum) và RMSProp.

- Adam duy trì trung bình trượt của gradient, tương ứng với giá trị trung bình và phương sai của chúng.

- Phần trung bình trượt gradient giúp Adam duy trì hướng di chuyển ngay cả khi gradient thay đổi nhỏ dần. Phần phương sai giúp Adam thích ứng tốc độ học riêng biệt cho từng tham số.

- Ngoài ra, Adam còn áp dụng kỹ thuật chỉnh sửa sai số ban đầu của trung bình trượt để cải thiện hiệu năng. Adam phổ biến nhờ khả năng hội tụ nhanh, xử lý tốt gradient thưa thớt & nhiễu, đồng thời không đòi hỏi nhiều siêu tham số như một số thuật toán khác.

- Hiệu quả trong nhiều tình huống và thường được sử dụng mặc định.

**5. RMSprop (Root Mean Square Propagation):**

- RMSProp là một thuật toán tối ưu hóa được sử dụng trong học máy và học sâu để tối ưu hóa quá trình huấn luyện mạng nơ-ron.

- Không giống Adagrad tích lũy tất cả gradient, RMSProp tính trung bình di động của bình phương các gradient. Điều này cho phép điều chỉnh tốc độ học một cách mượt mà hơn, và ngăn chặn tốc độ học giảm quá đột ngột.

- RMSProp cũng sử dụng một yếu tố suy giảm để điều tiết ảnh hưởng của gradient quá khứ, cho phép gán trọng số lớn hơn cho gradient gần đây.

- Một ưu điểm của RMSProp là khả năng xử lý tốt hàm mục tiêu thay đổi theo thời gian. Trong khi đó, trong trường hợp này Adagrad có thể hội tụ nhanh quá mức. RMSProp có thể điều chỉnh tốc độ học phù hợp với sự thay đổi của hàm mục tiêu.

- Thích hợp cho các bài toán với các đặc trưng không đồng nhất.

**6. Adagrad (Adaptive Gradient Algorithm):**

- Adagrad (Adaptive Gradient) là một thuật toán tối ưu hoá được sử dụng để tối ưu hoá quá trình huấn luyện của các mạng nơ-ron.

- Thuật toán Adagrad điều chỉnh tốc độ học của mỗi tham số của mạng nơ-ron một cách thích ứng trong quá trình huấn luyện. Cụ thể, nó tỉ lệ tốc độ học của mỗi tham số dựa trên

các gradient lịch sử được tính toán cho tham số đó. Các tham số có gradient lớn được cho tốc độ học nhỏ hơn, trong khi những tham số có gradient nhỏ được cho tốc độ học lớn hơn.

- Cập nhật learning rate cho từng trọng số dựa trên lịch sử gradient.

- Hiệu quả cho các bài toán với các đặc trưng thưa (sparse features).

- Adagrad cần lưu trữ lịch sử gradient của tất cả các tham số, có thể tốn kém về bộ nhớ và tính toán.

- Sau khi hội tụ, Adagrad có thể tiếp tục giảm tốc độ học, dẫn đến việc huấn luyện bị chậm lại.

**7. Adadelta:**

- Cải tiến của Adagrad.

- Giới hạn lịch sử gradient để chỉ xem xét lượng gradient gần đây.

- Không cần thiết đặt learning rate ban đầu.

**8. Adamax:**

Phương pháp tương tự như Adam, nhưng thay vì sử dụng L2 norm, nó sử dụng L-infinity norm.

**9. Nadam (Nesterov-accelerated Adaptive Moment Estimation):**

- Kết hợp các ý tưởng từ Nesterov Momentum và Adam.

- Cải thiện tốc độ hội tụ và khả năng thoát khỏi các điểm tối ưu cục bộ.

**10. RAdam (Rectified Adam):**

- Cải tiến của Adam.

- Điều chỉnh biên độ của learning rate để giảm hiện tượng overshooting

Các phương pháp Optimizer có thể hoạt động tốt trong các tình huống khác nhau và tùy thuộc vào dữ liệu và mô hình. Thường thì Adam là phương pháp được sử dụng nhiều nhất và là một lựa chọn tốt cho hầu hết các bài toán.

** So sánh các Optimizer:*

| Optimizer | Ưu điểm | Nhược điểm |
|---|---|---|
| **SGD** | - Dễ dàng thực hiện và tính toán hiệu quả.<br>- Hiệu quả đối với các tập dữ liệu lớn với không gian đặc trưng nhiều chiều<br>- Độ nhạy cao với tốc độ học ban đầu | - SGD có thể bị mắc kẹt trong các cực tiểu cục bộ. |
| **Adam** | - Hiệu quả và dễ thực hiện.<br>- Áp dụng cho các tập dữ liệu lớn và mô hình nhiều chiều.<br>- Khả năng khái quát hóa tốt. | - Cần phải điều chỉnh cẩn thận các hyperparameter. |
| **RMSProp** | - Tốc độ học thích ứng trên mỗi tham số giúp hạn chế sự tích lũy độ dốc.<br>- Hiệu quả đối với các mục tiêu không cố định. | - Có thể có tốc độ hội tụ chậm trong một số trường hợp. |
| **Adagrad** | - Tỷ lệ học tập thích ứng cho mỗi tham số.<br>- Hiệu quả đối với dữ liệu thưa thớt. | - Việc tích lũy gradient bình phương trong mẫu số có thể khiến tốc độ học giảm xuống quá nhanh.<br>- Có thể dừng việc học quá sớm. |

## PHẦN II. TÌM HIỂU VỀ CONTINUAL LEARNING VÀ TEST PRODUCTION KHI XÂY DỰNG MỘT GIẢI PHÁP HỌC MÁY ĐỂ GIẢI QUYẾT MỘT BÀI TOÁN NÀO ĐÓ.

### 1. Continual Learning:

Continual Learning (học liên tục) là một lĩnh vực quan trọng trong học máy, nhằm khắc phục khả năng quên đi thông tin cũ khi hệ thống học máy tiếp tục học thêm thông tin mới. Trong quá trình học liên tục, mô hình học máy phải không chỉ học từ các dữ liệu mới mà còn phải duy trì kiến thức đã học trước đó để áp dụng cho các nhiệm vụ mới. Điều này đặc biệt quan trọng khi dữ liệu mới không có sẵn hoặc giới hạn, và mô hình cần tiếp tục học và cải thiện hiệu suất của mình khi gặp phải nhiệm vụ mới.

Các giải pháp học liên tục thường bao gồm:

- **Regularization**: Sử dụng các phương pháp regularization như L1 hoặc L2 regularization để giảm hiện tượng quên thông tin quan trọng trong quá trình học mới. Regularization giúp giữ cho các trọng số của mô hình ổn định và ngăn chặn quá trình overfitting.

- **Elastic Weight Consolidation (EWC)**: EWC là một kỹ thuật cho phép mô hình học máy giữ lại kiến thức quan trọng từ quá khứ bằng cách đánh giá mức độ quan trọng của các trọng số trong quá trình học. EWC ước lượng mức độ quan trọng của các trọng số bằng cách tính toán các đạo hàm riêng tại các điểm tối ưu của các nhiệm vụ trước đó và sau đó sử dụng các trọng số này để giới hạn sự thay đổi của các trọng số trong quá trình học mới.

**Generative Replay**: Kỹ thuật Generative Replay sử dụng một mô hình sinh dữ liệu để tạo ra các mẫu dữ liệu từ các nhiệm vụ cũ. Dữ liệu được tạo ra từ các nhiệm vụ cũ được sử dụng để huấn luyện mô hình cho các nhiệm vụ mới. Điều này giúp mô hình duy trì kiến thức đã học trước đó và ngăn chặn quên thông tin quan trọng.

Ưu điểm:

- Khả năng khái quát hóa và dự đoán tốt hơn nhờ tích lũy kiến thức theo thời gian.
- Giữ lại và xây dựng dựa trên kiến thức đã học.
- Thích ứng tốt với dữ liệu và kiến thức mới.

Hạn chế:

- Khó quản lý các các phiên bản mô hình khác nhau

- Cần xử lý liên tục dữ liệu mới, dễ bị ảnh hưởng bởi dữ liệu trôi dạt

**2. Test Production:**

Test Production là một khía cạnh quan trọng trong quá trình xây dựng giải pháp học máy. Khi triển khai một hệ thống học máy vào môi trường thực tế, việc kiểm tra và đánh giá hiệu suất của mô hình là cần thiết. Điều này đảm bảo rằng mô hình hoạt động đúng và có khả năng dự đoán chính xác trên dữ liệu mới. Quá trình test production bao gồm các bước sau:

- Chuẩn bị dữ liệu: Dữ liệu mới được thu thập hoặc lựa chọn từ môi trường thực tế để kiểm tra mô hình. Dữ liệu này nên phản ánh các điều kiện và tình huống thực tế mà mô hình sẽ gặp phải. Tập dữ liệu này cần có cùng đặc tính phân phối với dữ liệu huấn luyện và độc lập hoàn toàn với dữ liệu đã dùng để huấn luyện mô hình.

- Chạy kiểm tra: Mô hình được triển khai và chạy trên dữ liệu kiểm tra. Đầu ra của mô hình được đánh giá và so sánh với kết quả mong đợi (nếu có). Cho mô hình dự đoán trên tập dữ liệu kiểm tra và thu thập các metric (accuracy, recall, precision, F1 score,...).

- Đánh giá hiệu suất: Hiệu suất của mô hình được đánh giá bằng các độ đonhư độ chính xác, độ phủ (recall), độ chính xác dương tính (precision), F1-score, và các độ đo khác tùy thuộc vào bài toán cụ thể. Đánh giá hiệu suất giúp xác định khả năng dự đoán của mô hình trên dữ liệu mới và cung cấp thông tin về độ tin cậy của mô hình. So sánh các metric thu được với ngưỡng mong đợi và yêu cầu thực tế để đánh giá hiệu quả của mô hình.

- Tinh chỉnh và cải thiện: Dựa trên kết quả kiểm tra, mô hình có thể được tinh chỉnh và cải thiện để cải thiện hiệu suất trên dữ liệu mới. Điều này có thể bao gồm việc điều chỉnh siêu tham số, thay đổi kiến trúc mô hình, hoặc sử dụng các kỹ thuật khác như kỹ thuật học liên tục để duy trì và mở rộng kiến thức đã học.