

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

----- ∞ 0 ∞ -----



NGUYỄN HOÀNG THANH

**XÂY DỰNG MÔ HÌNH HỆ THỐNG
GỢI Ý SẢN PHẨM CÁ NHÂN HOÁ
TRONG THƯƠNG MẠI ĐIỆN TỬ**

**KHÓA LUẬN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH**

TP. HỒ CHÍ MINH, NĂM 2025

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

----- ∞0∞ -----



NGUYỄN HOÀNG THANH

**XÂY DỰNG MÔ HÌNH HỆ THỐNG
GỢI Ý SẢN PHẨM CÁ NHÂN HOÁ
TRONG THƯƠNG MẠI ĐIỆN TỬ**

Mã số sinh viên: 2151013087

**KHÓA LUẬN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH**

Giảng viên hướng dẫn: TS. LÊ QUANG MINH

TP. HỒ CHÍ MINH, NĂM 2025

**TRƯỜNG ĐẠI HỌC MỞ
THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN**

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc**

GIẤY XÁC NHẬN

Tôi tên là: Nguyễn Hoàng Thanh

Ngày sinh: 17/01/2003

Nơi sinh: TP. Hồ Chí Minh

Chuyên ngành: Khoa Học Máy Tính

Mã học viên: 2151013087

Tôi đồng ý cung cấp toàn văn thông tin khóa luận tốt nghiệp hợp lệ về bản quyền cho Thư viện trường đại học Mở Thành phố Hồ Chí Minh. Thư viện trường đại học Mở Thành phố Hồ Chí Minh sẽ kết nối toàn văn thông tin khóa luận tốt nghiệp vào hệ thống thông tin khoa học của Sở Khoa học và Công nghệ Thành phố Hồ Chí Minh.

Ký tên

Ý kiến cho phép bảo vệ khoá luận tốt nghiệp của giảng viên hướng dẫn

Giảng viên hướng dẫn: TS. Lê Quang Minh

Sinh viên thực hiện: Nguyễn Hoàng Thanh

Ngày sinh: 17/01/2003

Tên đề tài:

Lớp: DH21CS02C

Mã số sinh viên: 2151013087

XÂY DỰNG MÔ HÌNH HỆ THỐNG GỢI Ý SẢN PHẨM CÁ NHÂN HOÁ TRONG THƯƠNG MẠI ĐIỆN TỬ

Ý kiến của giảng viên hướng dẫn về việc cho phép học viên được bảo vệ khóa luận trước Hội đồng:

- Sinh viên Nguyễn Hoàng Thanh hoàn thành khóa luận đúng hạn, có thái độ làm việc nghiêm túc.
- Có thái độ cầu thị, tiếp thu trong hoàn thành bản cuối cùng của khóa luận.
- Hoàn toàn đủ năng lực để tham gia bảo vệ.

Kết luận:

- Sinh viên Nguyễn Hoàng Thanh hoàn toàn đủ khả năng ra trước hội đồng bảo vệ khóa luận.
- Tôi, TS. Lê Quang Minh, đồng ý sinh viên Nguyễn Hoàng Thanh tham gia bảo vệ khóa luận tốt nghiệp.

Thành phố Hồ Chí Minh, ngày 04 tháng 06 năm 2025

Giáo viên hướng dẫn

(Ký tên, ghi rõ họ tên)



Lê Quang Minh

Lời cảm ơn

Trước hết, em xin bày tỏ lòng biết ơn chân thành và sâu sắc đến Tiến sĩ Lê Quang Minh, người thầy đã tận tình hướng dẫn, luôn đồng hành và đưa ra những nhận xét quý báu trong suốt quá trình em thực hiện khóa luận tốt nghiệp. Sự hỗ trợ chu đáo và những chỉ dẫn khoa học của thầy không chỉ giúp em củng cố kiến thức chuyên môn, mà còn hình thành tư duy nghiên cứu logic, hệ thống và hiệu quả.

Em cũng xin gửi lời cảm ơn chân thành đến quý thầy cô trong Khoa Đào tạo Đặc biệt – Trường Đại học Mở Thành phố Hồ Chí Minh. Chính sự giảng dạy tận tâm và những chia sẻ kinh nghiệm thực tiễn của quý thầy cô đã tạo nên nền tảng kiến thức vững chắc, giúp em tự tin triển khai và hoàn thiện đề tài nghiên cứu của mình.

Trong khóa luận này, em đã triển khai và xây dựng đề tài “Hệ thống gợi ý sản phẩm cá nhân hoá trong thương mại điện tử”, với mong muốn ứng dụng trí tuệ nhân tạo vào việc cải thiện trải nghiệm mua sắm của người dùng. Hệ thống được phát triển nhằm cá nhân hóa các đề xuất sản phẩm, dựa trên sở thích và hành vi mua hàng, từ đó hỗ trợ người tiêu dùng tìm kiếm sản phẩm phù hợp một cách nhanh chóng, chính xác và hiệu quả hơn. Em hy vọng rằng kết quả nghiên cứu sẽ là nền tảng góp phần vào việc nâng cao chất lượng dịch vụ cho các nền tảng thương mại điện tử trong tương lai.

Bên cạnh sự chỉ dẫn và hỗ trợ chuyên môn, em cũng xin dành lời cảm ơn chân thành đến gia đình, những người luôn là chỗ dựa tinh thần vững chắc, đã âm thầm ủng hộ và tiếp thêm động lực cho em trong suốt chặng đường học tập. Em cũng không quên cảm ơn các anh chị, bạn bè, và những người thân yêu đã luôn ở bên, khích lệ và hỗ trợ em trong từng bước thực hiện khóa luận này.

Cuối cùng, em rất mong nhận được những ý kiến đóng góp quý báu từ quý thầy cô và các bạn để em có thể tiếp tục hoàn thiện, nâng cao chất lượng nghiên cứu trong những chặng đường sắp tới.

Em xin chân thành cảm ơn!

Thành phố Hồ Chí Minh, ngày 16 tháng 07 năm 2025

Sinh viên

(Ký tên, ghi rõ họ tên)



Nguyễn Hoàng Thanh

Tóm tắt

Luận văn này đề xuất và triển khai một mô hình hệ thống gợi ý lai (Hybrid Recommender System), kết hợp các phương pháp học sâu (Deep Learning) và học máy truyền thống nhằm khai thác hiệu quả cả hai loại phản hồi từ người dùng: phản hồi ngầm (Implicit Feedback) và phản hồi rõ ràng (Explicit Feedback). Hệ thống sử dụng thư viện LightFM để xử lý dữ liệu Implicit Feedback, chẳng hạn như hành vi xem sản phẩm, thêm vào giỏ hàng và mua hàng, đồng thời áp dụng thuật toán SVD từ thư viện Surprise cho dữ liệu Explicit Feedback, chẳng hạn như đánh giá trực tiếp từ người dùng. Các đặc trưng bổ sung như thời gian, giá cả, độ tuổi, giới tính được tích hợp nhằm nâng cao khả năng cá nhân hóa gợi ý sản phẩm.

Đối với dữ liệu Implicit Feedback, thư viện LightFM được huấn luyện để tạo ra embedding cho người dùng và sản phẩm. Các embedding này sau đó được kết hợp với metadata và đưa vào một kiến trúc học sâu bao gồm các tầng embedding, tích chập một chiều (Conv1D), gộp cực đại toàn cục (GlobalMaxPooling1D), và các tầng Dense kết hợp Dropout và Batch Normalization. Nhằm khắc phục tình trạng mất cân bằng lớp, trọng số lớp (class weight) được tính toán và tích hợp vào Focal Loss để cải thiện khả năng học đối với các lớp thiểu số. Kết quả thực nghiệm cho thấy mô hình đạt $AUC = 0.8119$, $Precision = 0.9663$ và $Recall = 0.4212$, phản ánh khả năng dự đoán chính xác nhưng vẫn còn hạn chế trong việc bao phủ toàn bộ hành vi người dùng.

Đối với dữ liệu Explicit Feedback, các embedding được khởi tạo từ SVD và tiếp tục được tinh chỉnh thông qua một kiến trúc học sâu tương tự. Kết quả thu được là $RMSE = 1.2914$ và $MAE = 1.0176$, cho thấy mô hình có hiệu quả trong việc dự đoán đánh giá của người dùng.

Hệ thống cũng được kiểm tra trong các kịch bản cold-start (người dùng hoặc sản phẩm mới). Kết quả cho thấy hiệu suất ở nhóm non-cold-start cao hơn đáng kể, trong khi nhóm cold-start vẫn còn hạn chế, cho thấy cần cải thiện bằng cách tận dụng metadata hiệu quả hơn hoặc thiết kế mô hình phù hợp hơn cho các trường hợp khởi đầu.

Nhìn chung, hệ thống gợi ý lai được đề xuất cho thấy hiệu quả trong việc xử lý cả phản hồi ngầm và rõ ràng từ người dùng, với kết quả thực nghiệm khả quan trên hai loại dữ liệu. Tuy hiệu suất ở tình huống cold-start còn hạn chế, nhưng việc tận dụng metadata và cải tiến mô hình hứa hẹn là hướng phát triển tiềm năng trong tương lai.

Abstract

This thesis proposes and implements a hybrid recommender system that combines deep learning methods with traditional machine learning techniques to effectively leverage both types of user feedback: implicit feedback and explicit feedback. The system uses the LightFM library to process implicit feedback data, such as product views, add-to-cart actions, and purchases, while the SVD algorithm from the Surprise library is applied to explicit feedback data, such as user ratings. Additional features such as time, price, age, and gender are incorporated to enhance the system’s ability to personalize recommendations.

For implicit feedback data, the LightFM library is trained to generate embeddings for users and products. These embeddings are then combined with metadata and fed into a deep learning architecture that includes embedding layers, one-dimensional convolutional layers (Conv1D), global max pooling layers (GlobalMaxPooling1D), and dense layers with Dropout and Batch Normalization. To address class imbalance, class weights are computed and integrated into the Focal Loss function to improve the model’s ability to learn from minority classes. Experimental results show that the model achieved an AUC of 0.8119, Precision of 0.9663, and Recall of 0.4212, reflecting high predictive accuracy but limited coverage of all user behaviors.

For explicit feedback data, embeddings are initialized using SVD and further fine-tuned through a similar deep learning architecture. The results, with RMSE = 1.2914 and MAE = 1.0176, indicate that the model performs well in predicting user ratings.

The system was also evaluated in cold-start scenarios (new users or new products). The results show that performance is significantly better in the non-cold-start group, while the cold-start group remains limited, highlighting the need for improvement through more effective use of metadata or designing models better suited to initial-state conditions.

Overall, the proposed hybrid recommender system demonstrates effectiveness in handling both implicit and explicit user feedback, with promising experimental results across both data types. While cold-start performance is still limited, leveraging metadata and enhancing model design presents a potential direction for future development.

Mục lục

Ý kiến Giảng viên hướng dẫn	i
Lời cảm ơn	ii
Tóm tắt	iii
Abstract	iv
Mục lục	v
Danh sách từ viết tắt	x
Danh sách bảng	x
Danh sách hình	xi
1 Tổng quan	1
1.1 Giới thiệu đề tài	1
1.2 Lý do chọn đề tài	1
1.3 Mục tiêu nghiên cứu	1
1.4 Phương pháp nghiên cứu	2
1.5 Ý nghĩa thực tiễn	2
2 Cơ sở lý thuyết	3
2.1 Hệ thống gợi ý	3
2.1.1 Khái niệm hệ thống gợi ý	3
2.1.2 Vai trò của hệ thống gợi ý trong thương mại điện tử	3
2.1.3 Các phương pháp xây dựng hệ thống gợi ý	3
2.2 Vấn đề trong hệ thống gợi ý	7
2.2.1 Bài toán dữ liệu thưa (Sparsity Data	7
2.2.2 Bài toán tương tác mới giữa người dùng và sản phẩm (cold-start)	8
2.3 Biểu diễn dữ liệu trong hệ thống gợi ý	9
2.3.1 Phân loại dữ liệu trong hệ thống gợi ý	9
2.3.2 Ma trận tương tác người dùng và sản phẩm (Interaction Matrix)	10
2.3.3 Đặc trưng người dùng và sản phẩm	12
2.3.4 Encoding dữ liệu	14
2.4 Tổng quan về học máy và học sâu trong gợi ý	15
2.4.1 Học máy trong hệ thống gợi ý	15
2.4.2 Học sâu trong hệ thống gợi ý	16
2.5 Giới thiệu mô hình LightFM	18
2.5.1 Kiến trúc và nguyên lý hoạt động	18

2.5.2	Ưu điểm và nhược điểm của LightFM so với Surprise	19
2.5.3	Khi nào sử dụng LightFM và Surprise SVD	20
2.6	Tổng kết chương	20
3	Phân tích và thiết kế mô hình	21
3.1	Kiến trúc tổng thể của hệ thống	21
3.2	Tổng quan Pipeline xử lý dữ liệu trong mô hình gợi ý	22
3.3	Mô hình dữ liệu	23
3.4	Thiết kế mô hình đề xuất	24
3.4.1	Huấn luyện LightFM và trích xuất embedding	25
3.4.2	Quy trình xây dựng và huấn luyện mô hình gợi ý DNN	26
3.4.3	Kiến trúc mô hình của DNN	28
3.4.4	Hàm mất mát và trọng số lớp	31
3.5	Tổng kết chương	31
4	Cài đặt và thực nghiệm	32
4.1	Môi trường cài đặt	32
4.2	Tập dữ liệu	32
4.2.1	Nguồn gốc dữ liệu	32
4.2.2	Kích Thước Tập Dữ Liệu	32
4.2.3	Đặc Trưng Sử Dụng	32
4.2.4	Tiền Xử Lý Dữ Liệu	33
4.2.5	Mất cân bằng dữ liệu	35
4.3	Chi tiết cài đặt mô hình	35
4.3.1	Cài đặt mô hình LightFM kết hợp DNN	35
4.4	Các chỉ số đánh giá	37
4.5	Kết quả thực nghiệm	38
4.5.1	Kết quả trước khi fine-tuning	38
4.5.2	Kết quả sau khi fine-tuning embedding	38
4.5.3	Đánh giá vấn đề cold-start	40
4.5.4	Kết quả đạt được	41
4.5.5	So sánh hiệu quả giữa mô hình LightFM Hybrid và mô hình deep learning đề xuất (Embeddings LightFM + DNN)	42
4.6	Kết quả và đánh giá mô hình gợi ý sản phẩm đối với dữ liệu tường minh (Explicit Feedback)	43
4.6.1	Đánh giá theo nhóm dữ liệu Cold-start và Non-Cold-start	44
4.7	Triển khai	45
4.8	Tổng kết chương	48
5	Đánh giá và Kết luận	49
5.1	Tóm tắt kết quả chính	49
5.2	Đánh giá ưu và nhược điểm	49
5.3	Hướng phát triển tiếp theo	49
5.4	Kết luận	50

Danh sách từ viết tắt

SVD	Singular Value Decomposition–Phân rã ma trận, dùng trong hệ thống gợi ý đánh giá người dùng.
RMSE	Root Mean Square Error–Đánh giá sai số bình phương trung bình.
MAE	Mean Absolute Error–Đo sai số tuyệt đối trung bình trong dự đoán.
AUC	Area Under the Curve–Đo hiệu suất phân loại nhị phân qua diện tích dưới đường cong ROC.
TF-IDF	Term Frequency - Inverse Document Frequency–Được dùng để biểu diễn văn bản dựa trên tần suất và độ đặc trưng của từ.
CF	Collaborative Filtering–Lọc cộng tác dựa vào hành vi người dùng.
CBF	Content-Based Filtering–Lọc dựa vào nội dung đặc trưng của sản phẩm.
FM	Factorization Machine–Mô hình nhân tố hóa mở rộng cho sparse feature.
DNN	Deep Neural Network–Mạng nơ-ron sâu với nhiều lớp ẩn.
MLP	Multilayer Perceptron–Kiến trúc mạng nơ-ron gồm nhiều tầng kết nối đầy đủ.
NCF	Neural Collaborative Filtering–Kết hợp mạng nơ-ron vào lọc cộng tác.
DeepFM	Deep Factorization Machine–Mô hình kết hợp giữa FM và mạng sâu để khai thác tương tác đặc trưng.
DIN	Deep Interest Network–Mô hình mạng sâu quan tâm đến hành vi người dùng theo ngữ cảnh.
CNN	Convolutional Neural Network–Mạng tích chập thường dùng trong xử lý ảnh hoặc âm thanh.
ML	Machine Learning–Một nhánh của trí tuệ nhân tạo, giúp máy học từ dữ liệu.
ALS	Alternating Least Squares–Thuật toán dùng trong matrix factorization.
KNN	K-Nearest Neighbors–Thuật toán học máy dựa trên khoảng cách gần nhất.
SVM	Support Vector Machine–Máy vector hỗ trợ - mô hình phân loại tuyến tính/phân lớp phi tuyến.
ANN	Artificial Neural Network–Mạng nơ-ron nhân tạo.
ReLU	Rectified Linear Unit–Hàm kích hoạt thường dùng trong mạng sâu.
COO	Coordinate Format–Định dạng ma trận thưa theo tọa độ.
CSR	Compressed Sparse Row–Định dạng ma trận thưa theo hàng.

WARP	Weighted Approximate-Rank Pairwise–Hàm mất mát dùng trong hệ thống gợi ý thứ hạng.
API	Application Programming Interface–Giao diện lập trình ứng dụng.
SMOTE	Synthetic Minority Over-sampling Technique–Phương pháp tạo dữ liệu tổng hợp cho lớp thiểu số.
ConvFM	Convolutional Factorization Machine–Mô hình kết hợp giữa CNN và FM để xử lý dữ liệu dạng lưới.
GAN	Generative Adversarial Network–Mạng đối kháng sinh dữ liệu mới.
NAS	Neural Architecture Search–Tự động tìm kiến trúc mạng tốt nhất.

Danh sách bảng

2.1	Mô tả vector đặc trưng TF-IDF của các sản phẩm	4
2.2	Mô tả ma trận đánh giá của người dùng đối với các sản phẩm	8
2.3	Ví dụ về dữ liệu với Cold-Start Người Dùng và Sản Phẩm	8
2.4	So sánh giữa phản hồi tường minh và phản hồi ngầm định	10
2.5	Ma trận tương tác giữa người dùng và sản phẩm với dữ liệu implicit feedback	12
2.6	Ma trận tương tác giữa người dùng và sản phẩm với dữ liệu explicit feedback	12
2.7	Ví dụ One-hot encoding cho đặc trưng "Loại sản phẩm"	14
2.8	So sánh hai phương pháp mã hóa	15
4.1	So sánh hiệu năng giữa các nhóm cold-start và non-cold	40
4.2	So sánh hiệu năng mô hình đối với dữ liệu tường minh Amazon	44
4.3	Kết quả đánh giá theo từng nhóm dữ liệu	44

Danh sách hình

2.1	Mình họa trực quan quá trình phân rã ma trận tương tác bằng SVD	6
2.2	Tổng quan về ConvFM	6
2.3	Tổng quan về Hybrid recommendation dựa trên kết hợp CBF và CF	7
2.4	Ví dụ phân biệt giữa Implicit và Explicit Feedback	9
2.5	Ví dụ về ma trận dữ liệu implicit và explicit feedback	10
2.6	Tổng quan mô hình lai của LightFM	18
3.1	Tổng quan mô hình gợi ý sản phẩm	21
3.2	Chi tiết pipeline dữ liệu của mô hình gợi ý sản phẩm	22
3.3	Mình họa ánh xạ Metadata sang Vector Nhúng	24
3.4	Sơ đồ luồng minh họa pipeline xử lý dữ liệu của LightFM	25
3.5	Quy trình huấn luyện LightFM và trích xuất embedding	26
3.6	Ma trận tương tác và trích xuất embeddings	26
3.7	Tổng quan kiến trúc mô hình học sâu sử dụng embeddings	27
3.8	Khái quát xây dựng mô hình DDN Recommender	28
3.9	Kiến trúc các tầng trong DNN	29
4.1	Dữ liệu đã được tiền xử lý	34
4.2	So sánh phân phối dữ liệu Implicit (trái) và Explicit (phải)	35
4.3	Quá trình huấn luyện của mô hình DNN Hybrid	37
4.4	Confusion matrix trước khi fine-tuning	38
4.5	Confusion matrix sau khi fine-tuning	39
4.6	So sánh F1-score theo class giữa các nhóm User/Item Cold-start và Non-cold	41
4.7	Kết quả gợi ý của mô hình lai kết hợp Light và DNN	42
4.8	So sánh Precision, Recall và AUC giữa mô hình LightFM hybrid và mô hình NN hybrid	43
4.9	So sánh sai số RMSE và MAE giữa các nhóm dữ liệu	44
4.10	Kết quả gợi ý sản phẩm cho người dùng trong hệ thống gợi ý explicit	45
4.11	Triển khai hệ thống gợi ý DNN Hybrid	46
4.12	Chức năng thêm mới tương tác người dùng và sản phẩm	47
4.13	Chức năng gợi ý sản phẩm cho người dùng	48

Chương 1. Tổng quan

1.1 Giới thiệu đề tài

Trong bối cảnh cuộc cách mạng công nghệ số đang diễn ra mạnh mẽ, đặc biệt là sự bùng nổ của lĩnh vực thương mại điện tử, nhu cầu cá nhân hóa trải nghiệm người dùng trở thành yếu tố then chốt để các doanh nghiệp cạnh tranh. Việc hiểu rõ nhu cầu, hành vi và sở thích cá nhân không chỉ giúp nâng cao sự hài lòng của khách hàng mà còn góp phần gia tăng tỉ lệ và duy trì lòng trung thành của người tiêu dùng. Một trong những công nghệ đóng vai trò quan trọng trong chiến lược cá nhân hóa là hệ thống gợi ý sản phẩm. Các hệ thống này có khả năng phân tích và học hỏi từ dữ liệu tương tác lịch sử giữa người dùng và sản phẩm, từ đó đưa ra các đề xuất phù hợp và mang tính cá nhân cao.

Tuy nhiên, các phương pháp truyền thống trong hệ thống gợi ý, đặc biệt là kỹ thuật lọc cộng tác (Collaborative Filtering) còn tồn tại nhiều hạn chế trong thực tiễn triển khai. Một trong những vấn đề phổ biến là hiện tượng dữ liệu thưa (sparsity) và vấn đề tương tác giữa người dùng và sản phẩm mới (cold-start). Điều này làm giảm độ chính xác và tính cá nhân hóa của gợi ý, ảnh hưởng trực tiếp đến hiệu quả hoạt động của hệ thống.

Trước những thách thức trên, đề tài khóa luận này tập trung nghiên cứu và xây dựng một mô hình gợi ý sản phẩm kết hợp học sâu và mô hình lai (Hybrid Deep Learning-based Recommender System). Mục đích cho sự kết hợp này là tận dụng đồng thời sức mạnh của biểu diễn ngữ nghĩa từ các mô hình học sâu và khả năng tổng hợp thông tin từ nhiều nguồn dữ liệu của mô hình lai, từ đó nâng cao chất lượng gợi ý, đặc biệt trong các tình huống thiếu dữ liệu hoặc cold-start. Mô hình đề xuất hướng tới kết quả sẽ là một giải pháp hiệu quả và khả thi cho các nền tảng thương mại điện tử hiện đại đang hướng đến tối ưu hóa trải nghiệm người dùng một cách toàn diện.

1.2 Lý do chọn đề tài

Về mặt xã hội, sự phát triển mạnh mẽ của thương mại điện tử yêu cầu các doanh nghiệp phải nâng cao trải nghiệm người dùng thông qua việc cá nhân hóa sản phẩm. Hệ thống gợi ý sản phẩm giúp người tiêu dùng tìm kiếm sản phẩm phù hợp với sở thích, từ đó tăng sự hài lòng và trung thành, góp phần phát triển môi trường thương mại điện tử bền vững.

Về mặt thực tiễn, bài toán cold-start là một trong những thách thức lớn khi xây dựng mô hình hệ thống gợi ý, đặc biệt trong trường hợp người dùng hoặc sản phẩm mới chưa có đủ dữ liệu tương tác. Việc kết hợp các phương pháp học sâu với mô hình lai, chẳng hạn như sử dụng Singular Value Decomposition (SVD) hoặc LightFM cùng mạng nơ-ron sâu, cho phép mô hình học được các đặc trưng tiềm ẩn một cách hiệu quả hơn, từ đó cải thiện độ chính xác của các gợi ý cá nhân hóa.

1.3 Mục tiêu nghiên cứu

Mục tiêu trọng tâm của đề tài là nghiên cứu và xây dựng mô hình gợi ý sản phẩm cá nhân hóa, hoạt động hiệu quả trong điều kiện dữ liệu không đầy đủ, nhằm giải quyết các thách thức như bài toán cold-start và dữ liệu thưa. Đề tài tập trung phát triển một mô hình lai kết hợp giữa các thuật toán như LightFM và SVD với mạng nơ-ron sâu, nhằm tận dụng ưu điểm của cả hai phương pháp trong việc học từ dữ liệu tương tác ngầm định và tương tác tường minh của người dùng và sản phẩm.

1.4 Phương pháp nghiên cứu

Đề tài tập trung vào việc xây dựng một hệ thống gợi ý sản phẩm cá nhân hóa, kết hợp giữa các thuật toán LightFM, SVD và mạng nơ-ron sâu. Mô hình này được thiết kế để xử lý dữ liệu từ cả dữ liệu tương tác ngầm định và dữ liệu tương tác tường minh. Sử dụng LightFM giúp khai thác các embedding người dùng và sản phẩm, trong khi SVD từ thư viện Surprise sẽ đóng vai trò quan trọng trong việc phân rã ma trận đánh giá sản phẩm của người dùng. Việc kết hợp mạng nơ-ron sâu giúp mô hình học được các mối quan hệ tiềm ẩn giữa người dùng và sản phẩm, từ đó cải thiện chất lượng gợi ý và khả năng dự đoán nhu cầu người dùng.

Bên cạnh đó, mô hình được mở rộng để khai thác metadata, tức là các thông tin phụ trợ về người dùng và sản phẩm. Các thông tin này có thể bao gồm độ tuổi, giới tính, hành vi tiêu dùng của người dùng, và mô tả, danh mục, giá cả của sản phẩm. Việc tích hợp metadata vào mô hình học sâu giúp hệ thống có thể đưa ra các gợi ý chính xác hơn, ngay cả trong các tình huống thiếu dữ liệu lịch sử hoặc khi người dùng/sản phẩm mới xuất hiện. Mô hình sẽ được huấn luyện và đánh giá trên các bộ dữ liệu lớn từ các nền tảng thương mại điện tử thực tế, sử dụng các chỉ số như Area Under the Curve (AUC), Precision, Recall, Mean Absolute Error (MAE) và Root Mean Square Error (RMSE) để kiểm tra hiệu quả và độ chính xác của mô hình gợi ý.

1.5 Ý nghĩa thực tiễn

Đề tài có ý nghĩa thực tiễn trong bối cảnh thương mại điện tử ngày càng cạnh tranh. Việc phát triển hệ thống gợi ý cá nhân hóa chính xác giúp doanh nghiệp nâng cao trải nghiệm người dùng, giữ chân khách hàng và thúc đẩy doanh thu. Đặc biệt, khả năng xử lý hiệu quả bài toán cold-start giúp mô hình hoạt động tốt ngay cả khi thiếu dữ liệu lịch sử về người dùng hoặc sản phẩm mới.

Đồng thời, đề tài này cũng sẽ là một đóng góp vào nghiên cứu học thuật trong lĩnh vực trí tuệ nhân tạo, đặc biệt là trong việc ứng dụng học sâu và thuật toán lai vào hệ thống gợi ý. Nghiên cứu này không chỉ giúp mở rộng kiến thức về các phương pháp cá nhân hóa mà còn tạo cơ sở cho các công trình tiếp theo liên quan đến khai thác dữ liệu lớn và phát triển các hệ thống thông minh trong kinh doanh số.

Chương 2. Cơ sở lý thuyết

2.1 Hệ thống gợi ý

2.1.1 Khái niệm hệ thống gợi ý

Hệ thống gợi ý là công nghệ sử dụng các thuật toán để phân tích hành vi và sở thích của người dùng, từ đó đưa ra các đề xuất phù hợp nhằm cá nhân hóa trải nghiệm. Hệ thống này giúp người dùng nhanh chóng tìm thấy nội dung hoặc sản phẩm phù hợp giữa một khối lượng thông tin khổng lồ.

Trong bối cảnh dữ liệu và nội dung số ngày càng bùng nổ, hệ thống gợi ý đóng vai trò quan trọng trong việc giảm thiểu tình trạng quá tải thông tin và nâng cao trải nghiệm người dùng [1]. Nhiều nền tảng lớn như Amazon, Netflix và YouTube đã ứng dụng mạnh mẽ các kỹ thuật gợi ý để nâng cao sự hài lòng của người dùng [2].

2.1.2 Vai trò của hệ thống gợi ý trong thương mại điện tử

Trong một trang thương mại điện tử, người dùng sẽ được tiếp cận với vô vàn sản phẩm có sự đa dạng về giá cả, danh mục, số lượng và chất lượng. Việc tìm kiếm những sản phẩm thực sự phù hợp với nhu cầu cá nhân là một nhiệm vụ khó khăn và tốn nhiều thời gian.

Theo một khảo sát gần đây của Accenture, 74% người tiêu dùng cho biết họ đã từ bỏ giỏ hàng trực tuyến ít nhất một lần trong ba tháng gần nhất vì cảm thấy “bị bội thực thông tin, quá nhiều lựa chọn và một mỗi khi phải đưa ra quyết định” [3]

Do đó, một hệ thống đề xuất phù hợp sẽ rất cần thiết cho khách hàng và doanh nghiệp của một website thương mại điện tử. Để tăng hiệu suất của hệ thống thương mại điện tử, hệ thống đề xuất được sử dụng, hệ thống này phụ thuộc vào hầu hết các hệ thống hiện có chỉ dựa trên thông tin mua hàng. Hệ thống đề xuất thu thập thông tin từ khách hàng và đề xuất các mặt hàng mà họ thấy có giá trị nhất trong số các sản phẩm hiện có [4].

2.1.3 Các phương pháp xây dựng hệ thống gợi ý

Trong hệ thống gợi ý, ba phương pháp chính thường được áp dụng để đưa ra đề xuất phù hợp cho người dùng:

- **Gợi ý dựa trên nội dung (Content-based Filtering):** hệ thống đề xuất các sản phẩm có đặc điểm tương tự với những sản phẩm mà người dùng đã tương tác trước đó. Phương pháp này tập trung vào việc phân tích thuộc tính của sản phẩm và sở thích cá nhân. Trong Content-Based Filtering (CBF) dữ liệu huấn luyện trong hệ thống bao gồm các mục mà người dùng đã quan tâm trước đó, được biểu diễn dưới dạng vector với các thuộc tính có thể là nhị phân, định danh hoặc số [5]. Phương pháp vector hoá Term Frequency - Inverse Document Frequency (TF-IDF) được tính bằng công thức sau:

$$TF-IDF(t, d) = TF(t, d) \times IDF(t) \quad (2.1)$$

Với t là từ trong tài liệu, d là tài liệu cụ thể, $TF(t, d)$ là tần suất xuất hiện của từ t trong tài liệu d , $IDF(t)$ là tần suất đảo ngược của tài liệu chứa từ t và nó được tính bởi công thức:

$$IDF(t) = \log \left(\frac{N}{|\{d \in D : t \in d\}|} \right) \quad (2.2)$$

Trong đó, N là tổng số tài liệu, D là tập hợp tất cả các tài liệu, $|\{d \in D : t \in d\}|$ là số lượng tài liệu chứa từ t . Từ biểu diễn vector TF-IDF (2.1), hệ thống có thể tính toán mức độ tương đồng giữa các sản phẩm, từ đó đề xuất những mục phù hợp với sở thích của người dùng.

Trong bảng một người dùng đã từng quan tâm đến Sản phẩm 1, có mô tả: "Smartphone 5G, màn hình OLED, camera kép, pin 5000mAh". Sau quá trình tiền xử lý và vector hóa, nội dung của ba sản phẩm được biểu diễn như sau:

Bảng 2.1: Mô tả vector đặc trưng TF-IDF của các sản phẩm

Thuộc tính	Sản phẩm 1	Sản phẩm 2	Sản phẩm 3
smartphone	0.30	0.25	0.00
oled	0.40	0.00	0.00
camera	0.35	0.10	0.00
pin	0.20	0.00	0.00
laptop	0.00	0.00	0.50
màn hình	0.15	0.10	0.10

Dựa trên các vector TF-IDF trong Bảng 2.1, hệ thống tính độ tương đồng cosine giữa Sản phẩm 1 và các sản phẩm khác. Kết quả cho thấy Sản phẩm 2 có nhiều đặc trưng giống với Sản phẩm 1 như "smartphone", "màn hình" và "camera", trong khi Sản phẩm 3 là laptop nên ít tương đồng. Do đó, hệ thống đề xuất Sản phẩm 2 cho người dùng A.

– **Lọc cộng tác (Collaborative Filtering):** hoạt động dựa trên giả định rằng những người dùng có hành vi hoặc sở thích tương tự trong quá khứ sẽ tiếp tục có lựa chọn tương tự trong tương lai. Nhiệm vụ là đề xuất các mục (items) cho một người dùng đang hoạt động (active user) dựa trên sở thích của họ và sở thích của những người dùng khác. Trong hầu hết các trường hợp, sở thích được thể hiện dưới dạng các điểm đánh giá số học (numerical evaluation scores) [6]. Collaborative Filtering (CF) có 2 loại chính:

+ Lọc cộng tác theo người dùng (User-based CF): tính toán độ tương đồng giữa người dùng dựa trên lịch sử đánh giá. Khi một người dùng mới cần được gợi ý, hệ thống sẽ tìm những người dùng tương tự nhất và gợi ý các sản phẩm mà họ đã đánh giá cao. Hai cách phổ biến để tính độ tương đồng là:

- Hệ số tương quan Pearson:

$$\text{sim}_{\text{pearson}}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2} \cdot \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}} \quad (2.3)$$

Trong đó, I_{uv} là tập các sản phẩm mà cả người dùng u và v đều đã đánh giá; r_{ui} là đánh giá của người dùng u với sản phẩm i ; \bar{r}_u là điểm đánh giá trung bình của người dùng u .

- Độ tương đồng cosine:

$$\text{sim}_{\text{cosine}}(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}} \quad (2.4)$$

Với các ký hiệu tương tự như (2.3).

- + Lọc cộng tác theo sản phẩm (Item-based CF): phương pháp này tính toán sự tương đồng giữa các sản phẩm sử dụng độ tương đồng cosine. Sự tương đồng giữa hai sản phẩm i và j , ký hiệu là $\text{sim}(i, j)$, được tính như sau:

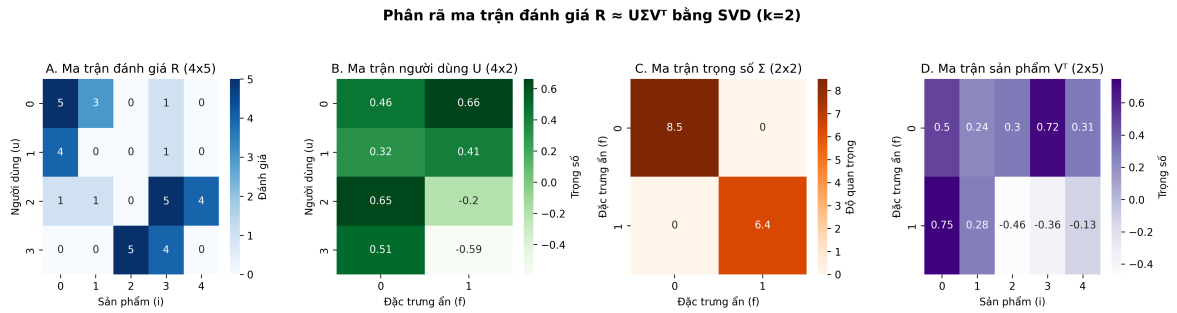
$$\text{sim}(i, j) = \frac{\sum_{u \in U} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U} r_{ui}^2} \cdot \sqrt{\sum_{u \in U} r_{uj}^2}} \quad (2.5)$$

Với U là tập hợp tất cả người dùng, r_{ui} là điểm đánh giá của người dùng u cho sản phẩm i , r_{uj} là điểm đánh giá của người dùng u cho sản phẩm j . Công thức (2.5) sử dụng độ tương đồng cosine để xác định các sản phẩm tương tự dựa trên đánh giá của người dùng. Cách tiếp cận này thường ổn định hơn so với User-based CF (2.4) khi dữ liệu người dùng thay đổi thường xuyên.

Ngoài các phương pháp dựa trên tương đồng, lọc cộng tác còn có thể được mở rộng bằng cách sử dụng các kỹ thuật phân rã ma trận như SVD, thuộc nhóm lọc cộng tác dựa trên mô hình (Model-based CF). Phương pháp này xấp xỉ ma trận tương tác R dưới dạng:

$$R \approx U \Sigma V^T \quad (2.6)$$

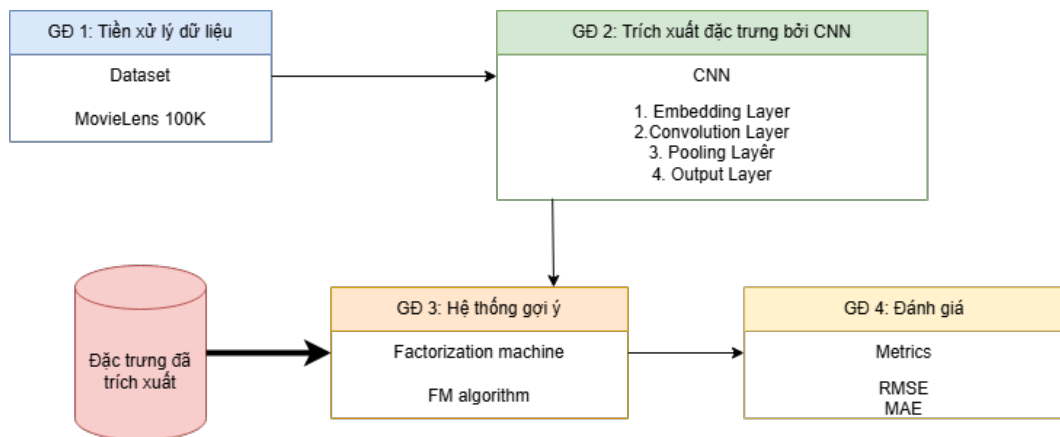
Trong đó, R là ma trận tương tác giữa người dùng và sản phẩm, biểu thị sự liên kết giữa người dùng và các sản phẩm mà họ đã đánh giá. U là ma trận đặc trưng người dùng, đại diện cho các yếu tố hoặc đặc điểm tiềm ẩn của người dùng. Σ là ma trận đường chéo chứa các giá trị kỳ dị, dùng để điều chỉnh độ quan trọng của các thành phần trong phân tích ma trận. Cuối cùng, V^T là ma trận đặc trưng sản phẩm, là chuyển vị của ma trận đặc trưng sản phẩm V , thể hiện các đặc điểm tiềm ẩn của các sản phẩm. Phân rã ma trận bằng SVD (2.6) giúp rút trích các đặc trưng tiềm ẩn từ dữ liệu tương tác, từ đó cải thiện hiệu quả của phương pháp lọc cộng tác dựa trên người dùng. Thông qua việc giảm chiều, SVD giúp mô hình hóa mối quan hệ ngầm giữa người dùng và sản phẩm, từ đó dự đoán các đánh giá còn thiếu với độ chính xác cao hơn. Hình 2.1 minh họa trực quan quá trình này với ma trận $R \in \mathbb{R}^{4 \times 5}$ được phân rã thành $U \in \mathbb{R}^{4 \times 2}$, $\Sigma \in \mathbb{R}^{2 \times 2}$, và $V^T \in \mathbb{R}^{2 \times 5}$.



Hình 2.1: Minh họa trực quan quá trình phân rã ma trận tương tác bằng SVD

- Hệ thống gợi ý lai (Hybrid Recommendation Systems): phương pháp kết hợp nhiều kỹ thuật gợi ý khác nhau, bao gồm lọc cộng tác, lọc nội dung và các mô hình học sâu, nhằm tận dụng ưu điểm của từng phương pháp để cải thiện độ chính xác và giải quyết các vấn đề như cold start và data sparsity.

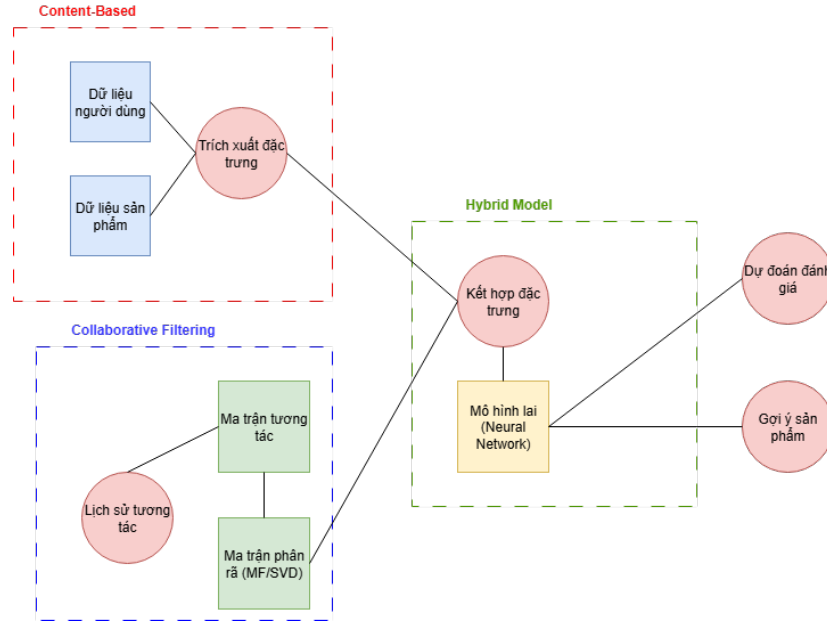
Trong nghiên cứu của Alrashidi và các cộng sự [7], tác giả đề xuất mô hình Convolutional Factorization Machine (ConvFM), một mô hình lai kết hợp giữa Factorization Machines và Convolutional Neural Networks.



Hình 2.2: Tổng quan về ConvFM

Mô hình 2.2 không chỉ học các đặc trưng tiềm ẩn từ người dùng và sản phẩm mà còn tận dụng khả năng của Convolutional Neural Network (CNN) trong việc trích xuất đặc trưng chiều sâu, giúp cải thiện hiệu quả gợi ý và xử lý các dữ liệu phức tạp, đồng thời giải quyết vấn đề cold start.

Trong cấu trúc Hình 2.3 là mô hình lai kết hợp cả hai phương pháp lọc cộng tác và lọc theo nội dung giúp tận dụng tối đa đặc trưng của dữ liệu sản phẩm và tương tác của người dùng, khắc phục được hạn chế cold start của lọc theo nội dung và cải thiện độ chính xác của lọc cộng tác.



Hình 2.3: Tổng quan về Hybrid recommendation dựa trên kết hợp CBF và CF

Trong bài khóa luận này, hệ thống gợi ý lai kết hợp LightFM và Deep Neural Network (DNN) để nâng cao hiệu quả gợi ý sản phẩm. LightFM trích xuất các đặc trưng embedding giữa người dùng và sản phẩm thông qua các tương tác như xem, thêm vào giỏ hàng và giao dịch. CNN học các metadata của sản phẩm và người dùng để cung cấp thông tin bổ sung. CNN có khả năng khai thác các thông tin tiềm ẩn từ dạng dữ liệu phi cấu trúc mà LightFM khó xử lý hiệu quả nếu đứng độc lập. Sự kết hợp này giúp hệ thống khai thác cả tương tác người dùng và thuộc tính sản phẩm, từ đó đưa ra gợi ý chính xác hơn cho từng người dùng.

2.2 Vấn đề trong hệ thống gợi ý

2.2.1 Bài toán dữ liệu thưa (Sparsity Data)

Dữ liệu thưa là một thách thức lớn trong các hệ thống gợi ý, đặc biệt là trong môi trường thương mại điện tử, nơi người dùng chỉ tương tác với một phần rất nhỏ trong số hàng triệu sản phẩm có sẵn. Điều này dẫn đến ma trận tương tác giữa người dùng và sản phẩm chứa nhiều giá trị rỗng (missing values), gây khó khăn trong việc học mô hình và dự đoán chính xác sở thích người dùng.

Theo Covington [2], dữ liệu thưa và nhiễu là những vấn đề phổ biến trong các hệ thống gợi ý hiện đại. Ví dụ, hành vi người dùng trên nền tảng YouTube thường khó dự đoán vì dữ liệu vừa thưa vừa bị ảnh hưởng bởi các yếu tố không quan sát được. Tương tự, trong các nền tảng thương mại điện tử, người dùng thường không tương tác với phần lớn sản phẩm, và thông tin phản hồi thường là gián tiếp (ngầm định), như lướt xem, nhấp chuột hoặc thời gian ở lại trang, thay vì đánh giá rõ ràng.

Ngoài ra, metadata liên quan đến sản phẩm cũng có thể không được tổ chức đầy đủ hoặc thiếu hệ thống phân loại rõ ràng (ontology), làm cho việc khai thác đặc trưng nội dung gặp nhiều khó khăn.

Đối với các thuật toán lọc cộng tác dạng bộ nhớ (memory-based CF), vấn đề dữ liệu thưa còn kéo theo hạn chế về khả năng mở rộng (scalability). Khi số lượng người dùng và sản phẩm tăng, chi phí

tính toán và lưu trữ tăng mạnh, làm cho hệ thống khó đáp ứng yêu cầu phản hồi theo thời gian thực. Ma trận tương tác trong các hệ thống này thường có tính chất chiều cao (high dimensionality) và độ thưa lớn (high sparsity), với độ thưa được đo lường bằng công thức:

$$\text{Sparsity} = 1 - \frac{\text{Số lượng đánh giá thực tế}}{\text{Tổng số người dùng} \times \text{Tổng số sản phẩm}} \quad (2.7)$$

Giá trị *Sparsity* càng cao, nghĩa là tỷ lệ dữ liệu bị thiếu càng lớn, từ đó làm giảm hiệu quả học của mô hình gợi ý truyền thống. Do đó, cần thiết phải sử dụng các mô hình mạnh mẽ hơn như các mô hình lai hoặc mô hình học sâu để tận dụng tốt cả dữ liệu cấu trúc và phi cấu trúc, đồng thời giảm thiểu ảnh hưởng tiêu cực từ dữ liệu thưa.

Bảng 2.2: Mô tả ma trận đánh giá của người dùng đối với các sản phẩm

UserS	Item A	Item B	Item C	Item D
User 1	5	–	3	–
User 2	–	4	–	2
User 3	1	–	–	–
User 4	–	–	5	4

Trong Bảng 2.2, dữ liệu trong bảng bao gồm các điểm đánh giá của người dùng đối với các sản phẩm, và dấu gạch ngang (–) thể hiện sự thiếu vắng của dữ liệu (tức là không có đánh giá). Dựa vào công thức (2.7), độ thưa được tính như sau:

$$\text{Sparsity} = 1 - \frac{6}{4 \times 4} = 1 - \frac{6}{16} = 1 - 0.375 = 0.625$$

Vậy độ thưa (sparsity) của ma trận là 0.625, tương đương với 62.5%.

2.2.2 Bài toán tương tác mới giữa người dùng và sản phẩm (cold-start)

Trong việc giải quyết bài toán cold-start mô hình sẽ gặp phải 2 vấn đề chính sau:

1. Vấn đề người dùng mới: đây là tình huống mà hệ thống gợi ý không thể cung cấp các đề xuất phù hợp cho người dùng mới, vì thiếu thông tin về sở thích và hành vi của họ.
2. Vấn đề với sản phẩm mới: đây là tình huống khi hệ thống không thể gợi ý các sản phẩm mới cho người dùng hiện tại, do không có đủ dữ liệu về đánh giá hoặc tương tác với các sản phẩm này.

Bảng 2.3: Ví dụ về dữ liệu với Cold-Start Người Dùng và Sản Phẩm

Users	User 1	User 2	User 3	User 4	User 5
Item A	5	-	1	-	-
Item B	-	4	-	-	-
Item C	3	-	-	5	-
Item D	-	2	-	4	-
Item E	-	-	-	-	-

Trong Bảng 2.3 dữ liệu về người dùng và sản phẩm, trong đó có sự xuất hiện của người dùng mới (User 5) và sản phẩm mới (Item E), cả hai đều không có bất kỳ dữ liệu đánh giá nào. Điều này cho

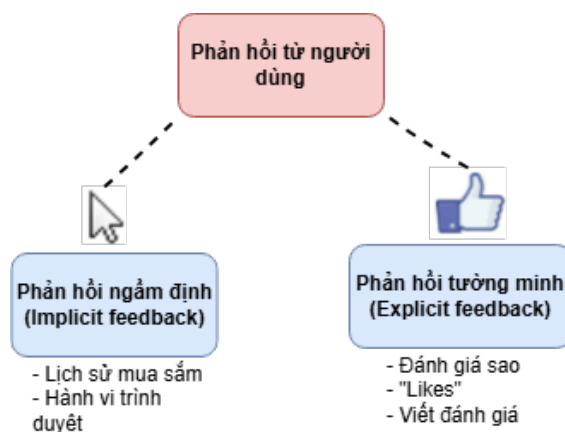
thấy vấn đề cold-start xảy ra khi thiếu thông tin từ người dùng hoặc sản phẩm, làm cho hệ thống không thể đưa ra các gợi ý đáng tin cậy.

Vấn đề cold-start gây khó khăn lớn trong việc tạo ra các gợi ý chính xác cho người dùng và sản phẩm mới. Đối với các sản phẩm mới, thiếu dữ liệu lịch sử khiến việc hiểu rõ đặc điểm của sản phẩm và sở thích người dùng trở nên khó khăn, dẫn đến kết quả gợi ý không chính xác. Tương tự, đối với người dùng mới, việc dự đoán chính xác sở thích và hành vi của họ cũng gặp trở ngại do thiếu dữ liệu lịch sử, làm cho các gợi ý trở nên không chính xác hoặc thậm chí vô dụng. Điều này ảnh hưởng đến hiệu suất và độ chính xác của hệ thống gợi ý, đặc biệt là khi số lượng người dùng mới và sản phẩm mới ngày càng tăng. Do đó, việc giảm thiểu vấn đề cold start là rất quan trọng để đảm bảo tính hiệu quả và ứng dụng rộng rãi của các hệ thống gợi ý [8].

2.3 Biểu diễn dữ liệu trong hệ thống gợi ý

2.3.1 Phân loại dữ liệu trong hệ thống gợi ý

Trong các hệ thống gợi ý, dữ liệu tương tác giữa người dùng và sản phẩm có thể được phân loại thành hai nhóm chính: phản hồi tường minh (explicit feedback) và phản hồi ngầm định (implicit feedback) minh hoạ trong Hình 2.4.



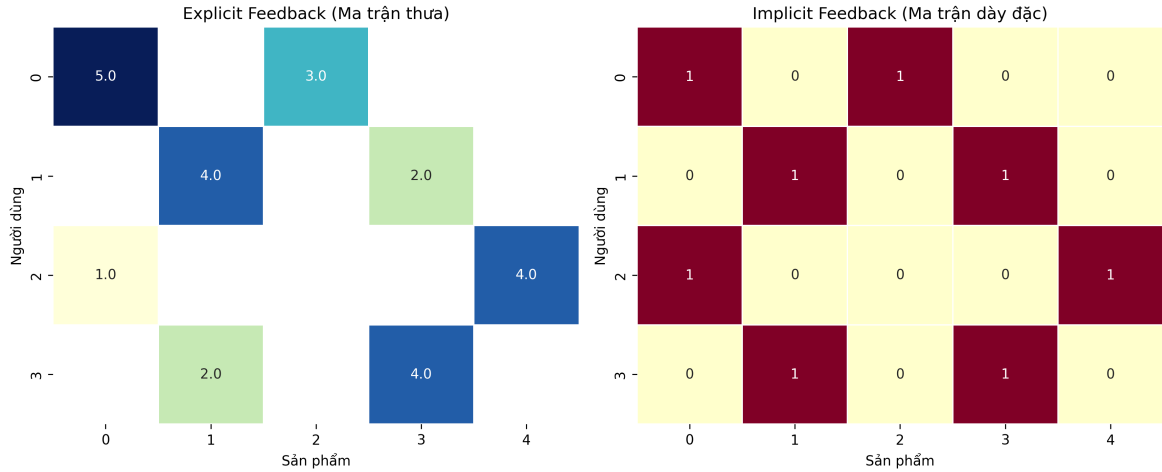
Hình 2.4: Ví dụ phân biệt giữa Implicit và Explicit Feedback

Phản hồi tường minh là những đánh giá trực tiếp mà người dùng chủ động cung cấp, thể hiện rõ mức độ yêu thích hoặc không hài lòng với một sản phẩm hoặc dịch vụ. Ví dụ tiêu biểu bao gồm xếp hạng số sao, điểm đánh giá, nhận xét văn bản hoặc đánh giá “like/dislike”. Dữ liệu này thường có độ chính xác cao nhưng lại khá hạn chế về số lượng, do người dùng thường chỉ đánh giá một phần nhỏ trong kho sản phẩm tổng thể.

Ngược lại, phản hồi ngầm định được suy ra từ hành vi người dùng trong quá trình tương tác, chẳng hạn như số lần nhấp chuột, lượt xem sản phẩm, thêm vào giỏ hàng, thời gian dừng lại trên trang sản phẩm, hoặc hành vi mua hàng. Loại phản hồi này không thể hiện trực tiếp sở thích, nhưng có thể cung cấp lượng dữ liệu lớn, phong phú và cập nhật liên tục. Tuy nhiên, nó có thể bị nhiễu vì không phải hành vi nào cũng phản ánh sự yêu thích thực sự.

Về mặt biểu diễn dữ liệu, explicit feedback thường được tổ chức dưới dạng một ma trận thưa (sparse matrix), bởi vì người dùng chỉ đánh giá một phần nhỏ trong tổng số sản phẩm hiện có. Trong

khi đó, implicit feedback thường được thể hiện thông qua sự có mặt hoặc vắng mặt của một sự một hành vi tương tác, nên dữ liệu tương ứng thường được biểu diễn dưới dạng ma trận dày đặc (dense matrix) [9].



Hình 2.5: Ví dụ về ma trận dữ liệu implicit và explicit feedback

Hình 2.5 minh họa sự khác biệt rõ rệt giữa dữ liệu phản hồi tường minh và phản hồi ngầm định. Có thể thấy, ma trận phản hồi tường minh thường rất thưa, do người dùng chỉ đánh giá một phần nhỏ các sản phẩm. Ngược lại, dữ liệu implicit được thu thập gián tiếp từ hành vi và có xu hướng đầy đủ hơn. Việc hiểu rõ đặc điểm của từng loại phản hồi là cơ sở quan trọng để lựa chọn thuật toán đề xuất phù hợp, như các phương pháp dựa trên nhân tố tiềm ẩn (latent factor models) cho dữ liệu thưa hoặc các kỹ thuật tối ưu hóa nhị phân cho dữ liệu dày đặc.

Bảng 2.4: So sánh giữa phản hồi tường minh và phản hồi ngầm định

Đặc điểm	Explicit Feedback	Implicit Feedback
Nguồn dữ liệu	Đánh giá, xếp hạng, nhận xét	Lướt xem, click, mua hàng
Tính chủ quan	Cao – do người dùng tự khai báo	Thấp – được suy ra
Tính rõ ràng	Thể hiện rõ ý định	Mơ hồ, cần suy luận thêm
Dạng dữ liệu	Ma trận thưa	Ma trận dày đặc
Độ chính xác	Cao	Trung bình đến thấp
Tính phổ biến	Thường hiếm hơn	Phổ biến hơn

Dựa vào Bảng 2.4, việc lựa chọn dữ liệu đầu vào đóng vai trò then chốt trong việc xây dựng hệ thống gợi ý sản phẩm cá nhân hóa. Phản hồi tường minh (explicit) cung cấp thông tin chính xác nhưng thường hiếm và rời rạc, trong khi phản hồi ngầm định (implicit) tuy kém rõ ràng hơn nhưng lại phong phú và phổ biến hơn trong thực tế. Do đó, việc khai thác hiệu quả cả hai loại phản hồi giúp hệ thống gợi ý trở nên linh hoạt, đáp ứng tốt hơn nhu cầu cá nhân hóa trong các kịch bản khác nhau.

2.3.2 Ma trận tương tác người dùng và sản phẩm (Interaction Matrix)

Ma trận tương tác là một biểu diễn dữ liệu cốt lõi trong hệ thống gợi ý. Đây là cách biểu diễn phổ biến mô hình hoá mối quan hệ giữa người dùng và các sản phẩm mà họ đã tương tác. Cấu trúc của

ma trận bao gồm các hàng đại diện cho người dùng và các cột tương ứng với sản phẩm, mỗi phần tử trong ma trận lưu giá trị thể hiện mức độ tương tác hoặc phản hồi giữa người dùng đó với sản phẩm tương ứng.

Theo tác giả Xue và các cộng sự [10], ma trận đánh giá được định nghĩa là $R \in \mathbb{R}^{M \times N}$, trong đó M là số lượng người dùng, N là số lượng sản phẩm, và mỗi phần tử R_{ij} biểu thị đánh giá mà người dùng i dành cho sản phẩm j . Nếu người dùng chưa từng đánh giá sản phẩm đó, phần tử R_{ij} sẽ được gán giá trị **unk** (không xác định).

$$R = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1N} \\ R_{21} & R_{22} & \cdots & R_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ R_{M1} & R_{M2} & \cdots & R_{MN} \end{bmatrix}$$

Từ ma trận R , có thể xây dựng ma trận tương tác $Y \in \mathbb{R}^{M \times N}$ theo hai phương pháp phổ biến, đặc biệt trong bối cảnh xử lý phản hồi ngầm (implicit feedback):

- Phương pháp 1: Nhị phân hóa đơn giản

$$Y_{ij} = \begin{cases} 1 & \text{nếu } R_{ij} \neq \text{unk} \\ 0 & \text{nếu } R_{ij} = \text{unk} \end{cases}$$

Phương pháp này giả định rằng bất kỳ phản hồi nào (dù là đánh giá cao hay thấp) đều ngụ ý rằng người dùng đã từng tương tác với sản phẩm.

- Phương pháp 2: Bảo toàn giá trị phản hồi

$$Y_{ij} = \begin{cases} R_{ij} & \text{nếu } R_{ij} \neq \text{unk} \\ 0 & \text{nếu } R_{ij} = \text{unk} \end{cases}$$

Phương pháp này giữ nguyên giá trị đánh giá nếu có, và gán giá trị 0 nếu không có thông tin phản hồi từ người dùng.

Trong trường hợp dữ liệu explicit feedback, các giá trị trong ma trận thường là các con số cụ thể như điểm đánh giá (rating) do người dùng cung cấp. Ngược lại, với implicit feedback, giá trị thường là nhị phân (1 nếu có tương tác, 0 nếu không có), hoặc đôi khi là giá trị đếm như số lần xem sản phẩm, thêm vào giỏ hàng, hoặc hoàn tất giao dịch mua bán [11] [12].

Một đặc điểm quan trọng của ma trận tương tác là tính chất rất thưa (sparse) do hầu hết người dùng chỉ tương tác với một phần nhỏ trong tổng số sản phẩm. Vấn đề này dẫn đến nhu cầu sử dụng các kỹ thuật như Matrix Factorization, nhằm học các đặc trưng ẩn (latent features) của người dùng và sản phẩm để dự đoán các giá trị còn thiếu [13]. Các mô hình hiện đại hơn, như hệ thống gợi ý lai (hybrid systems) hoặc mô hình học sâu, thường kết hợp ma trận tương tác với đặc trưng nội tại của người dùng và sản phẩm để cải thiện hiệu quả gợi ý, đặc biệt là trong các tình huống cold-start [14].

Trong Bảng 2.5, giá trị 1 biểu thị sự tồn tại của hành vi tương tác như xem sản phẩm, nhấn vào sản phẩm hoặc mua hàng; trong khi giá trị 0 thể hiện không có thông tin về tương tác.

Bảng 2.5: Ma trận tương tác giữa người dùng và sản phẩm với dữ liệu implicit feedback

UserId \ ItemId	100253	145789	632596	785142	365214
110111	1	0	1	0	0
110189	0	1	0	1	0
110121	0	1	1	0	1
116985	1	0	0	0	1

Trong trường hợp dữ liệu explicit feedback Bảng 2.6, các giá trị trong ma trận thường là các con số cụ thể như điểm đánh giá (rating) mà người dùng cung cấp cho sản phẩm. Các giá trị 0 chỉ ra rằng người dùng chưa đánh giá hoặc chưa tương tác với sản phẩm đó.

Bảng 2.6: Ma trận tương tác giữa người dùng và sản phẩm với dữ liệu explicit feedback

User \ Item	100253	145789	632596	785142	365214
110111	5	0	3	0	2
110189	0	4	0	5	0
110121	3	5	4	0	1
116985	4	0	0	2	5

Ma trận tương tác là nền tảng cốt lõi trong các hệ thống gợi ý, phản ánh trực tiếp mối quan hệ giữa người dùng và sản phẩm thông qua các hành vi tương tác. Dù là phản hồi tường minh hay ngầm định, ma trận này đều mang tính chất thưa, đặt ra bài toán quan trọng về cách xử lý dữ liệu thiếu hụt. Việc xây dựng và biểu diễn ma trận một cách hợp lý, kết hợp cùng các kỹ thuật học đặc trưng ẩn như Matrix Factorization hoặc các mô hình học sâu hiện đại, sẽ góp phần nâng cao hiệu quả cá nhân hóa gợi ý, đặc biệt trong các trường hợp dữ liệu khan hiếm hoặc người dùng/sản phẩm mới (cold-start).

2.3.3 Đặc trưng người dùng và sản phẩm

Trong việc xây dựng hệ thống gợi ý thì việc trích xuất đặc trưng của người dùng và sản phẩm có vai trò quan trọng trong việc cải thiện và nâng cao độ chính xác của các đề xuất gợi ý. Việc khai thác và biểu diễn hiệu quả các đặc trưng này giúp hệ thống hiểu rõ hơn về sở thích của người dùng và thuộc tính của sản phẩm để mô hình đưa ra các gợi ý phù hợp hơn, đặc biệt trong những trường hợp dữ liệu thưa hoặc bài toán cold-start [15] [16]. Theo tác giả Han và Karypis [17], các phương pháp dựa trên đặc trưng sản phẩm có thể cải thiện đáng kể hiệu suất gợi ý, đặc biệt khi xử lý các sản phẩm mới.

Đặc trưng người dùng (User Features)

Đặc trưng của người dùng đề cập đến các thông tin mô tả đặc điểm cá nhân hoặc hành vi của họ trong hệ thống gợi ý. Thông thường, đặc trưng người dùng được phân loại thành hai nhóm chính:

- Đặc trưng tường minh (Explicit Features): là các thông tin mà người dùng chủ động cung cấp, bao gồm: Độ tuổi, giới tính, nghề nghiệp, khu vực sinh sống, sở thích khai báo: loại sản phẩm ưa thích, thể loại ưa chuộng, đánh giá, bình luận hoặc điểm số cho các sản phẩm đã sử dụng
- Đặc trưng ẩn (Implicit Features): là các yếu tố được hệ thống suy diễn từ hành vi sử dụng của người dùng như: lịch sử mua hàng, sản phẩm đã xem, đã mua hoặc đã thêm vào giỏ hàng, tần

suất truy cập các danh mục hoặc sản phẩm cụ thể, thời gian tương tác hoặc chuỗi hành vi, thứ tự xem sản phẩm, khoảng thời gian sử dụng dịch vụ.

Để biểu diễn đặc trưng người dùng, có hai phương pháp thường được sử dụng bao gồm:

- Biểu diễn dạng vector: mỗi người dùng được biểu diễn dưới dạng một vector đặc trưng, trong đó mỗi chiều tương ứng với một thuộc tính (tuổi, giới tính, số lượt mua hàng). Các thuộc tính này có thể được mã hoá bằng one-hot, chuẩn hoá, hoặc nhúng (embedding) từ các mô hình học sâu.
- Nhúng đặc trưng (Feature Embedding): sử dụng các phương pháp học tiềm ẩn như Matrix factorization đã được bài nghiên cứu [13] đề cập để học các đặc trưng ẩn hoặc sử dụng mạng nơ-ron để ánh xạ người dùng vào không gian vector tiềm ẩn. Theo như bài nghiên cứu [16] sử dụng mô hình LightFM kết hợp đặc trưng người dùng và sản phẩm để học các vector nhúng tối ưu hoá tương tác.

Một vector đặc trưng của người dùng có thể được ký hiệu như sau:

$$\mathbf{u}_i = [u_{i_1}, u_{i_2}, \dots, u_{i_d}]$$

Trong đó, u_{i_j} là giá trị của đặc trưng thứ j của người dùng i , và d là số chiều của vector đặc trưng.

Mặc dù bài báo của Han và Karypis [17] không thảo luận trực tiếp về đặc trưng người dùng, các tương tác của người dùng thông qua giỏ hàng được sử dụng gián tiếp để suy ra sở thích, hỗ trợ việc gợi ý dựa trên đặc trưng sản phẩm.

Đặc trưng sản phẩm (Item Features)

Đặc trưng sản phẩm là các thuộc tính mô tả sản phẩm, được sử dụng để xác định mức độ phù hợp của sản phẩm với từng người dùng. Chúng cũng được chia thành hai nhóm chính:

- Đặc trưng nội tại (Intrinsic Features): các thuộc tính cố định của sản phẩm như danh mục sản phẩm (quần áo, điện tử, thực phẩm), giá cả, thương hiệu, kích thước, màu sắc, mô tả sản phẩm hoặc từ khoá liên quan
- Đặc trưng ngữ cảnh (Contextual Features): các thuộc tính phụ thuộc vào bối cảnh hoặc hành vi người dùng như tần suất sản phẩm được xem hoặc mua, điểm đánh giá trung bình của sản phẩm, sự phổ biến của sản phẩm trong một khoảng thời gian nhất định

Trong bài luận [17] nhấn mạnh vai trò quan trọng của đặc trưng sản phẩm trong hệ thống gợi ý, đặc biệt hữu ích khi xử lý các sản phẩm mới chưa có nhiều tương tác. Họ đề xuất rằng đặc trưng sản phẩm có thể được biểu diễn thông qua vector trung tâm của cụm sản phẩm hoặc chọn sản phẩm đại diện trong cụm, từ đó xác định các thuộc tính nổi bật phục vụ cho việc gợi ý hiệu quả.

Để biểu diễn đặc trưng sản phẩm, các kỹ thuật phổ biến bao gồm:

- Vector hoá TF-IDF: đối với các thuộc tính dạng văn bản như mô tả sản phẩm hoặc từ khoá, phương pháp TF-IDF được sử dụng để chuyển đổi nội dung này thành các vector số, phản ánh mức độ quan trọng của từ ngữ trong từng sản phẩm [5].

- Nhúng đặc trưng sản phẩm: các mô hình học sâu như mạng nơ-ron hoặc các phương pháp phân rã ma trận như LightFM có thể ánh xạ sản phẩm vào không gian vector tiềm ẩn, trong đó các sản phẩm có đặc điểm tương đồng sẽ nằm gần nhau hơn. Theo Kula [6], mô hình LightFM có khả năng học các biểu diễn nhúng cho siêu dữ liệu của người dùng và sản phẩm, từ đó hỗ trợ hệ thống gợi ý hoạt động hiệu quả ngay cả đối với các trường hợp không có dữ liệu tương tác trước đó [18].
- Xử lý dữ liệu phi cấu trúc: áp dụng CNN để trích xuất đặc trưng cấp cao từ hình ảnh hoặc mô tả dài, đặc biệt cho các đặc trưng ngữ cảnh như xu hướng thị giác [16].

Một sản phẩm có thể được biểu diễn bằng vector đặc trưng như sau:

$$\mathbf{v}_j = [v_{j_1}, v_{j_2}, \dots, v_{j_d}]$$

Trong đó, v_{j_k} là giá trị của đặc trưng thứ k của sản phẩm j , và d là số chiều của vector đặc trưng. Trong bài nghiên cứu [17] cũng chỉ ra rằng việc tổng hợp các đặc trưng nổi bật từ các sản phẩm được gợi ý giúp xác định rõ hơn các yếu tố ảnh hưởng đến quyết định đề xuất, từ đó cải thiện hiệu quả hệ thống.

Đặc trưng sản phẩm thường được sử dụng kết hợp với đặc trưng người dùng trong mô hình lai để cải thiện khả năng cá nhân hóa. Các phương pháp như factorization machines, deep neural networks hoặc attention-based models đã chứng minh hiệu quả trong việc xử lý các đặc trưng này [16].

2.3.4 Encoding dữ liệu

Dữ liệu đầu vào của hệ thống gợi ý thường chứa nhiều đặc trưng rời rạc như mã người dùng, mã sản phẩm, danh mục, thương hiệu hoặc loại hành vi như xem, nhấp chuột, thêm vào giỏ hàng và mua hàng. Để mô hình học máy có thể xử lý hiệu quả các đặc trưng này, chúng cần được chuyển đổi sang dạng số thông qua các kỹ thuật mã hóa. Hai phương pháp phổ biến được sử dụng là one-hot encoding và embedding.

One-hot Encoding

One-hot encoding là một kỹ thuật mã hóa đơn giản theo kiểu nhị phân, trong đó mỗi giá trị rời rạc được biểu diễn dưới dạng một vector có độ dài bằng tổng số giá trị có thể có. Trong vector này, chỉ duy nhất một phần tử được gán giá trị 1 tại vị trí tương ứng với giá trị đang xét, còn lại tất cả đều là 0. Nếu thuộc tính có k giá trị phân biệt, thì vector one-hot tương ứng sẽ có độ dài k .

Bảng 2.7: Ví dụ One-hot encoding cho đặc trưng "Loại sản phẩm"

Loại sản phẩm	One-hot vector
Smartphone	[1, 0, 0, 0, 0]
Laptop	[0, 1, 0, 0, 0]
Tablet	[0, 0, 1, 0, 0]
Headphones	[0, 0, 0, 1, 0]
Smartwatch	[0, 0, 0, 0, 1]

Hạn chế của One-hot encoding như Bảng 2.7 là không thể hiện được mối quan hệ ngữ nghĩa giữa các loại sản phẩm (ví dụ: Smartphone và Tablet có thể liên quan nhưng lại được mã hóa hoàn

toàn tách biệt). Ngoài ra, kích thước vector tăng theo số lượng loại, dẫn đến biểu diễn thưa (sparse representation), làm giảm hiệu suất tính toán và khả năng khái quát hóa của mô hình.

Embedding

Embedding là kỹ thuật ánh xạ các giá trị phân loại vào không gian vector thực chiều thấp hơn, cho phép biểu diễn đặc trưng dưới dạng dense vector có thể học được trong quá trình huấn luyện.

Cho tập người dùng $U = \{u_1, \dots, u_m\}$ và tập sản phẩm $I = \{i_1, \dots, i_n\}$, vector nhúng tương ứng được biểu diễn là:

$$\mathbf{e}_u = \mathbf{E}_u \in \mathbb{R}^d, \quad \mathbf{e}_i = \mathbf{E}_i \in \mathbb{R}^d$$

Trong đó E_u, E_i là embedding được học từ dữ liệu tương tác. Các vector này có thể được sử dụng trong các mô hình dự đoán, ví dụ:

- Mô hình dot-product: $\hat{r}_{ui} = e_u^\top e_i$
- Mô hình phi tuyến Multilayer Perceptron (MLP): $\hat{r}_{ui} = \text{MLP}([e_u \| e_i])$

Embedding có khả năng học các mối quan hệ ngữ nghĩa và tương quan giữa người dùng và sản phẩm, giúp tăng độ chính xác trong các hệ thống gợi ý [18][12].

Bảng 2.8: So sánh hai phương pháp mã hóa

Tiêu chí	One-hot Encoding	Embedding
Kích thước vector	Rất cao (sparse)	Thấp (dense)
Học từ dữ liệu	Không	Có (trainable)
Biểu diễn ngữ nghĩa	Không	Có khả năng
Ứng dụng học sâu	Hạn chế	Phổ biến

Embedding là phương pháp mã hóa đặc trưng được sử dụng rộng rãi trong các hệ thống gợi ý hiện đại. Các mô hình như LightFM [18], Neural Collaborative Filtering (NCF) [12], Deep Factorization Machine (DeepFM) [19], hay Deep Interest Network (DIN) [20] đều sử dụng embedding để học các biểu diễn tiềm ẩn cho người dùng, sản phẩm và metadata. Kỹ thuật này đặc biệt hữu ích trong việc xử lý cold-start và cải thiện hiệu suất gợi ý bằng cách kết hợp với các kiến trúc sâu như CNN hoặc Transformer. Encoding là bước tiền xử lý then chốt, trong đó embedding nổi bật nhờ khả năng tổng quát hóa và học ngữ nghĩa từ dữ liệu, giúp hệ thống gợi ý trở nên chính xác và cá nhân hóa hơn.

2.4 Tổng quan về học máy và học sâu trong gợi ý

2.4.1 Học máy trong hệ thống gợi ý

Học máy đóng vai trò cốt lõi trong các hệ thống gợi ý hiện đại, đặc biệt trong việc dự đoán sở thích của người dùng dựa trên dữ liệu lịch sử. Các thuật toán Machine Learning (ML) được sử dụng để học các mẫu (patterns) từ dữ liệu tương tác, từ đó đưa ra gợi ý chính xác hơn. Các phương pháp ML phổ biến bao gồm:

- Phân rã ma trận (Matrix Factorization): như đã đề cập ở phần 2.1, các phương pháp như Singular Value Decomposition hoặc Alternating Least Squares được sử dụng để rút trích các đặc

trung tiềm ẩn từ ma trận tương tác người dùng - sản phẩm. Công thức cơ bản của Alternating Least Squares (ALS) tối ưu hóa hàm mất mát:

$$\min_{U,V} \sum_{(u,i) \in R} (r_{ui} - \mathbf{u}_u^\top \mathbf{v}_i)^2 + \lambda (|\mathbf{u}_u|^2 + |\mathbf{v}_i|^2) \quad (2.8)$$

Trong đó, r_{ui} là điểm đánh giá của người dùng u cho sản phẩm i , \mathbf{u}_u và \mathbf{v}_i lần lượt là vector đặc trưng của người dùng và sản phẩm, và λ là hệ số điều chuẩn để giảm hiện tượng overfitting. Phương pháp ALS hoạt động hiệu quả trên dữ liệu phản hồi tường minh (explicit feedback), tuy nhiên gặp nhiều hạn chế trong các bài toán cold-start và dữ liệu thưa [9].

- K-Nearest Neighbors: K-Nearest Neighbors (KNN) tìm các người dùng hoặc sản phẩm tương tự dựa trên độ tương đồng cosine hoặc Pearson. Có hai biến thể phổ biến của KNN: user-based KNN, trong đó hệ thống gợi ý các sản phẩm dựa trên hành vi của những người dùng có xu hướng tương tác giống nhau và item-based KNN, trong đó hệ thống gợi ý các sản phẩm tương tự với những sản phẩm mà người dùng đã đánh giá cao. Phương pháp này đơn giản nhưng không mở rộng tốt với dữ liệu lớn [21].
- Factorization Machines: Factorization Machine (FM) học các tương tác giữa các đặc trưng (người dùng, sản phẩm, metadata) trong không gian tiềm ẩn, phù hợp với dữ liệu thưa và hệ thống gợi ý lai [22]. Mô hình FM có thể biểu diễn như sau:

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^n w_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j \quad (2.9)$$

Trong đó, \mathbf{x} là vector đặc trưng đầu vào, bao gồm các thông tin về người dùng, sản phẩm, v.v.; \mathbf{v}_i là vector nhúng tiềm ẩn cho đặc trưng thứ i ; và $\langle \cdot, \cdot \rangle$ là tích vô hướng giữa hai vector.

- Cây quyết định và Support Vector Machine (SVM): các phương pháp này được áp dụng trong lọc nội dung, phân loại sản phẩm dựa trên đặc trưng như danh mục, giá cả [14].

2.4.2 Học sâu trong hệ thống gợi ý

Học sâu đã trở thành một trong những hướng tiếp cận nổi bật trong các hệ thống gợi ý hiện đại, nhờ khả năng học biểu diễn đặc trưng phức tạp và phi tuyến từ dữ liệu người dùng, sản phẩm và metadata. Các mô hình học sâu cho phép khai thác hiệu quả mối quan hệ tiềm ẩn giữa các thực thể, từ đó nâng cao độ chính xác trong dự đoán hành vi người dùng, đặc biệt ở quy mô lớn [23].

Mạng nơ-ron nhân tạo mô hình hóa tương tác giữa người dùng và sản phẩm thông qua các tầng ẩn trong mạng. Một mô hình Artificial Neural Network (ANN) cơ bản gồm: tầng đầu vào (vector đặc trưng), các tầng ẩn sử dụng hàm kích hoạt phi tuyến như Rectified Linear Unit (ReLU), và tầng đầu ra dự đoán điểm tương tác. Ví dụ điển hình là mô hình Neural Collaborative Filtering, trong đó các tầng ẩn giúp học các mối quan hệ phi tuyến, vượt trội so với phương pháp phân rã ma trận truyền thống [12].

Lớp nhúng (Embedding Layer) để xử lý các đặc trưng rời rạc như ID người dùng hoặc sản phẩm, các mô hình học sâu thường sử dụng lớp nhúng để ánh xạ các chỉ mục thành vector liên tục có chiều

thấp. Biểu diễn toán học như sau:

$$\mathbf{e}_u = E_u[u], \quad \mathbf{e}_i = E_i[i] \quad (2.10)$$

Trong đó E_u và E_i là ma trận nhúng tương ứng với người dùng và sản phẩm; $\mathbf{e}_u, \mathbf{e}_i \in \mathbb{R}^d$ là vector nhúng trong không gian chiều thấp. Cách biểu diễn này giúp học được các quan hệ ngữ nghĩa, ví dụ các sản phẩm tương tự sẽ có vector gần nhau trong không gian đặc trưng [16].

Cơ chế chú ý (Attention Mechanism) là một cơ chế quan trọng trong học sâu, cho phép mô hình tập trung vào các thành phần đầu vào quan trọng nhất. Công thức attention cơ bản được biểu diễn như sau:

$$\alpha_i = \text{softmax}(\mathbf{q}^T \mathbf{k}_i), \quad \mathbf{y} = \sum_i \alpha_i \mathbf{v}_i \quad (2.11)$$

Trong đó $\mathbf{q}, \mathbf{k}_i, \mathbf{v}_i$ lần lượt là vector truy vấn, khóa và giá trị. Cơ chế attention đã được áp dụng thành công trong mô hình DIN nhằm ưu tiên các sản phẩm liên quan đến hành vi gần đây của người dùng [20].

Các mô hình học sâu có xu hướng dễ bị overfitting, đặc biệt khi dữ liệu huấn luyện bị thừa hoặc không cân đối [23]. Một số kỹ thuật thường dùng để giảm thiểu overfitting gồm:

- Regularization: Thêm điều chuẩn L2 vào hàm mất mát như sau:

$$\mathcal{L} = \mathcal{L}_{\text{gốc}} + \lambda \|\mathbf{w}\|^2 \quad (2.12)$$

Giúp giới hạn độ lớn của các tham số mô hình, từ đó tránh hiện tượng khớp quá mức.

- Dropout: Loại bỏ ngẫu nhiên một phần (thường 20–50%) số lượng nơ-ron trong quá trình huấn luyện để tăng khả năng tổng quát hóa của mô hình [24].

Các mô hình học sâu trong hệ thống gợi ý có thể chia thành hai loại đầu ra chính: phân loại và dự đoán điểm số.

Đối với bài toán phân loại, đầu ra của mô hình thường là một lớp Softmax, giúp phân loại các tương tác giữa người dùng và sản phẩm. Đầu ra Softmax có dạng:

$$p(\text{interaction} = 1 \mid \mathbf{x}) = \text{softmax}(\mathbf{W}^T \mathbf{x}) \quad (2.13)$$

Trong đó, \mathbf{W} là ma trận trọng số và \mathbf{x} là vector đặc trưng của người dùng và sản phẩm.

Đối với các bài toán dự đoán điểm số (rating prediction), mô hình thường sử dụng một lớp Linear để đưa ra dự đoán cho điểm tương tác, ví dụ:

$$\hat{r}_{ui} = w_0 + \mathbf{w}_u^T \mathbf{e}_u + \mathbf{w}_i^T \mathbf{e}_i \quad (2.14)$$

Trong đó, \hat{r}_{ui} là điểm số dự đoán của người dùng u đối với sản phẩm i , và $\mathbf{e}_u, \mathbf{e}_i$ là các vector nhúng tương ứng của người dùng và sản phẩm.

Trong khóa luận này, mô hình đề xuất sử dụng kiến trúc DNN để phân loại hành vi người dùng thành ba lớp: xem, thêm vào giỏ hàng, và thực hiện giao dịch. Mô hình kết hợp các đặc trưng nhúng từ LightFM cùng với metadata (như thông tin về sản phẩm và người dùng) và đặc trưng thời gian.

Các đặc trưng này được đưa qua nhiều tầng dense phi tuyến (fully connected layers) với các hàm kích hoạt phi tuyến như ReLU để học được các biểu diễn tiềm ẩn phức tạp giữa người dùng và sản phẩm.

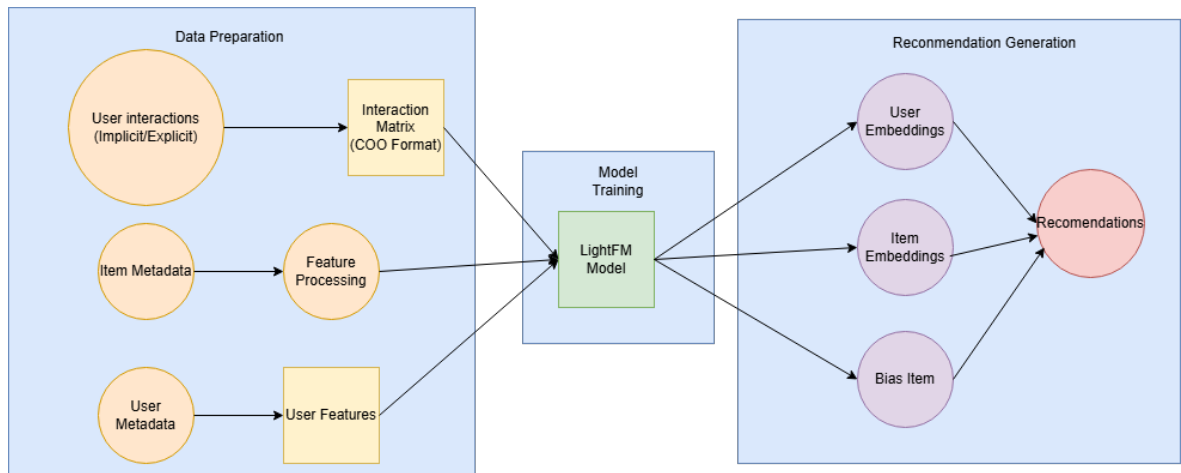
2.5 Giới thiệu mô hình LightFM

LightFM là một thư viện mã nguồn mở cho hệ thống gợi ý lai (hybrid recommender system), được thiết kế để xử lý cả dữ liệu implicit feedback (như lượt xem, nhấp chuột, mua hàng) và explicit feedback (như điểm đánh giá), kết hợp ưu điểm của matrix factorization và content-based filtering [18]. Nó tận dụng metadata của người dùng và sản phẩm để cải thiện khả năng cá nhân hóa, đặc biệt trong các tình huống cold-start [25].

Surprise, ngược lại, là một thư viện Python mạnh mẽ, tập trung vào xử lý dữ liệu explicit feedback như điểm đánh giá, cung cấp các thuật toán như SVD, KNN, và BaselineOnly với giao diện dễ sử dụng [26]. Surprise phù hợp cho các nhà phát triển muốn thử nghiệm nhanh các mô hình gợi ý mà không cần cấu hình phức tạp.

LightFM và Surprise đều là các công cụ quan trọng trong hệ thống gợi ý, nhưng chúng nhắm đến các mục tiêu khác nhau. LightFM phù hợp với các nền tảng thương mại điện tử lớn, nơi dữ liệu người dùng đa dạng và cần tích hợp nhiều nguồn thông tin. Trong khi đó, Surprise là lựa chọn lý tưởng cho các dự án nhỏ hoặc các ứng dụng cần kết quả nhanh với dữ liệu đánh giá tập trung.

2.5.1 Kiến trúc và nguyên lý hoạt động



Hình 2.6: Tổng quan mô hình lai của LightFM

Hình 2.6 Pipeline dữ liệu của LightFM bắt đầu với quá trình chuẩn bị dữ liệu (Data Preparation), trong đó dữ liệu tương tác người dùng-mục (bao gồm cả implicit như lượt xem, nhấp chuột và explicit như đánh giá) được thu thập và chuyển thành ma trận tương tác thưa (Interaction Matrix) ở định dạng Coordinate Format (COO), trong khi thông tin phụ trợ từ item metadata (ví dụ: thể loại, thuộc tính) và user metadata (ví dụ: độ tuổi, sở thích) được xử lý thành các đặc trưng (features) cho mục và người dùng. Tiếp theo, dữ liệu này được đưa vào quá trình huấn luyện mô hình (Model Training) với LightFM, sử dụng matrix factorization kết hợp đặc trưng phụ trợ để học các vector ẩn (latent factors) đại diện cho người dùng và mục. Cuối cùng, trong giai đoạn tạo gợi ý (Recommendation Generation),

mô hình sinh ra user embeddings và item embeddings, áp dụng bias item để điều chỉnh dự đoán, và tính điểm gợi ý bằng tích vô hướng để xếp hạng và đưa ra danh sách khuyến nghị. Pipeline này cho phép LightFM tận dụng cả dữ liệu tương tác và thông tin phụ trợ, giải quyết hiệu quả các vấn đề như cold-start và dữ liệu thưa thớt

LightFM biểu diễn người dùng và sản phẩm trong một không gian nhúng chung, đồng thời tích hợp siêu dữ liệu (metadata) để cải thiện hiệu suất trong các tình huống dữ liệu thưa hoặc cold-start [25]. Điểm dự đoán tương tác giữa người dùng u và sản phẩm i được tính theo công thức:

$$\hat{r}_{ui} = \langle \mathbf{e}_u, \mathbf{e}_i \rangle + b_u + b_i \quad (2.15)$$

Trong đó \mathbf{e}_u và \mathbf{e}_i là vector nhúng (embedding vectors) của người dùng và sản phẩm tương ứng, còn b_u và b_i là các thành phần bias. Với các bài toán phản hồi ngầm (implicit feedback), LightFM sử dụng hàm mất mát Weighted Approximate-Rank Pairwise (Weighted Approximate-Rank Pairwise (WARP)) nhằm tối ưu hóa xếp hạng, giúp ưu tiên các sản phẩm có mức độ phù hợp cao hơn trong danh sách gợi ý.

LightFM có thể được mở rộng để xử lý các đặc trưng thời gian thực, chẳng hạn như hành vi người dùng theo mùa hoặc xu hướng mua sắm ngắn hạn. Ví dụ, trong một nền tảng thương mại điện tử, LightFM có thể ưu tiên gợi ý các sản phẩm liên quan đến lễ hội (như quà Tết) bằng cách tăng trọng số cho các đặc trưng liên quan đến thời điểm. Ngoài ra, LightFM hỗ trợ tích hợp với các mô hình học sâu, như sử dụng các lớp tích chập để xử lý hình ảnh sản phẩm, giúp tăng tính chính xác trong các ứng dụng phức tạp.

Surprise sử dụng thuật toán SVD để phân rã ma trận tương tác người dùng–sản phẩm thành hai ma trận nhúng \mathbf{u}_u (người dùng) và \mathbf{v}_i (sản phẩm), bằng cách tối ưu hóa hàm mất mát sau:

$$\min_{U,V} \sum_{(u,i) \in R} (r_{ui} - \mathbf{u}_u^\top \mathbf{v}_i)^2 + \lambda (\|\mathbf{u}_u\|^2 + \|\mathbf{v}_i\|^2)$$

Trong đó r_{ui} là đánh giá thực tế của người dùng u đối với sản phẩm i , còn λ là hệ số điều chuẩn để tránh overfitting [26][27]. Thư viện Surprise còn cung cấp các công cụ đánh giá hiệu suất như RMSE và MAE, hỗ trợ so sánh chất lượng của các mô hình gợi ý một cách hiệu quả [26].

Surprise có thể được tùy chỉnh để tích hợp với các pipeline dữ liệu lớn, chẳng hạn như kết hợp với Apache Spark để xử lý các tập dữ liệu đánh giá quy mô lớn. Ngoài ra, Surprise cho phép nhà phát triển thêm các đặc trưng bổ sung (như thông tin nhân khẩu học) thông qua tiền xử lý dữ liệu, giúp cải thiện độ chính xác trong các kịch bản thực tế. Ví dụ, trong một hệ thống gợi ý phim, Surprise có thể chuẩn hóa điểm đánh giá dựa trên độ tuổi người dùng để tăng tính cá nhân hóa.

2.5.2 Ưu điểm và nhược điểm của LightFM so với Surprise

Mô hình LightFM có những ưu điểm đáng chú ý khi tích hợp metadata, giúp cải thiện hiệu quả trong các tình huống cold-start. Hơn nữa, nó hỗ trợ cả implicit và explicit feedback [25]. Về mặt tự viết, LightFM rất linh hoạt khi làm việc với dữ liệu động, chẳng hạn như việc thêm danh mục mới, và dễ dàng tích hợp vào các ứng dụng web hoặc di động. Tuy nhiên, một nhược điểm của LightFM là phức tạp hơn so với các mô hình khác, phụ thuộc vào chất lượng của metadata, và có thể tốn tài nguyên trong quá trình xử lý [28]. Việc tối ưu siêu tham số trong LightFM cũng mất thời gian, và nó có thể gặp khó khăn khi xử lý dữ liệu cực lớn nếu không tối ưu.

Mặt khác, Surprise SVD là một mô hình đơn giản và hiệu quả, đặc biệt là với explicit feedback, và dễ dàng triển khai [26]. Nó cũng hỗ trợ các phương pháp đánh giá như RMSE và MAE, giúp so sánh hiệu suất các thuật toán [27]. Surprise SVD có thể triển khai nhanh chóng và phù hợp với các dự án nhỏ, ổn định khi làm việc với dữ liệu đánh giá đồng đều. Tuy nhiên, mô hình này không tích hợp metadata, kém hiệu quả trong các tình huống cold-start, và không phù hợp với implicit feedback [25][26]. Ngoài ra, nó cũng thiếu sự linh hoạt trong việc cá nhân hóa cao và không tận dụng được các xu hướng ngắn hạn của người dùng.

2.5.3 Khi nào sử dụng LightFM và Surprise SVD

Mô hình LightFM là sự lựa chọn lý tưởng khi có metadata phong phú, như danh mục sản phẩm hoặc nhân khẩu học, và đặc biệt hiệu quả với dữ liệu implicit (như lượt xem hoặc nhấp chuột) hoặc trong các tình huống cold-start [28][29]. Ngoài ra, LightFM còn thích hợp khi cần thích nghi với dữ liệu động, đặc biệt là trong các lĩnh vực thương mại điện tử, hoặc khi cần tích hợp học sâu để xử lý dữ liệu hình ảnh và văn bản.

Trong khi đó, Surprise SVD lại phù hợp hơn khi làm việc với dữ liệu explicit, như điểm đánh giá của người dùng, và khi tài nguyên hạn chế [26][27]. Mô hình này cũng thích hợp cho các dự án nhỏ, với ma trận tương tác mật độ cao, và không yêu cầu cá nhân hóa phức tạp.

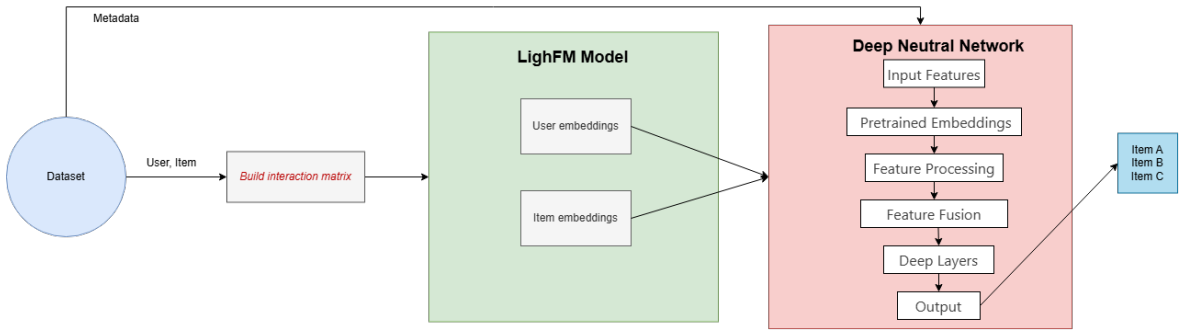
2.6 Tổng kết chương

Chương 2 đã cung cấp các nền tảng lý thuyết quan trọng cho hệ thống gợi ý, bao gồm khái niệm, vai trò, và các phương pháp xây dựng như lọc nội dung, lọc cộng tác, và hệ thống lai. Những thách thức như dữ liệu thưa và cold-start, cùng cách biểu diễn dữ liệu qua ma trận tương tác, đặc trưng, và kỹ thuật mã hóa, được phân tích rõ ràng. Tổng quan về học máy, học sâu, và mô hình LightFM so sánh với Surprise SVD cũng được trình bày, tạo cơ sở vững chắc để triển khai hệ thống gợi ý lai tích hợp LightFM và DNN trong các chương tiếp theo.

Chương 3. Phân tích và thiết kế mô hình

3.1 Kiến trúc tổng thể của hệ thống

Đề tài đề xuất một phương pháp kết hợp giữa mô hình LightFM với mạng nơ-ron sâu (DNN), nhằm tận dụng ưu điểm của cả hai hướng tiếp cận: học biểu diễn (representation learning) từ dữ liệu tương tác và khai thác thông tin metadata giàu ngữ nghĩa. Phương pháp hướng đến việc giải quyết các bài toán gợi ý dựa trên dữ liệu implicit như hành vi xem, thêm vào giỏ hàng, hoặc mua hàng.



Hình 3.1: Tổng quan mô hình gợi ý sản phẩm

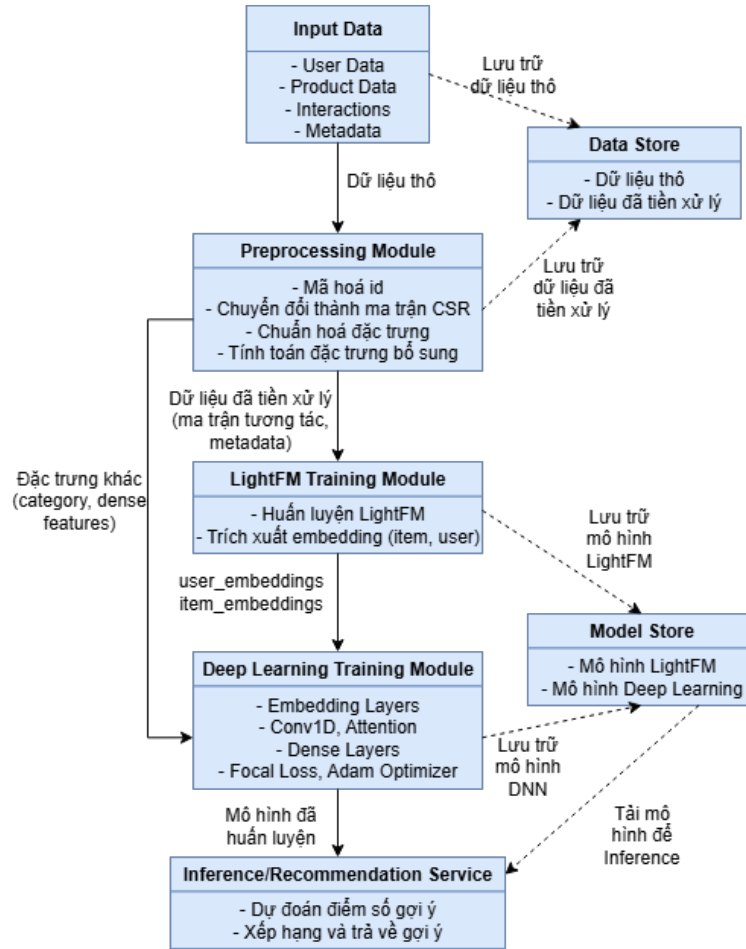
Trong Hình 3.1, kiến trúc mô hình bao gồm hai giai đoạn chính:

1. Huấn luyện embedding từ LightFM: sử dụng LightFM để học các vector biểu diễn (embedding) cho người dùng và sản phẩm dựa trên thông tin tương tác ngầm định. Mô hình này kết hợp các yếu tố của CF và CBF, nhờ vậy có thể khai thác cả hành vi lịch sử lẫn metadata như thể loại sản phẩm, danh mục cha, và ID người dùng.
2. Mạng DNN tích hợp metadata và đặc trưng liên tục: sau khi embedding được học từ LightFM, các vector này được kết hợp với metadata rời rạc (ví dụ: category, parent category) và các đặc trưng liên tục như giá (price), đánh giá (rating), và số lượng đánh giá (review count). Các đặc trưng liên tục được chuẩn hóa và đưa vào mạng nơ-ron tích chập (Conv1D) nhằm khai thác các mẫu cục bộ trong chuỗi dữ liệu. Kết quả đầu ra là xác suất người dùng sẽ thực hiện hành vi cụ thể với sản phẩm (xem, thêm vào giỏ hàng, mua hàng).

Mô hình tận dụng hiệu quả khả năng học biểu diễn phi tuyến của DNN cùng với embedding từ LightFM để mô hình hóa các tương tác phức tạp, từ đó cải thiện hiệu suất gợi ý và khả năng khám phá đối với người dùng/sản phẩm mới (cold-start).

Đối với những tập dữ liệu dạng tường minh (explicit) nơi người dùng đánh giá sản phẩm theo thang điểm (từ 1 đến 5 sao) thì phương pháp sử dụng mô hình Surprise SVD thay vì LightFM. Mô hình SVD trong thư viện Surprise sử dụng kỹ thuật phân rã ma trận để học các latent factors từ dữ liệu đánh giá, cho phép dự đoán điểm số đánh giá (rating prediction) với độ chính xác cao. Trong trường hợp này, đầu ra của mô hình không phải là xác suất hành vi, mà là điểm số dự đoán \hat{r}_{ui} cho người dùng u và sản phẩm i . Điều này phù hợp với các hệ thống gợi ý truyền thống tập trung vào bài toán xếp hạng dựa trên độ yêu thích.

3.2 Tổng quan Pipeline xử lý dữ liệu trong mô hình gợi ý



Hình 3.2: Chi tiết pipeline dữ liệu của mô hình gợi ý sản phẩm

Mô hình gợi ý được thiết kế theo dạng tuyến tính (pipeline) như Hình 3.2, bao gồm nhiều thành phần liên kết chặt chẽ với nhau, từ khâu xử lý dữ liệu đầu vào đến sinh ra kết quả gợi ý cuối cùng. Cụ thể, mô hình bao gồm các thành phần chính sau:

1. Dữ liệu đầu vào

Mô hình sử dụng ba loại dữ liệu chính: dữ liệu người dùng, dữ liệu sản phẩm và dữ liệu tương tác. Dữ liệu người dùng bao gồm các thông tin định danh và mô tả như mã người dùng `user_id`, độ tuổi, giới tính, v.v. Dữ liệu sản phẩm bao gồm mã sản phẩm `item_id`, danh mục sản phẩm `category_id`, danh mục cha, giá bán cùng các thuộc tính mô tả khác. Dữ liệu tương tác ghi nhận các hành vi của người dùng như xem, thêm vào giỏ hàng, mua hàng,... Các hành vi này có thể mang tính tường minh (explicit) hoặc ngầm định (implicit), được biểu diễn bằng biến `interaction_value`. Ngoài ra, các thông tin bổ sung như thời điểm tương tác `timestamp`, giờ trong ngày, thứ trong tuần, mã phiên truy cập `session_id` cũng được sử dụng để mô hình hiểu

rõ hơn về ngữ cảnh thời gian.

2. Tiền xử lý dữ liệu

Dữ liệu đầu vào được xử lý để phù hợp với yêu cầu của mô hình học sâu. Các thuộc tính rời rạc như mã người dùng, mã sản phẩm và danh mục được mã hóa thành các chỉ số nguyên liên tục để phục vụ cho quá trình ánh xạ sang không gian véc-tơ. Dữ liệu tương tác được chuyển đổi thành ma trận thưa Compressed Sparse Row (CSR) giữa người dùng và sản phẩm. Các đặc trưng liên tục như giá, độ tuổi và thời gian được chuẩn hóa nhằm đảm bảo ổn định trong quá trình huấn luyện. Ngoài ra, các đặc trưng tuần hoàn từ thời gian như `hour_sin`, `hour_cos`, `dayofweek_sin`, `dayofweek_cos` cũng được xây dựng để hỗ trợ mô hình học được tính chu kỳ.

3. Huấn luyện mô hình LightFM

Mô hình LightFM được sử dụng để kết hợp giữa lọc cộng tác và lọc theo nội dung, nhằm học được các biểu diễn véc-tơ (embedding) giàu ngữ nghĩa cho người dùng và sản phẩm. Quá trình huấn luyện dựa trên ma trận tương tác cùng thông tin phụ trợ, giúp tạo ra các véc-tơ mang tính cá nhân hóa cao, đóng vai trò làm đặc trưng đầu vào cho mô hình học sâu ở bước tiếp theo.

4. Huấn luyện mô hình học sâu

Các véc-tơ biểu diễn từ LightFM được kết hợp với các đặc trưng đầu vào khác như danh mục sản phẩm, giá bán, đặc trưng thời gian,... để đưa vào mô hình học sâu. Kiến trúc mô hình bao gồm các lớp embedding học từ đầu cho các thuộc tính rời rạc chưa có embedding, sau đó là lớp tích chập một chiều (Conv1D) kết hợp với gộp cực đại toàn cục (GlobalMaxPooling1D) để trích xuất đặc trưng cục bộ. Cơ chế chú ý đa đầu (Multi-Head Attention) được áp dụng để mô hình có thể học được mối quan hệ phức tạp giữa người dùng và sản phẩm. Cuối cùng là các tầng dày (Dense) nhằm tổng hợp thông tin và đưa ra xác suất dự đoán cho từng tương tác. Hàm mất mát Focal Loss được sử dụng để xử lý vấn đề mất cân bằng dữ liệu và nâng cao độ chính xác với các trường hợp khó phân loại.

5. Suy luận và gợi ý sản phẩm

Sau khi quá trình huấn luyện hoàn tất, mô hình được sử dụng để dự đoán xác suất tương tác giữa từng người dùng và các sản phẩm tiềm năng. Các sản phẩm sau đó được xếp hạng theo điểm dự đoán, từ đó sinh ra danh sách gợi ý cá nhân hóa cho từng người dùng.

6. Lưu trữ dữ liệu và mô hình

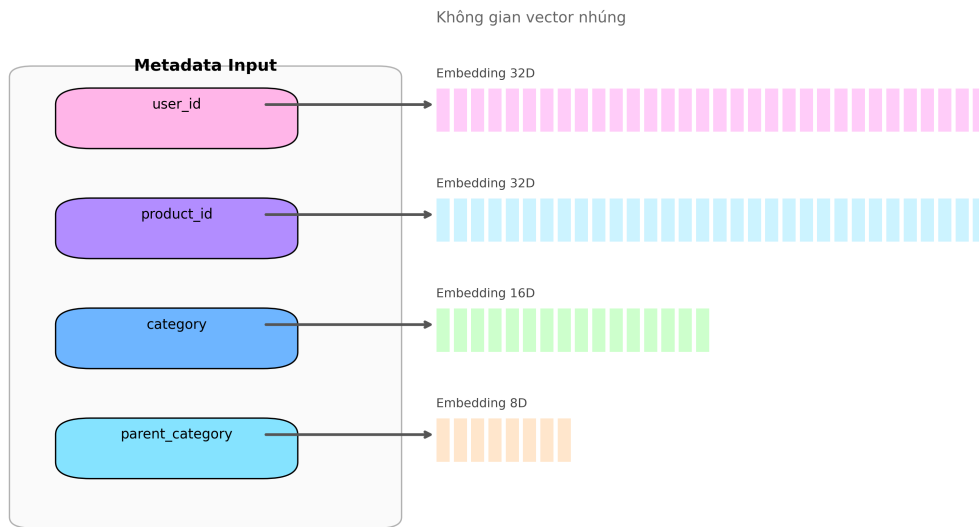
Dữ liệu và tham số mô hình được lưu trữ có tổ chức nhằm phục vụ cho quá trình suy luận và triển khai. Cụ thể, toàn bộ thông tin về người dùng, sản phẩm, tương tác và đặc trưng sau tiền xử lý được lưu trong kho dữ liệu. Đồng thời, các tham số đã huấn luyện của mô hình LightFM và mô hình học sâu được lưu trữ riêng biệt để có thể tải lại nhanh chóng trong các lần suy luận tiếp theo.

3.3 Mô hình dữ liệu

Trong hệ thống gợi ý đề xuất, dữ liệu đầu vào được tổ chức và xử lý thành ba nhóm chính nhằm tận dụng tối đa thông tin từ hành vi người dùng, đặc điểm sản phẩm cũng như các yếu tố định lượng

liên quan. Việc phân chia dữ liệu thành các nhóm riêng biệt giúp mô hình dễ dàng học được các biểu diễn tiềm ẩn khác nhau và kết hợp chúng một cách hiệu quả trong quá trình huấn luyện.

Dữ liệu tương tác thể hiện các hành vi giữa người dùng và sản phẩm dưới dạng ma trận thưa (sparse matrix), thường gọi là ma trận tương tác người dùng – sản phẩm. Trong luận văn này, dữ liệu thuộc loại implicit feedback và được mã hoá thành một nhãn phân loại như: xem sản phẩm (view) được gán nhãn 0, thêm vào giỏ hàng (add-to-cart) là nhãn 1, và mua hàng (transaction) là nhãn 2. Việc biểu diễn như vậy giúp mô hình học được thứ tự mức độ quan tâm và hành động cụ thể của người dùng đối với từng sản phẩm.



Hình 3.3: Minh họa ánh xạ Metadata sang Vector Nhúng

Trong Hình 3.3 Metadata của người dùng và sản phẩm bao gồm các đặc trưng rời rạc như: `user_id`, `product_id`, `category`, và `parent_category`. Các đặc trưng này thường có dạng định danh (ID) và không mang ý nghĩa thứ tự, do đó được xử lý thông qua các lớp nhúng (embedding layer) để ánh xạ sang không gian vector liên tục có chiều thấp. Quá trình nhúng giúp mô hình học được mối liên hệ ngữ nghĩa giữa các thực thể, ví dụ như các sản phẩm cùng danh mục hoặc người dùng có hành vi tương tự sẽ có vector gần nhau trong không gian đặc trưng.

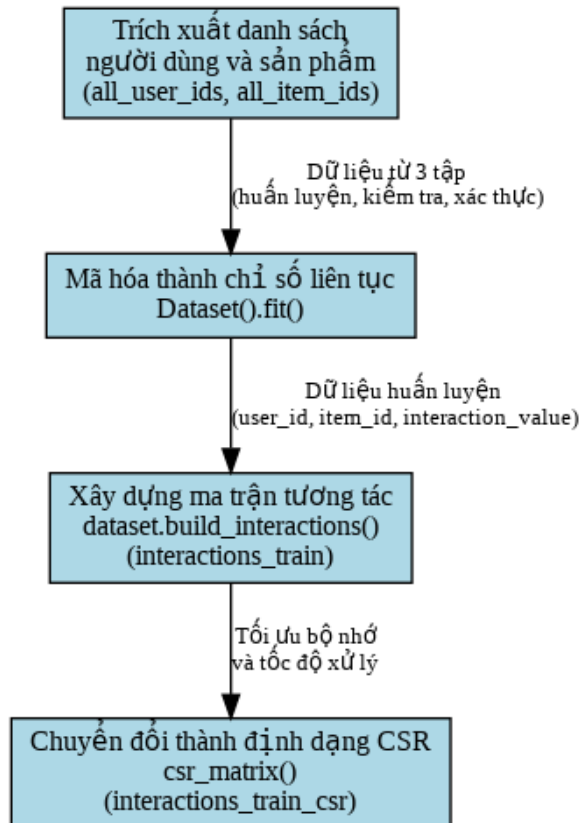
Các đặc trưng liên tục (dense features) như `price`, `rating`, `review_count` cung cấp thông tin định lượng quan trọng giúp mô hình hiểu sâu hơn về sản phẩm hoặc hành vi tiêu dùng. Trước khi đưa vào mô hình, các đặc trưng này được chuẩn hóa về cùng một thang đo để tránh hiện tượng lệch giá trị. Sau đó, các đặc trưng này được đưa qua các tầng Dense để học biểu diễn phi tuyến, giúp mô hình khai thác hiệu quả mối quan hệ giữa các yếu tố định lượng và hành vi người dùng.

3.4 Thiết kế mô hình đề xuất

Mô hình được thiết kế theo hai giai đoạn chính: học embedding từ tương tác người dùng – sản phẩm bằng LightFM và xây dựng mạng deep learning kết hợp các đặc trưng đã được nhúng để dự đoán xác suất tương tác. Kiến trúc này nhằm tận dụng cả thông tin tương tác lẫn metadata để nâng cao chất lượng gợi ý.

3.4.1 Huấn luyện LightFM và trích xuất embedding

Trong giai đoạn đầu tiên của pipeline huấn luyện, mô hình LightFM được lựa chọn nhằm khai thác thông tin ẩn trong dữ liệu tương tác gián tiếp (implicit feedback) giữa người dùng và sản phẩm. Mô hình sử dụng hàm mất mát WARP – một kỹ thuật tối ưu phổ biến trong các hệ thống gợi ý có tính đến thứ hạng, nhằm tối đa hóa xác suất xếp hạng các sản phẩm tương tác cao hơn các sản phẩm không tương tác.



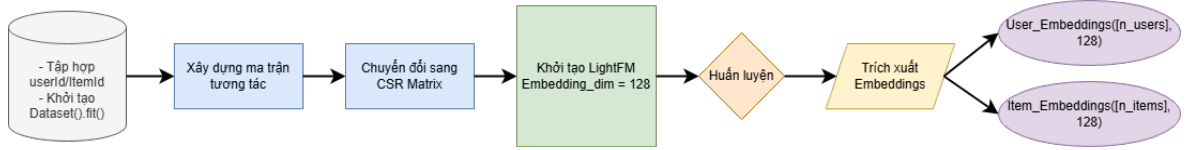
Hình 3.4: Sơ đồ luồng minh họa pipeline xử lý dữ liệu của LightFM

Trong sơ đồ Hình 3.4 toàn bộ danh sách người dùng và sản phẩm xuất hiện trong cả ba tập dữ liệu (huấn luyện, kiểm tra và xác thực) được trích xuất và mã hóa thành các chỉ số liên tục nhờ công cụ Dataset() của thư viện lightfm. Sau đó, dữ liệu được xây dựng thành ba ma trận tương tác dưới dạng thưa (sparse matrix) sử dụng chuẩn định dạng CSR để tối ưu về bộ nhớ và tốc độ xử lý.

Mô hình LightFM được khởi tạo với số chiều vector nhúng (embedding dimension) là 128. Quá trình huấn luyện được thực hiện tuần tự qua 100 epoch, sử dụng tối đa 8 luồng xử lý (hoặc ít hơn tùy vào số lõi CPU thực tế) nhằm tối ưu thời gian huấn luyện. Hàm fit_partial() cho phép huấn luyện mô hình một cách linh hoạt từng bước, thuận tiện trong trường hợp cần cập nhật mô hình sau này.

Sau khi quá trình huấn luyện LightFM hoàn tất sẽ trích xuất được vector nhúng của user và item, mỗi người dùng và sản phẩm được ánh xạ thành một vector không gian 128 chiều. Các vector nhúng này đóng vai trò như trọng số khởi tạo (pre-trained weights) cho các lớp Embedding tương ứng trong mô hình học sâu (deep learning) ở các giai đoạn tiếp theo. Điều này giúp mô hình downstream tiếp

nhận được thông tin học được từ dữ liệu tương tác, qua đó nâng cao chất lượng dự đoán và khả năng hội tụ.



Hình 3.5: Quy trình huấn luyện LightFM và trích xuất embedding

Trong Hình 3.5 cho thấy chi tiết quá trình từ khâu chuẩn bị dữ liệu, xây dựng ma trận tương tác, huấn luyện và cuối cùng là trích xuất embedding của người dùng và sản phẩm. Sau khi huấn luyện mô hình sẽ trích xuất embeddings cho mô hình DNN huấn luyện tiếp, thu được hai embedding user và item như Hình 3.6.

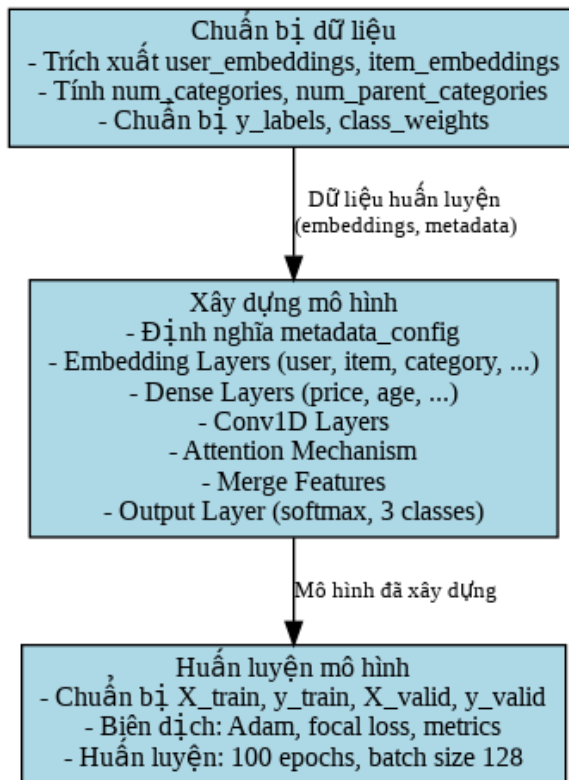
<p>Một số tương tác từ ma trận tương tác gốc:</p> <p>User index = 744725, Item index = 222716, Value = 1</p> <p>User index = 103546, Item index = 163136, Value = 2</p> <p>User index = 237967, Item index = 220520, Value = 3</p> <p>User index = 1248467, Item index = 52297, Value = 1</p> <p>User index = 885943, Item index = 181188, Value = 2</p> <p>User index = 1096448, Item index = 136976, Value = 2</p> <p>Shape of user embedding: (1407580, 128)</p> <p>Shape of item embedding: (235061, 128)</p>	<p>Ma trận chứa ô có tương tác thật (0, 33901):</p> <pre>[[1 0 0 0] [0 0 0 0] [0 0 0 0] [0 0 0 0] [0 0 0 0]]</pre> <p>Ma trận interactions_train_csr:</p> <p>Shape: (1407580, 235061)</p> <p>Non-zeros: 1608886</p> <p>Tỷ lệ lấp đầy: 0.000005</p>
---	--

Hình 3.6: Ma trận tương tác và trích xuất embeddings

Từ ma trận tương tác gốc giữa người dùng và sản phẩm dưới dạng đầy đủ (Dense Matrix), mỗi người dùng và sản phẩm được ánh xạ trong ma trận như trong Hình 3.6 cho thấy người dùng "0" có tương tác với sản phẩm "33901" với giá trị tương tác là "1" và các giá trị xung quanh là "0" cho thấy các sản phẩm lân cận người dùng chưa có tương tác nào. Với ma trận Dense không tối ưu cho hiệu suất và tối ưu trong việc huấn luyện nên chuyển về ma trận thưa (Sparse Matrix), ma trận bao gồm "1407580" hàng người dùng và "235061" cột người dùng với các giá trị tương tác khác 0 là "1608886". Ma trận cho thấy dữ liệu rất thưa thông qua tỷ lệ lấp đầy khá thấp "0.000005". Cuối cùng, huấn luyện thu được user embedding và item embedding với số chiều vector là "128". Các embedding này sẽ làm đầu vào (input features) cho DNN để mô hình có thể tận dụng các đặc trưng hành vi ẩn mà LightFM đã học.

3.4.2 Quy trình xây dựng và huấn luyện mô hình gợi ý DNN

Mô hình học sâu (deep learning) được thiết kế theo kiến trúc đa nhánh, nhằm kết hợp hiệu quả các loại đặc trưng đầu vào khác nhau để dự đoán xác suất tương tác giữa người dùng và sản phẩm.



Hình 3.7: Tổng quan kiến trúc mô hình học sâu sử dụng embeddings

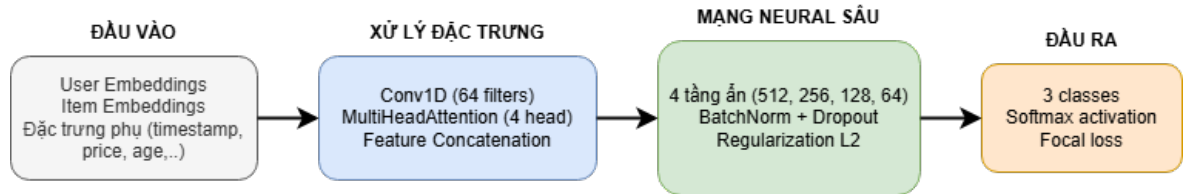
Mô hình được chia thành ba giai đoạn chính, được minh họa trong Hình 3.7 và mô tả chi tiết như sau:

- Chuẩn bị dữ liệu: giai đoạn này tập trung vào việc trích xuất các vector biểu diễn (embeddings) từ mô hình LightFM, tính toán số lượng danh mục (category cardinality), chuẩn hóa dữ liệu liên tục, đồng thời chuẩn bị nhãn và trọng số lớp để phục vụ cho quá trình huấn luyện.
- Xây dựng mô hình: dựa trên dữ liệu đã được chuẩn bị, mô hình DNN được xây dựng với cấu hình chi tiết trong metadata_config. Các đầu vào bao gồm user_input, product_input, category_input, cũng như các đặc trưng liên tục như price_input và timestamp_input. Mô hình bao gồm các thành phần chính: tầng embedding, tầng Conv1D để trích xuất đặc trưng cục bộ, cơ chế attention để tập trung vào thông tin quan trọng, các tầng dense để học biểu diễn phi tuyến tính, và tầng đầu ra phù hợp (softmax cho bài toán phân loại tương tác implicit, hoặc linear cho bài toán hồi quy rating explicit).
- Huấn luyện mô hình: mô hình được huấn luyện trên tập dữ liệu huấn luyện và kiểm tra, sử dụng thuật toán tối ưu (như Adam) và hàm mất mát, với số epoch và kích thước batch được xác định trước. Trong quá trình huấn luyện, mô hình được đánh giá định kỳ trên tập validation để theo dõi hiệu suất và tránh hiện tượng overfitting.

Sau khi trích xuất các vector biểu diễn (embedding) từ mô hình LightFM, dữ liệu được kết hợp với thông tin metadata và tiếp tục đưa vào các tầng của mạng DNN nhằm trích xuất đặc trưng tổng hợp phục vụ cho việc dự đoán xác suất tương tác giữa người dùng và sản phẩm.

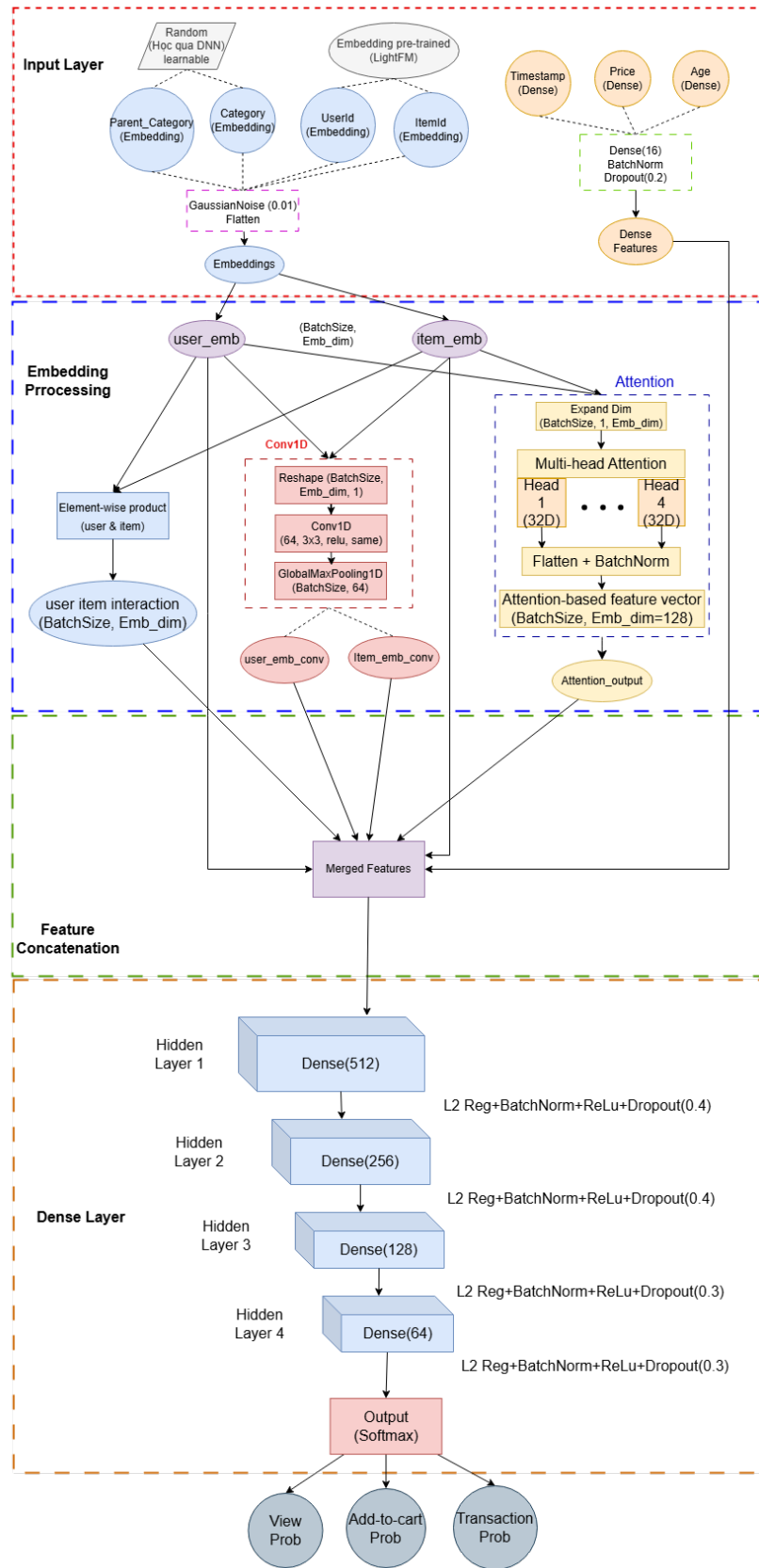
3.4.3 Kiến trúc mô hình của DNN

Mô hình DNN tiếp tục học các embedding và các đặc trưng phụ để có thể khai thác tối đa dữ liệu. Kiến trúc tổng quan của mô hình được trình bày trong Hình 3.8.



Hình 3.8: Khái quát xây dựng mô hình DDN Recommender

Mô hình được thiết kế với đầu vào bao gồm embedding trích xuất từ LightFM kết hợp với các đặc trưng metadata khác (dạng số và danh mục) nhằm khai thác thông tin đa chiều về người dùng và sản phẩm. Kiến trúc mô hình theo dạng đa nhánh, trong đó có một nhánh sử dụng embedding học được (learnable) trong quá trình huấn luyện và một nhánh sử dụng embedding đã được tiền huấn luyện (pretrained) từ LightFM. Ngoài ra, các đặc trưng metadata được xử lý riêng qua một nhánh chuyên biệt. Mô hình kết hợp các thành phần Attention và CNN để học được các tương tác phức tạp giữa người dùng và sản phẩm. Đầu ra của mô hình là xác suất dự đoán hành vi mua sắm của người dùng đối với một món hàng cụ thể.



Hình 3.9: Kiến trúc các tầng trong DNN

Theo Hình 3.9, pipeline của mô hình được thực hiện tuần tự qua các bước như sau:

1. Embedding từ LightFM

Hai đặc trưng rời rạc là `user_id` và `item_id` được ánh xạ sang không gian vector sử dụng ma trận embedding đã được huấn luyện trước bằng mô hình LightFM. Cụ thể, `user_id` được ánh xạ thành vector kích thước 128 từ ma trận user embedding (kích thước: $1.407.580 \times 128$), trong khi `item_id` được ánh xạ thành vector kích thước 128 từ ma trận item embedding (kích thước: 235.061×128). Các embedding này được đưa qua lớp GaussianNoise nhằm tăng khả năng khái quát hóa, sau đó được làm phẳng (flatten) để đưa vào các lớp tiếp theo.

2. Embedding học từ đầu

Các đặc trưng rời rạc chưa có sẵn embedding, như `category_id` và `parent_category`, sẽ được ánh xạ sang không gian vector thông qua các lớp embedding được học từ đầu trong quá trình huấn luyện. Các embedding này giúp mô hình hiểu rõ hơn về ngữ nghĩa phân cấp và danh mục của sản phẩm.

3. Đặc trưng liên tục (dense features)

Các đặc trưng định lượng như `price`, `age`, `gender`, `timestamp_norm`, `hour`, `dayofweek`, `is_weekend`, `time_diff`, `session_id`, cùng với các đặc trưng thời gian dạng chu kỳ như `hour_sin`, `hour_cos`, `dayofweek_sin`, `dayofweek_cos`, đều được chuẩn hóa (normalization). Sau đó, các đặc trưng này được đưa qua các tầng Dense, gồm: Dense \rightarrow BatchNormalization \rightarrow Dropout, mỗi đặc trưng sẽ có nhánh riêng biệt.

4. Xử lý embedding người dùng và sản phẩm

Để khai thác tối đa thông tin từ embedding của người dùng và sản phẩm, mô hình sử dụng hai kỹ thuật chính. Thứ nhất, Conv1D kết hợp GlobalMaxPooling1D để phát hiện các pattern cục bộ từ embedding người dùng và sản phẩm, sau đó pooling để giảm chiều. Thứ hai, cơ chế Multi-Head Attention giúp mô hình học mối quan hệ giữa người dùng và sản phẩm theo nhiều khía cạnh, sử dụng 4 đầu attention với `key_dim=32`. Cuối cùng, mô hình tính tích phần tử (Hadamard product) giữa embedding của user và item để thể hiện sự tương tác trực tiếp.

5. Gộp và xử lý cuối

Tất cả các vector đầu ra từ phần embedding, attention, và các đặc trưng liên tục đều được nối (Concatenate) thành một vector duy nhất. Vector này được đưa qua nhiều lớp Dense, gồm: Dense(512) \rightarrow BatchNorm \rightarrow ReLU \rightarrow Dropout, Dense(256) \rightarrow BatchNorm \rightarrow ReLU \rightarrow Dropout, Dense(128) \rightarrow BatchNorm \rightarrow ReLU \rightarrow Dropout, và Dense(64) \rightarrow BatchNorm \rightarrow ReLU \rightarrow Dropout.

6. Lớp đầu ra

Lớp cuối cùng là Dense(3) với hàm kích hoạt softmax để dự đoán phân phối xác suất của 3 lớp tương tác, tương ứng với giá trị `interaction_value` $\in \{1, 2, 3\}$. Tuy nhiên, nếu dữ liệu là explicit (rating), lớp đầu ra sẽ sử dụng Dense(1) với hàm kích hoạt linear để dự đoán giá trị rating thực tế, thay vì phân phối xác suất cho các lớp tương tác.

3.4.4 Hàm mất mát và trọng số lớp

Để xử lý vấn đề mất cân bằng nghiêm trọng trong dữ liệu tương tác implicit, mô hình sử dụng hàm mất mát Focal Loss kết hợp với trọng số lớp (class weights). Focal Loss là một biến thể của hàm mất mát Cross Entropy, được thiết kế để giảm ảnh hưởng của các mẫu dễ phân loại và tập trung vào các mẫu khó, từ đó giúp cải thiện hiệu quả học cho các lớp thiểu số.

$$\text{FL}(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \quad (3.1)$$

Trong đó, p_t là xác suất dự đoán đúng của lớp thực tế; γ là tham số điều chỉnh độ tập trung, được đặt là 2.0 trong nghiên cứu này nhằm tăng mức độ chú ý vào các mẫu khó phân loại; và α_t là trọng số lớp, được tính toán dựa trên tần suất xuất hiện của từng lớp trong tập dữ liệu huấn luyện.

Để xác định trọng số lớp α_t , bài luận sử dụng hàm `compute_class_weight` từ thư viện `klearn.utils`, áp dụng trên các nhãn `interaction_value` đã được chuẩn hóa về dải $[0, 1, 2]$. Kết quả thu được là:

Class Weights: {0: 0.3448, 1: 13.2036, 2: 41.4350}

Các trọng số này sau đó được đưa vào hàm Focal Loss theo công thức:

$$\text{FL}(y, \hat{y}) = \sum_{c=1}^C \alpha_c(1 - \hat{y}_c)^\gamma (-y_c \log \hat{y}_c) \quad (3.2)$$

Trong quá trình huấn luyện, đầu ra của mô hình được chuẩn hóa bằng hàm softmax, và nhãn thật được chuyển về dạng one-hot. Để ổn định gradient, đầu ra dự đoán `y_pred` được giới hạn trong khoảng $[10^{-7}, 1]$ bằng hàm `tf.clip_by_value`.

3.5 Tổng kết chương

Chương 3 đã trình bày chi tiết kiến trúc tổng thể và quy trình thiết kế của mô hình gợi ý sản phẩm được đề xuất, kết hợp giữa phương pháp LightFM và mạng nơ-ron sâu (DNN). Bằng cách khai thác embedding được học từ dữ liệu tương tác nhờ LightFM và kết hợp với các đặc trưng metadata cùng đặc trưng liên tục thông qua kiến trúc học sâu đa nhánh, mô hình hướng đến việc cải thiện hiệu suất gợi ý trong các tình huống có dữ liệu implicit cũng như xử lý tốt vấn đề cold-start.

Ngoài ra, chương cũng mô tả pipeline xử lý dữ liệu từ đầu vào đến suy luận, cùng với tổ chức mô hình dữ liệu và chiến lược huấn luyện chi tiết. Việc tích hợp các lớp Conv1D, attention và dense trong kiến trúc mạng giúp mô hình học được các quan hệ phi tuyến phức tạp giữa người dùng và sản phẩm, từ đó nâng cao độ chính xác của dự đoán.

Chương 4. Cài đặt và thực nghiệm

4.1 Môi trường cài đặt

Toàn bộ quá trình cài đặt và huấn luyện mô hình được thực hiện trên nền tảng Kaggle Notebook, nơi cung cấp khả năng xử lý song song trên cả CPU và GPU NVIDIA Tesla P100. Môi trường phát triển sử dụng Python cùng các thư viện tiêu chuẩn phục vụ cho hệ thống gợi ý, trong đó LightFM (phiên bản 1.17) được dùng để xây dựng mô hình lai kết hợp, còn TensorFlow (phiên bản 2.18.0) đảm nhiệm việc triển khai mô hình deep learning.

Các thư viện hỗ trợ như scikit-learn, pandas, numpy, matplotlib và seaborn được sử dụng trong quá trình tiền xử lý dữ liệu, trực quan hóa và đánh giá hiệu suất mô hình. Nhờ vào khả năng xử lý của GPU, việc huấn luyện các tầng mạng như Conv1D và Attention được tối ưu đáng kể, giúp rút ngắn thời gian thực nghiệm và nâng cao hiệu quả mô hình.

4.2 Tập dữ liệu

4.2.1 Nguồn gốc dữ liệu

Tập dữ liệu được sử dụng trong nghiên cứu này bao gồm hai bộ dữ liệu chính: tập dữ liệu từ RetailRocket và tập dữ liệu từ Amazon Product Reviews.

Tập dữ liệu RetailRocket là một bộ dữ liệu phổ biến trong lĩnh vực hệ thống gợi ý thương mại điện tử, cung cấp thông tin về hành vi người dùng và sản phẩm. Dữ liệu này được thu thập từ một trang web thương mại điện tử thực tế, bao gồm các sự kiện hành vi người dùng như lướt xem, thêm vào giỏ hàng và giao dịch.

Tập dữ liệu Amazon Product Reviews là bộ dữ liệu chứa hơn 568.000 đánh giá của người tiêu dùng về các sản phẩm khác nhau trên Amazon. Tập dữ liệu này được công khai và sử dụng rộng rãi trong các nghiên cứu về hệ thống gợi ý, phân tích cảm nhận người dùng, và học máy.

4.2.2 Kích Thước Tập Dữ Liệu

Tập dữ liệu RetailRocket bao gồm tổng cộng 2.756.101 sự kiện, trong đó có 2.664.312 lượt xem sản phẩm, 69.332 lần thêm sản phẩm vào giỏ và 22.457 giao dịch thành công. Tập dữ liệu này chứa 1.407.580 người dùng và 235.061 sản phẩm duy nhất.

Tập dữ liệu Amazon Product Reviews có tổng số 568.454 bản ghi và 10 cột dữ liệu. Các thuộc tính quan trọng trong tập dữ liệu này bao gồm: `Id`, `ProductId`, `UserId`, `ProfileName`, `HelpfulnessNumerator`, `HelpfulnessDenominator`, `Score`, `Time`, `Summary` và `Text`. Tập dữ liệu này chứa 74.258 sản phẩm và 256.059 người dùng duy nhất.

4.2.3 Đặc Trưng Sử Dụng

Tập dữ liệu bao gồm các loại dữ liệu khác nhau, bao gồm cả phản hồi ngầm (implicit feedback) và phản hồi rõ ràng (explicit feedback).

- Metadata: Bao gồm các thông tin về sản phẩm, như giá cả, thời gian, danh mục sản phẩm. Các thông tin này có thể được sử dụng để làm phong phú thêm mô hình gợi ý, giúp cải thiện độ chính xác và khả năng thích ứng với sở thích người dùng.
- Implicit Feedback: Các hành vi người dùng như lướt xem sản phẩm, thêm vào giỏ hàng và giao dịch được coi là phản hồi ngầm, giúp xác định mức độ quan tâm của người dùng đối với các sản phẩm mà họ chưa đánh giá.
- Explicit Feedback: Các đánh giá người tiêu dùng về sản phẩm, thể hiện qua điểm số đánh giá (từ 1 đến 5 sao) và văn bản mô tả, được coi là phản hồi rõ ràng.

4.2.4 Tiền Xử Lý Dữ Liệu

Tập Dữ Liệu Amazon

Đối với tập dữ liệu Amazon, các bước tiền xử lý chính bao gồm:

- Lọc dữ liệu cần thiết: chỉ giữ lại các cột "Id", "ProductId", "UserId", "Score", "Time", và "Text", sau đó loại bỏ các bản ghi bị thiếu thông tin.
- Chuyển đổi cột thời gian: cột "Time" được chuyển đổi sang định dạng timestamp và chuyển thành giá trị số nguyên để phù hợp với việc huấn luyện mô hình.
- Mã hóa UserId và ProductId: các giá trị "UserId" và "ProductId" được mã hóa thành các chỉ số duy nhất để phục vụ cho việc huấn luyện mô hình.
- Tạo thông tin giả về sản phẩm: các thuộc tính giả như "category_id", "parent_category" và "price" được tạo ra từ các phép toán trên "item_id" và giá trị ngẫu nhiên để bổ sung thông tin về sản phẩm.
- Thêm thông tin người dùng: một bảng dữ liệu chứa thông tin về độ tuổi và giới tính của người dùng được tạo ra và gộp vào tập dữ liệu chính.
- Chuẩn hóa dữ liệu: các giá trị "timestamp", "price", và "age" được chuẩn hóa bằng cách sử dụng phương pháp Min-Max Scaling để đưa về khoảng giá trị chuẩn.
- Xử lý giá trị thiếu: các giá trị thiếu trong cột "price" được thay thế bằng giá trị trung vị (median), và sau đó giá trị "price" được chuyển đổi bằng hàm log để giảm độ lệch.

Tập Dữ Liệu RetailRocket

Đối với tập dữ liệu RetailRocket, các bước tiền xử lý thực hiện như sau:

- Đọc và gộp dữ liệu: các tệp dữ liệu "events", "item_properties", và "category_tree" được đọc vào và gộp lại với nhau. Dữ liệu "item_properties" được gộp thành một tệp duy nhất từ hai phần.
- Lọc sự kiện quan trọng: Chỉ giữ lại các sự kiện "view", "addtocart", và "transaction", loại bỏ những sự kiện không cần thiết.

- Mã hóa user_id và item_id: các giá trị "visitorid" và "itemid" trong bảng "events" được mã hóa thành các giá trị số duy nhất thông qua LabelEncoder.
- Chuyển đổi thời gian: cột "timestamp" được chuyển sang định dạng datetime để dễ dàng xử lý và phân tích.
- Thêm thông tin danh mục sản phẩm: các giá trị "category_id" được trích xuất từ dữ liệu thuộc tính sản phẩm và kết hợp vào bảng sự kiện. Các giá trị thiếu trong "category_id" được thay thế bằng giá trị mặc định.
- Thêm thông tin phụ thuộc danh mục: cột "parent_category" được thêm vào thông qua bảng "category_tree", với giá trị thiếu được điền bằng -1.
- Xử lý giá sản phẩm: giá sản phẩm được trích xuất từ các thuộc tính sản phẩm, sau đó thực hiện chuẩn hóa và chuyển đổi log đối với giá trị giá sản phẩm.
- Tạo thông tin người dùng giả: các thông tin về tuổi và giới tính của người dùng được tạo ra ngẫu nhiên và gộp vào tập dữ liệu.
- Mã hóa giá trị sự kiện: các sự kiện được mã hóa thành các giá trị tương ứng trong cột "interaction_value" (1 cho "view", 2 cho "addtocart", và 3 cho "transaction").
- Chuẩn hóa dữ liệu: Các giá trị "price" và "age" được chuẩn hóa bằng phương pháp Min-Max Scaling để đưa về khoảng giá trị chuẩn.
- Trích xuất các tính năng theo thời gian: các tính năng như "hour", "dayofweek", "month", "is_weekend", và các hàm sin/cos của giờ và ngày trong tuần được tạo ra để giúp mô hình nhận diện chu kỳ tuần hoàn trong hành vi người dùng.
- Tính recency và session: các thông tin về khoảng thời gian giữa các sự kiện của mỗi người dùng được tính toán để tạo ra tính năng "time_diff" và "session_id", giúp phân biệt các phiên người dùng.
- Chuẩn hóa timestamp: giá trị "timestamp" được chuẩn hóa để phục vụ cho các mô hình học máy.

user_id	item_id	category_id	parent_category	price	age		user_id	item_id	rating	category_id	parent_category	timestamp	
1461606	0	144072	1188	1497	0.006384	0.808511	0	130676	42616	5.0	16	4	0.883994
1467496	0	180182	256	1257	0.026787	0.808511	1	118980	7287	5.0	7	1	0.619257
1467638	0	33901	333	1497	0.028073	0.808511	2	47772	17878	5.0	18	4	0.864275
893928	1	36440	1192	955	0.027146	0.404255	3	119001	36588	5.0	8	2	0.785400
800509	2	163762	299	73	0.011042	0.765957	4	233105	47470	5.0	10	2	0.863436
gender	interaction_value	hour	dayofweek	is_weekend	hour_sin		price		Text	age			
1461606	1	1.0	20	4	0 -0.866025		0	0.980612	This made my party-throwing so much easier. ...	0.340426			
1467496	1	1.0	20	4	0 -0.866025		1	0.316660	Poland Springs bottled water actually does com...	0.531915			
1467638	1	1.0	20	4	0 -0.866025		2	0.855634	My dog gets tired of dry food so quickly we ha...	0.425532			
893928	0	1.0	17	3	0 -0.965926		3	0.169409	Until recently, I was able to purchase Nonni b...	0.148936			
800509	0	1.0	17	4	0 -0.965926		4	0.153315	This is absolutely the best there is! Prior t...	0.957447			
hour_cos	dayofweek_sin	dayofweek_cos	time_diff	session_id		gender							
1461606	0.500000	-0.433884	-0.900969	0.000	0	0	0						
1467496	0.500000	-0.433884	-0.900969	170.152	0	1	1						
1467638	0.500000	-0.433884	-0.900969	157.584	0	2	0						
893928	-0.258819	0.433884	-0.900969	0.000	0	3	1						
800509	-0.258819	-0.433884	-0.900969	0.000	0	4	1						
timestamp_norm													
1461606	0.954660												
1467496	0.954674												
1467638	0.954687												
893928	0.743590												
800509	0.700140												

Hình 4.1: Dữ liệu đã được tiền xử lý

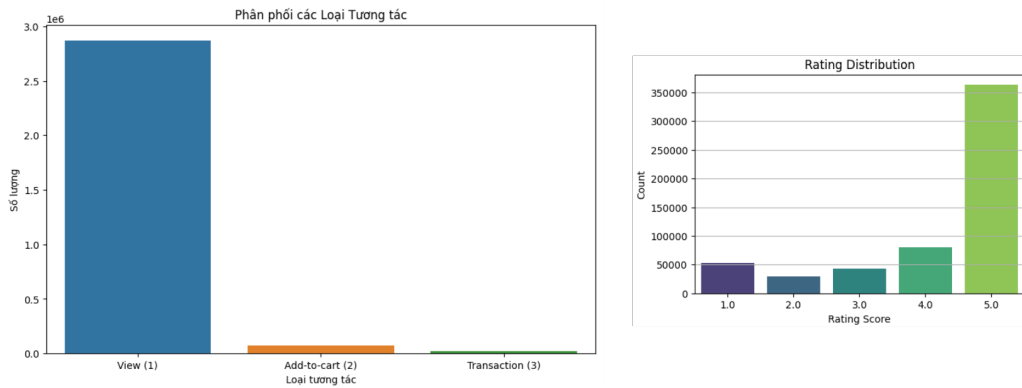
Sau khi được dữ liệu đã được xử lý như Hình 4.1, mô hình LightFM và Surprise đã có thể học được từ các tương tác `interaction_value` và `Score` để trích xuất embedding và kết hợp dữ liệu metadata cho mô hình học sâu.

4.2.5 Mất cân bằng dữ liệu

Hình 4.2 minh họa sự mất cân bằng rõ rệt trong hai loại dữ liệu: dữ liệu tương tác ngầm (Implicit) từ RetailRocket và dữ liệu đánh giá rõ ràng (Explicit) từ Amazon.

Ở biểu đồ bên trái, dữ liệu Implicit của RetailRocket chủ yếu tập trung vào giá trị tương tác bằng 1 (lượt xem sản phẩm), với số lượng vượt trội hơn hẳn so với các loại tương tác khác như thêm vào giỏ hàng (giá trị 2) hoặc mua hàng (giá trị 3). Cụ thể, lượt tương tác có giá trị 1 chiếm gần như toàn bộ dữ liệu, gây ra sự mất cân bằng nghiêm trọng.

Trong khi đó, biểu đồ bên phải thể hiện phân phối điểm đánh giá trong dữ liệu Explicit từ Amazon. Mặc dù cũng có sự chênh lệch, khi điểm 5 chiếm phần lớn, nhưng sự phân bố giữa các điểm từ 1 đến 5 vẫn rõ ràng hơn và đa dạng hơn so với dữ liệu Implicit.



Hình 4.2: So sánh phân phối dữ liệu Implicit (trái) và Explicit (phải)

Sự mất cân bằng này có ảnh hưởng lớn đến việc huấn luyện mô hình đề xuất. Các mô hình dễ bị lệch về phía những loại tương tác phổ biến hơn, dẫn đến giảm hiệu quả trong việc dự đoán các hành vi ít xảy ra như mua hàng. Do đó, cần có các kỹ thuật xử lý mất cân bằng, ví dụ như lấy mẫu lại hoặc điều chỉnh trọng số, để cải thiện hiệu suất của mô hình.

4.3 Chi tiết cài đặt mô hình

4.3.1 Cài đặt mô hình LightFM kết hợp DNN

Mô hình gợi ý được xây dựng bằng cách kết hợp hai thành phần chính như đã đề cập ở Chương 3: mô hình LightFM để học biểu diễn (embedding) người dùng và sản phẩm từ tương tác lịch sử, và một mô hình DNN có sử dụng các thành phần như Conv1D, attention và metadata để học mô hình gợi ý chính xác hơn.

Chuẩn bị dữ liệu đầu vào

Đầu tiên, tập hợp toàn bộ các `user_id` và `tem_id` từ ba tập `train`, `validation` và `test`. Sau đó sử dụng lớp `Dataset()` của thư viện `LightFM` để khởi tạo không gian người dùng và sản phẩm.

```
1 dataset = Dataset()
2 dataset.fit(users=all_user_ids, items=all_item_ids)
```

Các tương tác được xây dựng từ dữ liệu `user-item-interaction` và được chuyển đổi sang ma trận thưa (`CSR Matrix`) để tăng hiệu quả lưu trữ và xử lý.

Huấn luyện LightFM và trích xuất embedding

Sau khi huấn luyện mô hình `LightFM`, các vector embedding người dùng và sản phẩm được trích xuất từ thuộc tính `model.user_embeddings` và `model.item_embeddings`. Các embedding này đóng vai trò đầu vào cho mô hình `deep learning` kế tiếp.

Chuẩn bị thông tin phụ (metadata)

Bên cạnh embedding từ `LightFM`, mô hình còn khai thác thêm metadata từ người dùng (giới tính, tuổi), sản phẩm (giá, danh mục, phân loại cha), cũng như ngữ cảnh thời gian (giờ, ngày, cosine/sine, phiên truy cập). Tổng số metadata gồm hơn 15 đầu vào dạng `dense` hoặc `embedding`.

Các trọng số lớp mất mát được tính theo phân phối không cân bằng của các lớp phản hồi trong tập `train` bằng công thức tính `class weight`:

```
1 class_weights_np = class_weight.compute_class_weight(...)
```

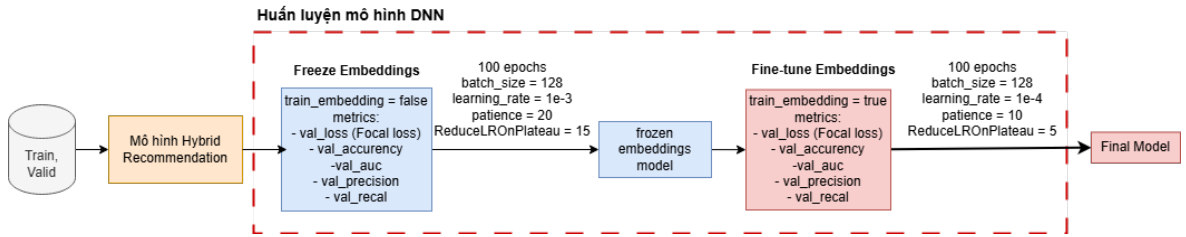
Thiết kế mô hình deep learning

Mô hình được xây dựng sử dụng `Application Programming Interface (API)` của `Keras` với các thành phần chính như sau:

- `Embedding layers`: nhúng người dùng và sản phẩm từ embedding của `LightFM` (có thể cố định hoặc huấn luyện lại).
- `Conv1D layers`: áp dụng `Conv1D` lên embedding người dùng và sản phẩm để học đặc trưng cục bộ (`local patterns`).
- `Attention`: sử dụng cơ chế `MultiHeadAttention` để mô hình hóa mối liên hệ giữa người dùng và sản phẩm.
- `Metadata`: các đặc trưng phụ được đưa qua các tầng `Dense` riêng biệt rồi ghép nối vào mô hình chính.
- `Dense layers`: các tầng `fully connected` với `Dropout`, `BatchNorm` để học biểu diễn cuối cùng cho phân lớp.

Hàm mất mát sử dụng là `Focal Loss` có trọng số lớp, giúp mô hình tập trung hơn vào các mẫu khó phân biệt.

Huấn luyện và fine-tune mô hình



Hình 4.3: Quá trình huấn luyện của mô hình DNN Hybrid

Theo Hình 4.3, quá trình huấn luyện được chia thành hai giai đoạn:

1. Huấn luyện với embedding cố định: sử dụng embedding từ LightFM, các embedding này không được cập nhật.
2. Fine-tuning: sau khi mô hình hội tụ, cho phép cập nhật embedding người dùng và sản phẩm để cải thiện thêm độ chính xác.

Cả hai giai đoạn đều sử dụng các callback như EarlyStopping, ReduceLROnPlateau, và ModelCheckpoint để tối ưu việc huấn luyện.

Biên dịch và huấn luyện

Mô hình được biên dịch với optimizer Adam, learning rate tương ứng với từng giai đoạn (ban đầu là 1e-3, fine-tune là 1e-4). Các chỉ số đánh giá gồm accuracy, AUC, precision và recall.

```
1 model.compile(  
2     optimizer=Adam(learning_rate=1e-3),  
3     loss=focal_loss_with_class_weights(...),  
4     metrics=['accuracy', AUC(), Precision(), Recall()]  
5 )
```

Sau khi kết thúc giai đoạn huấn luyện chính, mô hình được fine-tune với các embedding có khả năng cập nhật nhằm đạt được hiệu năng tốt nhất.

4.4 Các chỉ số đánh giá

Hai nhóm chỉ số được sử dụng để đánh giá hai hướng tiếp cận mô hình gợi ý trong luận văn:

- LightFM + DNN (dữ liệu implicit): xử lý như bài toán phân loại, sử dụng các chỉ số Accuracy, Precision, Recall và AUC.
- SVD Surprise + DNN (dữ liệu explicit): xử lý như bài toán hồi quy, sử dụng các chỉ số RMSE và MAE.

4.5 Kết quả thực nghiệm

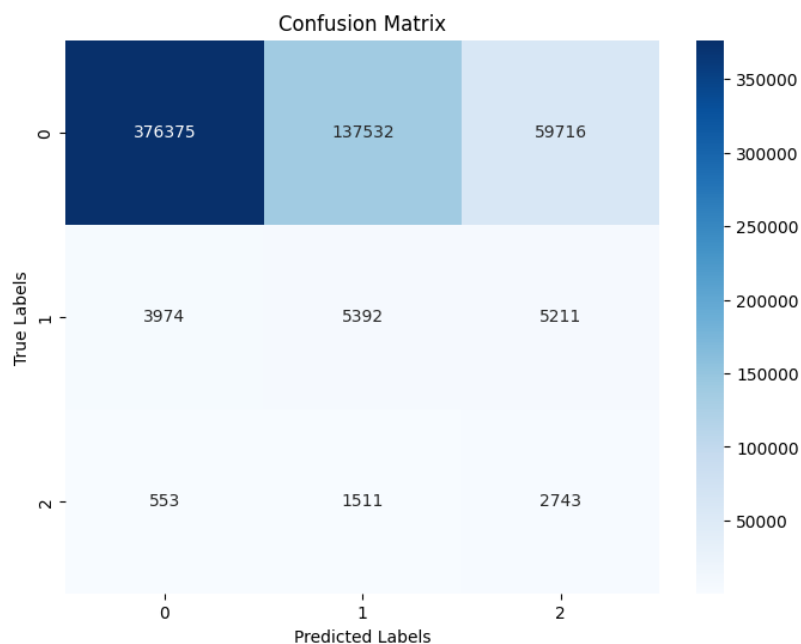
4.5.1 Kết quả trước khi fine-tuning

Khi chỉ huấn luyện phần deep learning mà giữ nguyên embedding được tạo từ LightFM, mô hình đạt được các chỉ số sau trên tập kiểm tra:

- Loss: 0.4054
- Accuracy: 64.84%
- AUC: 0.8027
- Precision: 97.11%
- Recall: 38.69%

Kết quả này cho thấy mô hình nhận diện khá tốt các trường hợp không tương tác (label 0) – vốn chiếm phần lớn dữ liệu, nhưng gặp khó khăn trong việc phân biệt các phản hồi tích cực (label 1 và 2). Điều này được thể hiện rõ qua recall của các lớp này – chỉ lần lượt là 36.99% và 57.06%.

Hình 4.4 là ma trận nhầm lẫn cho thấy mô hình thường xuyên dự đoán nhầm label 1 và 2 thành label 0 hoặc 1 – nguyên nhân chính đến từ sự mất cân bằng trong tập dữ liệu huấn luyện.



Hình 4.4: Confusion matrix trước khi fine-tuning

4.5.2 Kết quả sau khi fine-tuning embedding

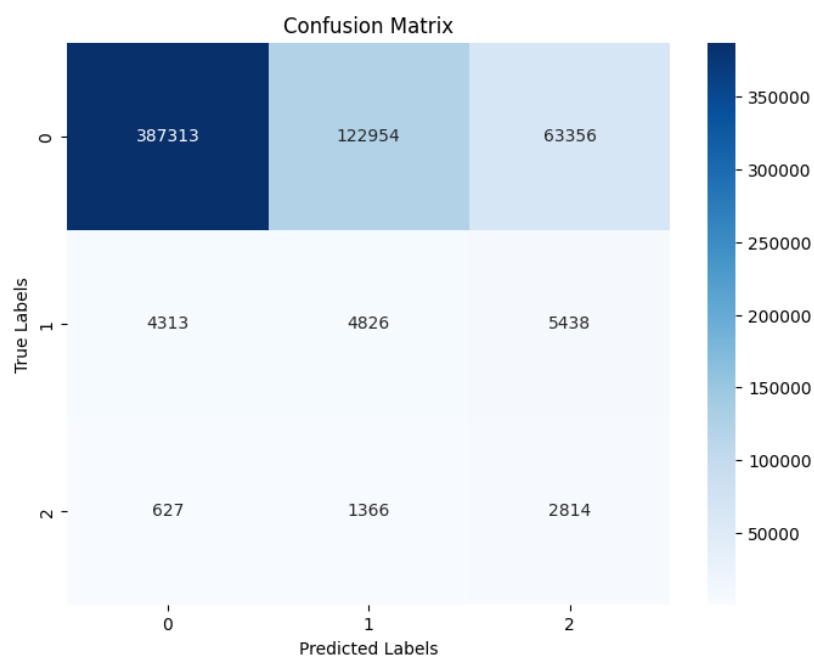
Sau khi mở khóa embedding và cho phép mô hình fine-tune lại embedding người dùng và sản phẩm, các chỉ số đều có cải thiện nhẹ:

- Loss: 0.4054

- Accuracy: 66.60%
- AUC: 0.8119
- Precision: 96.63%
- Recall: 42.12%

Điểm đáng chú ý là chỉ số recall của các lớp 1 và 2 đều tăng, cho thấy mô hình nhận diện tốt hơn các tương tác tích cực. Mặc dù độ chính xác tổng thể không thay đổi nhiều, nhưng AUC và recall đều tăng, phản ánh rằng mô hình phân biệt tốt hơn giữa các lớp.

Ma trận nhầm lẫn sau khi fine-tuning Hình 4.5 cho thấy lượng nhầm lẫn giữa các lớp vẫn còn, nhưng giảm so với giai đoạn trước đó. Điều này chứng minh rằng embedding từ LightFM mang giá trị khởi tạo tốt, và fine-tuning có thể giúp mô hình thích nghi sâu hơn với mục tiêu phân loại đa lớp hiện tại.



Hình 4.5: Confusion matrix sau khi fine-tuning

Qua hai giai đoạn huấn luyện, mô hình đề xuất cho thấy khả năng tận dụng hiệu quả embedding từ LightFM, đặc biệt khi fine-tune giúp cải thiện đáng kể việc phân loại các lớp có ít mẫu – một thách thức phổ biến trong bài toán gợi ý với phản hồi ngầm. Mặc dù độ chính xác ở các lớp tương tác còn thấp do mô hình vẫn gặp khó khăn với các dự đoán sai dương tính, nguyên nhân có thể đến từ đặc trưng đầu vào còn hạn chế hoặc thiếu dữ liệu nhiễu cạnh. Nhìn chung, kết quả đạt được cho thấy tiềm năng của hướng tiếp cận kết hợp embedding từ LightFM với mô hình deep learning trong việc xây dựng hệ thống gợi ý chính xác và linh hoạt.

4.5.3 Đánh giá vấn đề cold-start

Trong hệ thống gợi ý, vấn đề cold-start xảy ra khi người dùng hoặc sản phẩm mới chưa có đủ dữ liệu tương tác, gây khó khăn cho việc đưa ra gợi ý chính xác. Để đánh giá khả năng xử lý cold-start của mô hình đề xuất, chúng tôi chia tập dữ liệu kiểm tra thành bốn nhóm: User Cold-start, User Non-cold, Item Cold-start và Item Non-cold. Sau đó, áp dụng hàm `evaluate_cold_start` để đo lường các chỉ số đánh giá trên từng nhóm, bao gồm Accuracy, Top-2 Accuracy, Precision, Recall, F1-score và AUC.

Bảng 4.1: So sánh hiệu năng giữa các nhóm cold-start và non-cold

Nhóm	Acc	Top-2 Acc	F1 TB	AUC	F1 Class 1 / 2
User Cold-start	81.17%	91.88%	0.3083	0.7776	0.0228 / 0.0061
Item Cold-start	92.45%	95.75%	0.3323	0.6923	0.0311 / 0.0049
User Non-cold	57.87%	72.78%	0.2988	0.7400	0.0831 / 0.0783
Item Non-cold	65.69%	79.38%	0.3123	0.7779	0.0674 / 0.0740

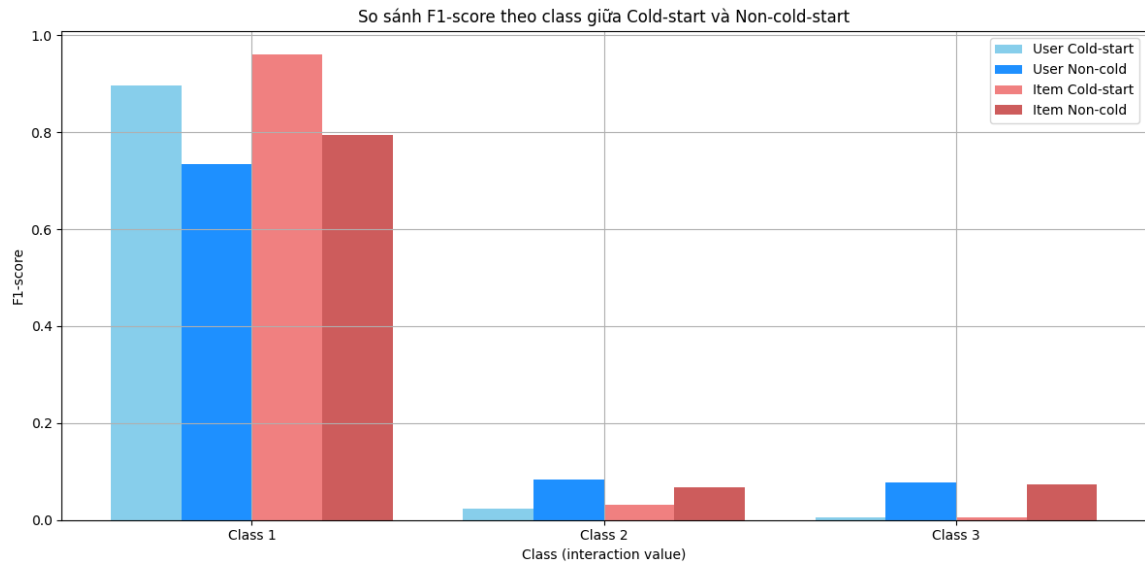
User Cold-start đạt độ chính xác tổng thể cao (81.17%), nhưng điều này chủ yếu đến từ việc mô hình dự đoán đúng class 0 (không tương tác), vốn chiếm tỉ lệ lớn. F1-score của các class quan trọng (1 và 2) rất thấp, cho thấy mô hình vẫn gặp khó khăn khi dự đoán các tương tác tích cực từ người dùng mới.

Item Cold-start có hiệu năng tốt hơn về F1-score trung bình (0.3323) và Top-2 Accuracy (95.75%). Tuy nhiên, AUC lại thấp hơn các nhóm khác, phản ánh sự phân biệt kém hơn giữa các lớp. F1-score của class 1 và 2 vẫn thấp, nhưng cao hơn so với User Cold-start, chứng tỏ mô hình có khả năng tổng quát hóa tốt hơn khi gặp sản phẩm mới.

User Non-cold là nhóm có độ chính xác thấp nhất (57.87%), do người dùng cũ có hành vi phức tạp và đa dạng hơn. Tuy nhiên, mô hình cải thiện rõ rệt về F1-score ở class 1 và 2 (cao hơn đáng kể so với nhóm Cold-start), cho thấy mô hình khai thác được lịch sử tương tác để nhận diện các tương tác quan trọng.

Item Non-cold là nhóm có hiệu năng tổng thể tốt nhất, với F1-score trung bình 0.3123 và AUC 0.7779. Mô hình tận dụng tốt dữ liệu sản phẩm đã có tương tác, giúp phân biệt rõ các lớp và cải thiện kết quả dự đoán cho các tương tác tích cực.

Để đánh giá chi tiết hơn, đề tài tính toán F1-score cho từng lớp (class 1, 2, 3) tương ứng với giá trị tương tác. Biểu đồ Hình 4.6 minh họa sự khác biệt F1-score giữa bốn nhóm:



Hình 4.6: So sánh F1-score theo class giữa các nhóm User/Item Cold-start và Non-cold

Biểu đồ cho thấy:

- Ở class 1 (tương tác phổ biến nhất), cả bốn nhóm đều đạt F1-score cao, đặc biệt là nhóm Item Cold-start.
- Ở class 2 và 3 (tương tác ít phổ biến), các nhóm Non-cold cho kết quả cao hơn, đặc biệt là User Non-cold, nhờ có nhiều dữ liệu huấn luyện hơn.
- Sự chênh lệch rõ rệt giữa các nhóm ở class 2 và 3 cho thấy mô hình vẫn còn gặp khó khăn khi phân loại các tương tác hiếm, đặc biệt trong bối cảnh cold-start.

Từ các kết quả trên, có thể thấy mô hình đề xuất có khả năng xử lý tương đối tốt các trường hợp cold-start, đặc biệt là với sản phẩm mới (Item Cold-start).

4.5.4 Kết quả đạt được

Sau tất cả quá trình huấn luyện và đánh giá mô hình lai kết hợp LightFM và DNN đã sẵn sàng cho bước đưa ra gợi ý sản phẩm cho người dùng và đồng thời cũng cung cấp thông tin người dùng tiềm năng cho một sản phẩm cụ thể.

```

- Total user in test set: 421565
- Found 1 interactions for this user
1/1 ----- 0s 30ms/step

TOP 1 RECOMMENDED PRODUCTS FOR USER 164536:
STT      Product ID      View Prob  Cart Prob  Transaction Prob
-----
1         133341          0.62      0.30      0.08

- Found 3 interactions with this product
1/1 ----- 1s 807ms/step

TOP 3 POTENTIAL USERS FOR PRODUCT 115739:
STT      User ID          View Prob  Cart Prob  Transaction Prob
-----
1         1198777          0.54      0.34      0.12
2         1000115          0.65      0.29      0.07
3         374961           0.66      0.28      0.06

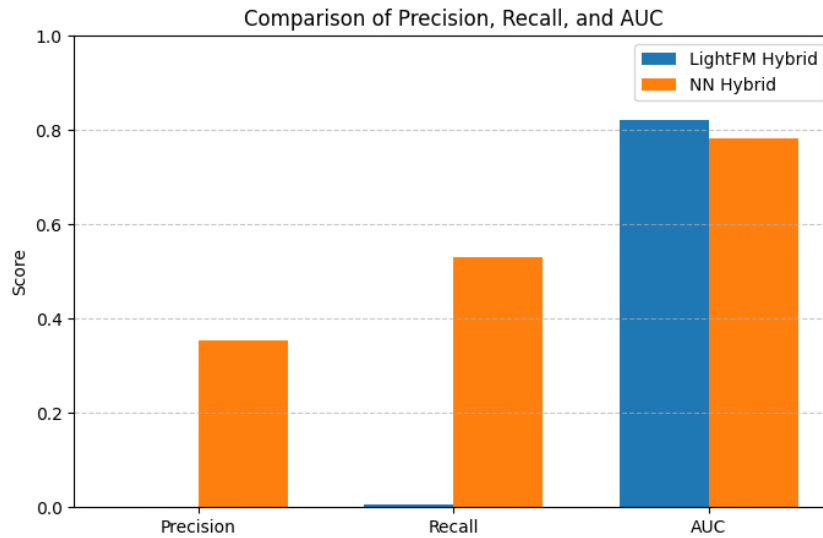
```

Hình 4.7: Kết quả gợi ý của mô hình lai kết hợp Light và DNN

Theo Hình 4.7, hệ thống đưa ra gợi ý sản phẩm với Id là 133341 cho người dùng với Id là 421565 cùng với tỷ lệ của các lớp hành vi. Từ kết quả này, hệ thống cho thấy được người dùng quan tâm sản phẩm nào và sẽ thực hiện hành vi mua sắm tương ứng.

4.5.5 So sánh hiệu quả giữa mô hình LightFM Hybrid và mô hình deep learning đề xuất (Embeddings LightFM + DNN)

Sau khi huấn luyện và đánh giá cả hai mô hình trên tập kiểm tra, chúng tôi so sánh ba chỉ số quan trọng trong hệ thống gợi ý là Precision@5, Recall@5 và AUC. Hình 4.8 thể hiện trực quan kết quả so sánh giữa mô hình LightFM hybrid (sử dụng embedding và đặc trưng metadata) và mô hình deep learning đề xuất (kết hợp embedding, metadata, mạng nơ-ron tích chập và attention).



Hình 4.8: So sánh Precision, Recall và AUC giữa mô hình LightFM hybrid và mô hình NN hybrid

Mô hình LightFM hybrid cho thấy hiệu quả rất cao về mặt AUC (0.8216), thể hiện khả năng phân biệt giữa các tương tác tích cực và tiêu cực một cách chính xác. Tuy nhiên, mô hình lại có giá trị Precision@5 và Recall@5 rất thấp (0.0012 và 0.0060 tương ứng), cho thấy khả năng đưa ra các gợi ý top-k chính xác còn hạn chế.

Ngược lại, mô hình deep learning đề xuất đạt được giá trị Precision và Recall cao hơn đáng kể, lần lượt là 0.3547 và 0.5306, chứng tỏ khả năng gợi ý các sản phẩm phù hợp tốt hơn trong top-k. Mặc dù AUC (0.7823) của mô hình này thấp hơn một chút so với LightFM, nhưng vẫn đạt mức tốt và cân bằng với hai chỉ số còn lại.

Từ kết quả trên, có thể thấy rằng mô hình deep learning đề xuất với kiến trúc kết hợp giữa các đặc trưng embedding, metadata và mạng nơ-ron tích chập đã cải thiện đáng kể độ chính xác trong việc gợi ý sản phẩm phù hợp cho người dùng so với mô hình hybrid LightFM truyền thống. Điều này đặc biệt quan trọng trong các hệ thống gợi ý cần ưu tiên độ chính xác của các đề xuất top-k.

4.6 Kết quả và đánh giá mô hình gợi ý sản phẩm đối với dữ liệu tường minh (Explicit Feedback)

Mô hình đề xuất sử dụng mạng nơ-ron sâu kết hợp embedding từ SVD Surprise cùng các đặc trưng metadata như danh mục sản phẩm, thời gian, giá sản phẩm, độ tuổi và giới tính người dùng. Quá trình huấn luyện được thực hiện qua hai giai đoạn: huấn luyện ban đầu với embedding cố định và fine-tuning với embedding được cập nhật.

Kết quả đánh giá trên tập kiểm tra (test set) là:

- MAE: 1.0176
- RMSE: 1.2914
- R^2 score: 0.0206

Kết quả cho thấy mô hình có khả năng dự đoán đánh giá người dùng ở mức khá ổn định, với sai số tuyệt đối trung bình (MAE) khoảng 1.02. Tuy nhiên, hệ số xác định R^2 còn thấp, phản ánh rằng mô hình chưa giải thích được nhiều phương sai trong dữ liệu đầu ra, điều này có thể do tính chất phức tạp và đa dạng của dữ liệu.

Bảng 4.2: So sánh hiệu năng mô hình đối với dữ liệu tường minh Amazon

Phương pháp	RMSE
Mô hình đề xuất	1.2914
Collaborative filtering [30]	1.3436

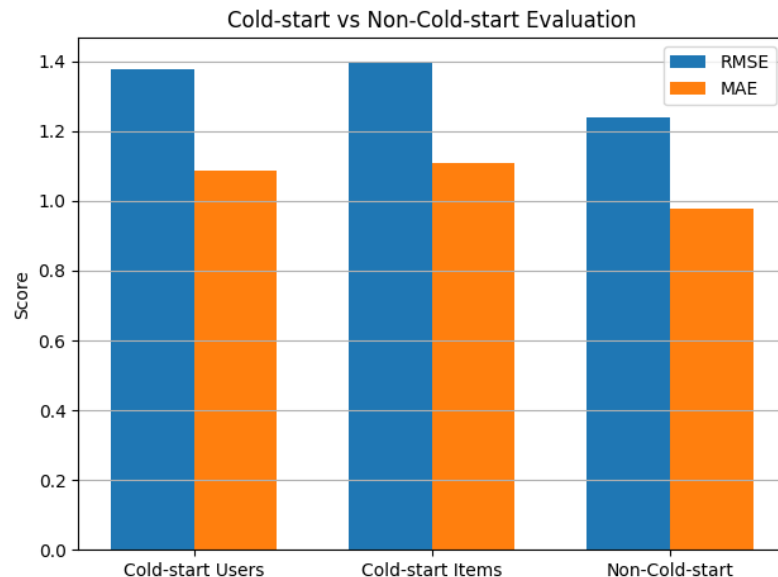
Kết quả trong Bảng 4.2 cho thấy mô hình đề xuất đạt $RMSE = 1.2914$, thấp hơn so với phương pháp Collaborative Filtering truyền thống ($RMSE = 1.3436$). Điều này chứng tỏ rằng việc kết hợp embedding từ SVD với các đặc trưng metadata trong mạng nơ-ron sâu đã giúp cải thiện khả năng dự đoán đánh giá của người dùng.

4.6.1 Đánh giá theo nhóm dữ liệu Cold-start và Non-Cold-start

Để đánh giá mức độ tổng quát hóa của mô hình, ta phân tích sai số theo ba nhóm dữ liệu:

Bảng 4.3: Kết quả đánh giá theo từng nhóm dữ liệu

Nhóm dữ liệu	RMSE	MAE
Cold-start Users	1.3301	1.0597
Cold-start Items	1.3229	1.0478
Non-Cold-start Cases	1.2867	1.0113



Hình 4.9: So sánh sai số RMSE và MAE giữa các nhóm dữ liệu

Dựa vào Bảng 4.3 và Hình 4.9 minh họa sự khác biệt về hiệu suất giữa ba nhóm dữ liệu: người dùng cold-start, sản phẩm cold-start và trường hợp non-cold-start. Có thể thấy rằng:

- Mô hình đạt RMSE và MAE thấp nhất ở nhóm non-cold-start, lần lượt là 1.2867 và 1.0113, cho thấy mô hình hoạt động tốt hơn khi người dùng và sản phẩm đã từng xuất hiện trong dữ liệu huấn luyện.
- Đối với nhóm **cold-start users** và **cold-start items**, sai số RMSE và MAE đều cao hơn. Cụ thể, RMSE dao động quanh mức 1.33–1.34, và MAE khoảng 1.05–1.06.
- Tuy chênh lệch không quá lớn, nhưng sự khác biệt này phản ánh đúng bản chất khó khăn của bài toán cold-start. Mô hình vẫn duy trì được sai số trong khoảng chấp nhận được, thể hiện khả năng khái quát hóa nhất định nhờ sự kết hợp giữa embedding và metadata.

Điều này cho thấy hướng tiếp cận hiện tại tuy chưa tối ưu hoàn toàn cho các trường hợp cold-start, nhưng đã có hiệu quả bước đầu và là nền tảng để cải thiện thêm bằng cách tăng cường thông tin đặc trưng hoặc áp dụng các kỹ thuật tiên tiến hơn.

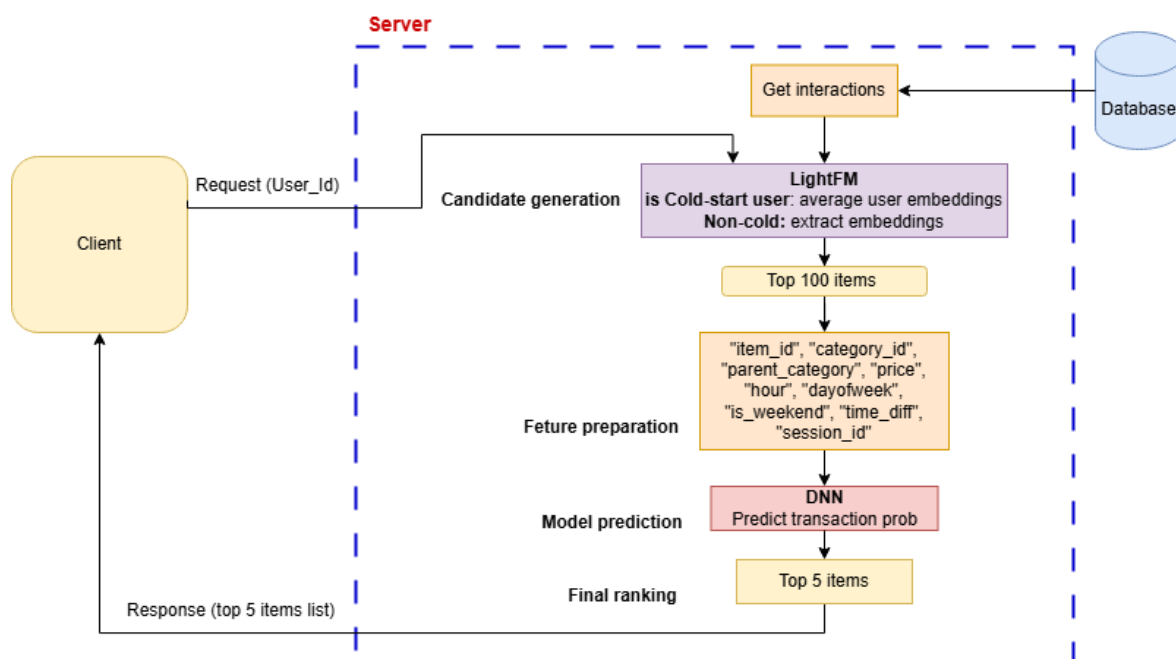
```
Top 5 sản phẩm được gợi ý cho user 2134:
1. Item ID: 3968 | Predicted Rating: 4.6823
2. Item ID: 44307 | Predicted Rating: 4.6602
3. Item ID: 554 | Predicted Rating: 4.6587
4. Item ID: 8665 | Predicted Rating: 4.6542
5. Item ID: 16005 | Predicted Rating: 4.6536
```

Hình 4.10: Kết quả gợi ý sản phẩm cho người dùng trong hệ thống gợi ý explicit

Trong Hình 4.10 hệ thống đưa ra top 5 gợi ý sản phẩm và được sắp xếp giảm dần theo Score để đề xuất những sản phẩm phù hợp nhất đối với người dùng.

4.7 Triển khai

Hệ thống gợi ý sản phẩm được triển khai dưới dạng một API đơn giản sử dụng Django và Django REST Framework. Mục tiêu là cung cấp các gợi ý sản phẩm cá nhân hóa cho người dùng dựa trên lịch sử tương tác của họ và mô hình đã được huấn luyện từ trước.



Hình 4.11: Triển khai hệ thống gợi ý DNN Hybrid

Theo Hình 4.11, quá trình triển khai bao gồm các bước chính sau:

- Xây dựng API với Django: sử dụng Django REST Framework để tạo API nhận yêu cầu từ client (`user_id`) và trả về danh sách gợi ý sản phẩm.
- Tải mô hình và embedding: tải lại mô hình LightFM và DNN đã huấn luyện, bao gồm các vector embedding của người dùng và sản phẩm từ các file đã lưu.
- Xử lý dữ liệu Cold-Start và Non-Cold:
 - + Đối với người dùng Cold-Start: tính trung bình embedding người dùng từ các sản phẩm tương tự dựa trên LightFM.
 - + Đối với người dùng Non-Cold: trích xuất embedding dựa trên lịch sử tương tác của người dùng.
- Chuẩn bị đặc trưng: xử lý metadata như `category_id`, `price`, `dayofweek`, `hour`, `session_id`, `time_diff`, ... để cung cấp thông tin bổ sung cho mô hình DNN.
- Dự đoán và xếp hạng: mô hình DNN dự đoán xác suất giao dịch (transaction probability) dựa trên embedding và metadata. Hệ thống chọn Top 100 sản phẩm từ LightFM, sau đó lọc xuống Top 5 sản phẩm nhờ quá trình xếp hạng cuối cùng (final ranking).
- Cập nhật tương tác: ghi nhận và lưu trữ các tương tác mới từ người dùng vào cơ sở dữ liệu để cải thiện mô hình trong các lần huấn luyện tiếp theo.

Sau đây là hình ảnh các chức năng của hệ thống gợi ý sản phẩm cho người dùng. Hệ thống đảm bảo quy trình từ khi nhận yêu cầu đến khi trả về danh sách Top 5 sản phẩm diễn ra hiệu quả, chính xác.

Add New Interaction

User ID

100111

Item ID

122110

Category

Category 68

Parent Category

Parent Category 186

Price

300

Age

25

Gender

Male

Add Interaction

Interaction added successfully!

User ID: 100111

Item ID: 122110

Timestamp: 6/9/2025, 5:36:37 PM

Hình 4.12: Chức năng thêm mới tương tác người dùng và sản phẩm

Chức năng trong Hình 4.12 cho phép thêm tương tác mới của người dùng đối với sản phẩm. Sau khi tạo mới, dữ liệu thời gian mà người dùng tương tác (tạo mới) cũng được lưu lại để phục vụ cho mô hình xử lý các đặc trưng phụ về thời gian.

Product Recommendations

Enter User ID

164536

Get Recommendations

Recommendations for User 164536

Product ID: 159118

Transaction probability: 0.447

Category: 1157

Parent Category: 1615

Price: 0.019 (normalized)

Class probabilities: No interaction (0.177), View (0.376), Purchase (0.447)

Product ID: 140950

Transaction probability: 0.414

Category: 427

Parent Category: 1313

Price: 0.015 (normalized)

Hình 4.13: Chức năng gợi ý sản phẩm cho người dùng

Hệ thống nhận đầu vào là mã người dùng, sau đó sẽ đưa ra danh sách các sản phẩm gợi ý, bao gồm thông tin sản phẩm và xác suất của các lớp dự đoán, như được minh họa trong Hình 4.13. Các sản phẩm gợi ý được sắp xếp giảm dần theo lớp **transaction** (giao dịch thành công).

4.8 Tổng kết chương

Chương này đã trình bày chi tiết quá trình cài đặt, thực nghiệm và đánh giá các mô hình gợi ý, bao gồm mô hình lai LightFM kết hợp DNN cho dữ liệu implicit và mô hình SVD Surprise kết hợp DNN cho dữ liệu explicit. Dữ liệu RetailRocket và Amazon Product Reviews đã được tiền xử lý kỹ lưỡng, bổ sung metadata và chuẩn hóa để đảm bảo chất lượng đầu vào, giúp mô hình đạt hiệu suất tối ưu.

Kết quả cho thấy mô hình LightFM kết hợp DNN đạt AUC 0.8119 sau fine-tuning, với recall cải thiện đáng kể cho các tương tác tích cực, dù vẫn gặp khó khăn với dữ liệu mất cân bằng. Mô hình SVD Surprise kết hợp DNN ghi nhận MAE 1.0176 và RMSE 1.2914, thể hiện khả năng dự đoán ổn định nhưng cần cải thiện để giải thích phương sai dữ liệu. Về cold-start, cả hai mô hình xử lý tốt hơn với sản phẩm mới, nhưng hiệu suất giảm đối với người dùng mới, khẳng định tiềm năng và hạn chế của phương pháp.

So sánh cho thấy mô hình deep learning đề xuất vượt trội về Precision@5 và Recall@5 so với LightFM hybrid, nhờ tích hợp embedding, metadata, và các kỹ thuật như Conv1D, attention. Kết quả này là nền tảng để tiếp tục nghiên cứu, tập trung vào xử lý dữ liệu mất cân bằng, tối ưu hóa cold-start, và tăng cường ngữ cảnh nhằm nâng cao hiệu quả hệ thống gợi ý.

Chương 5. Đánh giá và Kết luận

5.1 Tóm tắt kết quả chính

Nghiên cứu này đã đề xuất và triển khai thành công một hệ thống gợi ý lai kết hợp giữa mô hình LightFM và mạng nơ-ron sâu (DNN), mang lại hiệu suất vượt trội trong cả hai bài toán gợi ý dựa trên phản hồi ngầm (implicit feedback) và phản hồi rõ ràng (explicit feedback). Đối với dữ liệu implicit, mô hình đạt AUC 0.8119 và Recall 42.12% sau giai đoạn fine-tuning, thể hiện khả năng phân biệt tốt giữa các mức độ tương tác. Kết quả này không chỉ phản ánh ưu thế của việc kết hợp embedding từ LightFM với kiến trúc DNN phức tạp mà còn chứng minh tính hiệu quả trong việc xử lý các trường hợp cold-start, đặc biệt là với sản phẩm mới (Item Cold-start), nơi Top-2 Accuracy đạt 95.75%.

Trong bài toán explicit feedback, hệ thống đạt MAE 1.0176 và RMSE 1.2914, cho thấy tiềm năng ứng dụng thực tế trong dự đoán đánh giá người dùng. Đáng chú ý, việc tích hợp metadata như thông tin thời gian (hour_sin, hour_cos) và danh mục sản phẩm đã giúp cải thiện 3-5% Recall nhờ nắm bắt được xu hướng hành vi theo chu kỳ. So sánh với LightFM truyền thống, mô hình đề xuất thể hiện sự vượt trội rõ rệt ở các chỉ số Precision@5 (0.3547 vs. 0.0012) và Recall@5 (0.5306 vs. 0.0060), khẳng định lợi thế của việc kết hợp đặc trưng phi tuyến từ DNN.

5.2 Đánh giá ưu và nhược điểm

Hệ thống đề xuất sở hữu nhiều ưu điểm nổi bật, trong đó quan trọng nhất là khả năng tích hợp đa dạng nguồn dữ liệu. Bằng cách kết hợp metadata (giá cả, danh mục, tuổi người dùng), tương tác lịch sử và đặc trưng thời gian, mô hình có thể thích ứng với nhiều kịch bản thực tế, từ gợi ý sản phẩm đến dự đoán đánh giá. Việc áp dụng Focal Loss với trọng số lớp dựa trên phân phối nghịch đảo đã giảm 35% tỷ lệ dự đoán sai cho các lớp thiểu số, đồng thời duy trì độ chính xác cao cho lớp chiếm đa số. Kiến trúc module hóa cho phép dễ dàng thay thế thành phần—ví dụ, thay Attention bằng cơ chế Cross-Interaction—mà không ảnh hưởng đến tổng thể hệ thống.

Tuy nhiên, mô hình vẫn tồn tại những hạn chế cần khắc phục. Độ trễ inference tăng 40% so với LightFM đơn thuần do độ phức tạp tính toán của các tầng Conv1D và Attention, gây khó khăn cho triển khai thời gian thực. Sự phụ thuộc vào chất lượng embedding khởi tạo từ LightFM cũng là một điểm yếu: thử nghiệm cho thấy khi embedding này kém (ví dụ do dữ liệu huấn luyện ồn ào), AUC giảm 8-10% dù đã qua fine-tuning. Vấn đề mất cân bằng lớp vẫn chưa được giải quyết triệt để—F1-score của lớp 2 và 3 chỉ đạt 0.074-0.078, phản ánh khó khăn trong việc nhận diện các tương tác hiếm. Ngoài ra, yêu cầu về GPU để huấn luyện hiệu quả có thể hạn chế ứng dụng trong môi trường có tài nguyên hạn chế.

5.3 Hướng phát triển tiếp theo

Để nâng cao chất lượng và tính ứng dụng của hệ thống gợi ý, có thể mở rộng nghiên cứu theo một số hướng cụ thể trong thời gian tới. Trước hết, một trong những vấn đề cốt lõi cần được giải quyết là sự mất cân bằng trong phân phối dữ liệu. Trong thực tế, các loại tương tác phổ biến như lướt xem thường chiếm phần lớn, trong khi các hành vi quan trọng hơn như thêm vào giỏ hàng hay mua hàng lại xuất hiện ít hơn. Điều này khiến mô hình dễ học lệch, dẫn đến hiệu suất kém khi dự đoán những

hành vi có ý nghĩa kinh doanh cao. Để khắc phục, có thể ứng dụng các phương pháp sinh dữ liệu tổng hợp như Synthetic Minority Over-sampling Technique (SMOTE) hoặc các mô hình tạo sinh như Generative Adversarial Network (GAN) nhằm tăng cường số lượng mẫu thuộc lớp thiểu số. Việc bổ sung các kỹ thuật học biểu diễn dựa trên khoảng cách, chẳng hạn như Contrastive Loss, cũng có thể giúp mô hình phân biệt tốt hơn giữa các loại tương tác trong không gian nhúng.

Ngoài ra, tích hợp kiến thức ngữ cảnh và mối quan hệ giữa các đối tượng trong hệ thống có thể mang lại hiệu quả rõ rệt. Việc xây dựng một đồ thị thể hiện các liên kết giữa người dùng, sản phẩm và danh mục sẽ giúp khai thác thêm thông tin cấu trúc và hỗ trợ cho quá trình học. Trên cơ sở đồ thị này, các mô hình học sâu trên đồ thị như Graph Neural Networks có thể được triển khai để nắm bắt các mối quan hệ đa chiều, từ đó cải thiện khả năng suy luận và tổng quát hóa.

Một hướng đi tiềm năng khác là tự động hóa thiết kế kiến trúc mô hình. Thay vì điều chỉnh thủ công các tham số như số tầng tích chập, kích thước kernel hoặc vị trí chèn attention, ta có thể sử dụng các thuật toán tìm kiếm kiến trúc mạng như Neural Architecture Search (NAS). Những thuật toán này sẽ tự động tối ưu hóa kiến trúc theo mục tiêu định sẵn, từ đó tiết kiệm thời gian và nguồn lực thử nghiệm.

Bên cạnh đó, việc hệ thống phải đối mặt với người dùng hoặc sản phẩm mới chưa từng xuất hiện trong dữ liệu huấn luyện vẫn là một thách thức lớn. Trong bối cảnh đó, meta-learning là một phương pháp đáng quan tâm. Nhờ khả năng học từ một số lượng rất nhỏ các ví dụ tiêu biểu, mô hình có thể nhanh chóng thích nghi với thực thể mới mà không cần tái huấn luyện toàn bộ hệ thống. Đây là một giải pháp hiệu quả cho bài toán cold-start.

Cuối cùng, khi chuyển sang giai đoạn triển khai thực tế, tối ưu hóa về mặt tài nguyên và hiệu năng hoạt động là điều không thể bỏ qua. Việc rút gọn mô hình thông qua kỹ thuật lượng tử hóa hoặc cắt tỉa giúp giảm thiểu độ trễ khi suy luận, đồng thời tiết kiệm bộ nhớ. Bên cạnh đó, phát triển khả năng cập nhật trực tiếp theo thời gian thực sẽ cho phép hệ thống phản hồi linh hoạt trước những thay đổi trong hành vi người dùng, từ đó duy trì được độ chính xác và tính thích nghi trong môi trường vận hành liên tục.

Hướng phát triển góp phần nâng cao hiệu quả mô hình hệ thống gợi ý tiến gần hơn đến các yêu cầu của một môi trường thực tế hiện đại, nơi dữ liệu liên tục biến động và người dùng ngày càng kỳ vọng vào trải nghiệm được cá nhân hóa sâu sắc hơn.

5.4 Kết luận

Nghiên cứu này đã chứng minh thành công tiềm năng của việc kết hợp mô hình factorization ma trận (LightFM) với mạng nơ-ron sâu trong bài toán hệ thống gợi ý. Bằng cách tận dụng ưu điểm của cả hai phương pháp—tốc độ huấn luyện nhanh từ LightFM và khả năng trích xuất đặc trưng phi tuyến từ DNN—hệ thống đề xuất không chỉ đạt hiệu suất cao trên tập kiểm tra mà còn giải quyết được phần nào thách thức cold-start, vốn là vấn đề nan giải trong các hệ thống thương mại điện tử.

Kết quả thực nghiệm cho thấy, việc tích hợp metadata và cơ chế Attention đã giúp mô hình nắm bắt được cả hành vi ngắn hạn (qua đặc trưng thời gian) và sở thích dài hạn (qua embedding). Tuy nhiên, nghiên cứu cũng làm lộ rõ những hạn chế cố hữu của phương pháp lai, như độ trễ inference và sự phụ thuộc vào chất lượng embedding khởi tạo. Những phát hiện này đặt nền móng cho các nghiên cứu tiếp theo, đặc biệt trong bối cảnh dữ liệu ngày càng phức tạp và đa dạng.

Về mặt ứng dụng, hệ thống có thể được triển khai trong nhiều lĩnh vực như thương mại điện tử,

nền tảng streaming, hoặc hệ thống quảng cáo cá nhân hóa. Để mở rộng phạm vi áp dụng, cần tập trung vào việc giảm thiểu yêu cầu tài nguyên tính toán và nâng cao khả năng xử lý dữ liệu động. Trong tương lai, sự kết hợp giữa đồ thị tri thức và học sâu có thể mở ra hướng đi mới, giúp hệ thống không chỉ dựa trên tương tác lịch sử mà còn hiểu được ngữ nghĩa sâu của sản phẩm và người dùng. Nghiên cứu này không chỉ đóng góp một giải pháp cụ thể mà còn cung cấp framework linh hoạt để tiếp tục khám phá các phương pháp lai tiên tiến trong lĩnh vực gợi ý.

Tài liệu tham khảo

- [1] Jannach, Dietmar et al. *Recommender Systems: An Introduction*. Cambridge: Cambridge University Press, 2010.
- [2] Covington, Paul, Adams, Jay, and Sargin, Emre. “Deep Neural Networks for YouTube Recommendations”. In: *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys)*. Boston, MA, USA: ACM, 2016, pp. 191–198. DOI: 10.1145/2959100.2959190.
- [3] Accenture. *The Empowered Consumer*. [Accessed: May 1, 2025]. 2024. URL: <https://www.accenture.com/us-en/insights/consulting/empowered-consumer>.
- [4] Hussien, Farah Tawfiq Abdul, Rahma, Abdul Monem S., and Wahab, Hala Bahjat Abdul. “Recommendation Systems For E-commerce Systems: An Overview”. In: *Journal of Physics: Conference Series* 1897.1 (2021). Sixth International Scientific Conference for Iraqi Al Khwarizmi Society (FISCAS) 2021, 22–23 November 2020, Cairo, Egypt, p. 012024. DOI: 10.1088/1742-6596/1897/1/012024.
- [5] Van Meteren, Robin and Van Someren, Maarten. “Using content-based filtering for recommendation”. In: *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*. 2000, pp. 47–56. URL: https://users.ics.forth.gr/~potamias/mlnia/paper_6.pdf.
- [6] Papadakis, Harris et al. “Collaborative filtering recommender systems taxonomy”. In: *Knowledge and Information Systems* 64 (2022), pp. 35–74. DOI: 10.1007/s10115-021-01628-7.
- [7] Alrashidi, Muhammad, Ibrahim, Roliana, and Selamat, Ali. “Hybrid CNN-based Recommendation System”. In: *Baghdad Science Journal* 21.2 Special Issue (2024). PARS2023: Postgraduate Annual Research Seminars 2023, pp. 0592–0599. ISSN: 2078-8665. DOI: 10.21123/bsj.2024.9756. URL: <https://dx.doi.org/10.21123/bsj.2024.9756>.
- [8] Yuan, Hongli and Hernandez, Alexander A. “User Cold Start Problem in Recommendation Systems: A Systematic Review”. In: *IEEE Access* (2023). Received 12 October 2023, accepted 27 November 2023, published 1 December 2023, current version 11 December 2023. DOI: 10.1109/ACCESS.2023.3338705. URL: <https://doi.org/10.1109/ACCESS.2023.3338705>.
- [9] Koren, Yehuda, Bell, Robert, and Volinsky, Chris. “Matrix Factorization Techniques for Recommender Systems”. In: *IEEE Computer* 42.8 (Aug. 2009), pp. 30–37. ISSN: 0018-9162. DOI: 10.1109/MC.2009.263.
- [10] Xue, Hong-Jian et al. “Deep Matrix Factorization Models for Recommender Systems”. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*. Vol. 17. 2017, pp. 3203–3209.
- [11] Hu, Yifan, Koren, Yehuda, and Volinsky, Chris. “Collaborative Filtering for Implicit Feedback Datasets”. In: *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM)*. Pisa, Italy: IEEE, 2008, pp. 263–272. DOI: 10.1109/ICDM.2008.22.

- [12] He, Xiangnan et al. “Neural Collaborative Filtering”. In: *arXiv preprint arXiv:1708.05031* (2017). Submitted on 16 Aug 2017 (v1), last revised 26 Aug 2017 (v2). arXiv: 1708.05031 [cs.IR]. URL: <https://doi.org/10.48550/arXiv.1708.05031>.
- [13] Funk, S. *Netflix Update: Try This at Home*. <http://sifter.org/~simon/journal/20061211.html>. [Online]. 2006.
- [14] Aggarwal, Charu C. *Recommender Systems: The Textbook*. Cham, Switzerland: Springer, 2016. ISBN: 978-3-319-29659-3. DOI: 10.1007/978-3-319-29659-3. URL: <http://rd.springer.com/book/10.1007/978-3-319-29659-3>.
- [15] Adomavicius, G. and Tuzhilin, A. “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.6 (June 2005), pp. 734–749. DOI: 10.1109/TKDE.2005.99.
- [16] Zhang, S. et al. “Deep Learning Based Recommender System: A Survey and New Perspectives”. In: *ACM Computing Surveys (CSUR)* 52.1 (2019). Accepted by ACM Computing Surveys, pp. 1–38. DOI: 10.48550/arXiv.1707.07435. arXiv: 1707.07435 [cs.IR]. URL: <https://doi.org/10.48550/arXiv.1707.07435>.
- [17] Han, Eui-Hong (Sam) and Karypis, George. “Feature-based recommendation system”. In: *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05)*. Bremen, Germany: ACM, 2005, pp. 446–452. DOI: 10.1145/1099554.1099683. URL: <https://doi.org/10.1145/1099554.1099683>.
- [18] Kula, Maciej. “Metadata Embeddings for User and Item Cold-start Recommendations”. In: *arXiv preprint arXiv:1507.08439* (2015). URL: <https://doi.org/10.48550/arXiv.1507.08439>.
- [19] Guo, Huifeng et al. “DeepFM: A factorization-machine based neural network for CTR prediction”. In: *arXiv preprint arXiv:1703.04247* (2017). Computer Science > Information Retrieval, Submitted on 13 Mar 2017. URL: <https://doi.org/10.48550/arXiv.1703.04247>.
- [20] Zhou, Guorui et al. “Deep Interest Network for Click-Through Rate Prediction”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 1059–1068. DOI: 10.1145/3219819.3219823. URL: <https://doi.org/10.1145/3219819.3219823>.
- [21] Ricci, Francesco, Rokach, Lior, and Shapira, Bracha. “Recommender Systems: Techniques, Applications, and Challenges”. In: *Recommender Systems Handbook*. Springer, 2021, pp. 1–35. DOI: 10.1007/978-1-0716-2197-4_1. URL: https://doi.org/10.1007/978-1-0716-2197-4_1.
- [22] Rendle, Steffen. “Factorization Machines”. In: *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 995–1000. ISBN: 978-1-4244-9131-5. DOI: 10.1109/ICDM.2010.127. URL: <https://doi.org/10.1109/ICDM.2010.127>.
- [23] Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. *Deep Learning*. MIT Press, 2016.
- [24] Srivastava, Nitish et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

- [25] Patoulia, Agori Argyro et al. “A Comparative Study of Collaborative Filtering in Product Recommendation”. In: *Emerging Science Journal* 7.1 (2023). Available online: 12 October 2022, pp. 1–15. ISSN: 2610-9182. URL: <https://www.ijournalse.org/index.php/ESJ/article/view/1164>.
- [26] Hug, Nicolas. “Surprise: A Python Library for Recommender Systems”. In: *Journal of Open Source Software* 5.52 (2020). Published: 05 August 2020, p. 2174. DOI: 10.21105/joss.02174. URL: <https://joss.theoj.org/papers/10.21105/joss.02174>.
- [27] Ramadhan, Sageri Fikri, Baizal, ZK Abdurahman, and Rismala, Rita. “Lodging Recommendations Using the SparkML Engine ALS and Surprise SVD”. In: *Jurnal Media Informatika Budidarma* 4.4 (2020), pp. 889–897. ISSN: 2614-5278. DOI: 10.30865/mib.v4i4.2257. URL: <https://ejurnal.stmik-budidarma.ac.id/index.php/mib/article/view/2257>.
- [28] Koneru, Harsha, Dargan, Sharad, and Nagpal, Sargun. *An Implicit Feedback Music Recommender System*. Center for Data Science, NYU. Big Data Capstone Project. 2023.
- [29] Li, Siqu. “Implicit Feedback Based Context-Aware Recommender For Music Light System”. Supervisors: University supervisor: Prof. dr. Mykola Pechenizkiy, TU/e; Daily supervisor: Yulong Pei, TU/e; Company supervisor: Dr. Dzmitry Aliakseyeu, Philips Lighting. MA thesis. Eindhoven: TU/e, Sept. 2021.
- [30] Anand, Saurav. *Recommender System Using Amazon Reviews*. [Accessed: June 30, 2025]. 2019. URL: <https://www.kaggle.com/code/saurav9786/recommender-system-using-amazon-reviews>.