Mục lục

Nghiên cứu về các model trong Carskit , DeepCarskit

Huy Init

Ngày 29 tháng 8 năm 2024

Tóm tắt nội dung

Tóm tắt ngắn gọn về mục tiêu, phương pháp, và kết quả của nghiên cứu. Đây là phần cung cấp cái nhìn tổng quan về bài viết cho người đọc.

Keywords: Context-Aware Recommendation Systems , Deep Learning, Neural Collaborative Filtering (NeuCF) , Matrix Factorization (MF) , Autoencoder (AE) , Factorization Machine (FM)

1 Giới thiệu

Bài báo này giới thiệu ba cách tiếp cận chính để tích hợp thông tin ngữ cảnh vào quy trình gợi ý: lọc trước ngữ cảnh (contextual pre-filtering), lọc sau ngữ cảnh (contextual post-filtering), và mô hình hóa ngữ cảnh (contextual modeling). Mỗi cách tiếp cận đều có các phương pháp và thuật toán riêng để cải thiện độ chính xác của gợi ý bằng cách xem xét thông tin ngữ cảnh một cách trực tiếp hoặc gián tiếp.

1.1 Giải thích một số thuật ngữ liên quan

Khái niệm về Hệ thống gợi ý dựa trên ngữ cảnh (Context-aware Recommender Systems - CARS).

Hệ thống gợi ý truyền thống (Traditional RS) chỉ xem xét hai yếu tố là Người dùng (Users) và Sản phẩm (Items), sau đó đưa ra Đánh giá (Ratings).

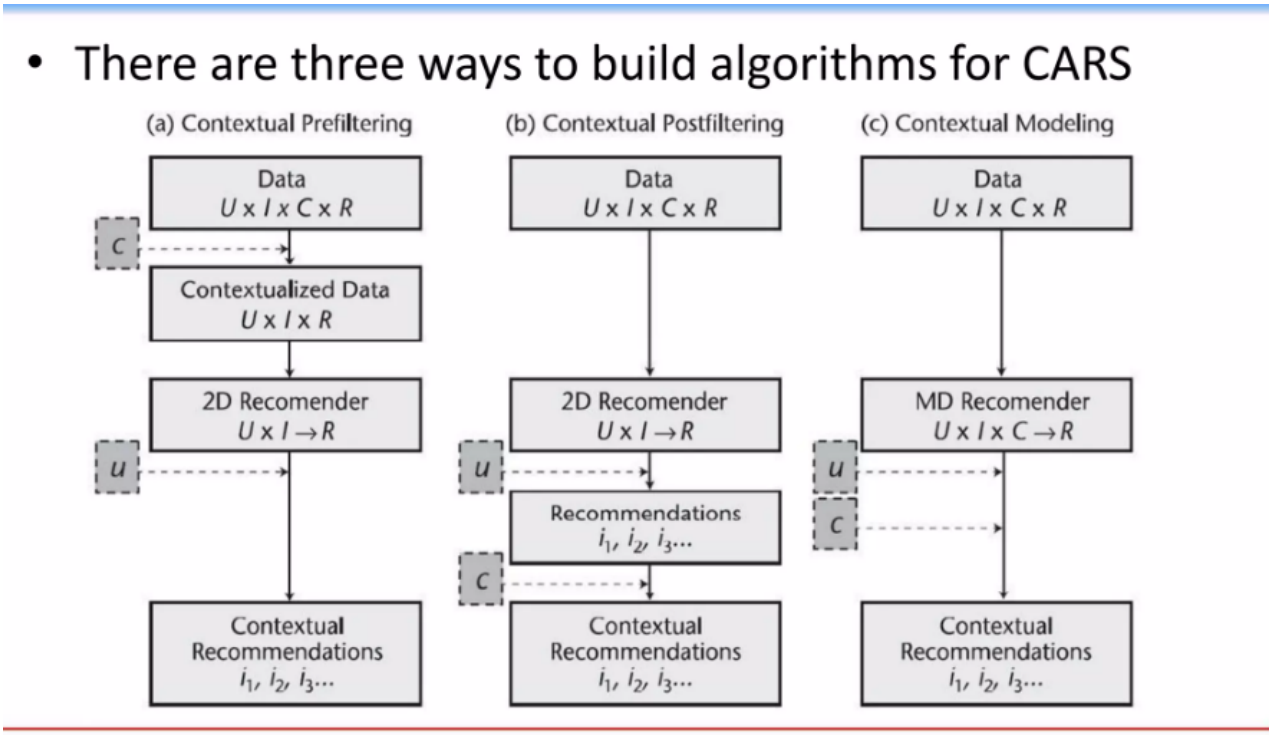
Hệ thống gợi ý dựa trên ngữ cảnh (Contextual RS) thêm một yếu tố nữa vào mô hình, đó là Ngữ cảnh (Contexts). Hệ thống này sẽ xem xét mối quan hệ giữa Người dùng, Sản phẩm, và Ngữ cảnh để đưa ra Đánh giá.

1.2 Contextual Prefiltering (Lọc trước ngữ cảnh)

Dữ liệu đầu vào: Dữ liệu bao gồm người dùng (User), sản phẩm (Item), ngữ cảnh (Context), và đánh giá (Rating), được ký hiệu là $U \times I \times C \times R$.

Quá trình xử lý: Trước khi thực hiện gợi ý, dữ liệu sẽ được lọc theo ngữ cảnh cụ thể C để tạo ra dữ liệu đã được ngữ cảnh hóa (Contextualized Data) gồm $U \times I \times R$. Sau đó, hệ thống gợi ý truyền thống 2D $U \times I \to R$ sẽ được áp dụng trên dữ liệu đã được lọc.

Kết quả: Tạo ra các gợi ý ngữ cảnh $i_1, i_2, i_3, \ldots$.



Hình 1: Biểu đồ mô tả kết quả

1.3 Contextual Postfiltering (Lọc sau ngữ cảnh)

Dữ liệu đầu vào: Tương tự như phương pháp lọc trước ngữ cảnh, dữ liệu đầu vào cũng bao gồm $U \times I \times C \times R$.

Quá trình xử lý: Hệ thống gợi ý 2D $U \times I \to R$ trước tiên sẽ được áp dụng trực tiếp lên toàn bộ dữ liệu để tạo ra danh sách gợi ý ban đầu $i_1, i_2, i_3, \ldots$. Sau đó, danh sách gợi ý này sẽ được điều chỉnh dựa trên ngữ cảnh C để tạo ra các gợi ý ngữ cảnh cuối cùng.

Kết quả: Tạo ra các gợi ý ngữ cảnh $i_1, i_2, i_3, \ldots$ đã được điều chỉnh sau khi gợi ý.

1.4 Contextual Modeling (Mô hình hóa ngữ cảnh)

Dữ liệu đầu vào: Cũng tương tự như hai phương pháp trên, dữ liệu đầu vào là $U \times I \times C \times R$.

Quá trình xử lý: Thay vì sử dụng hệ thống gợi ý 2D như hai phương pháp trên, phương pháp này sử dụng hệ thống gợi ý đa chiều (Multi-dimensional recommender) $U \times I \times C \to R$, trong đó ngữ cảnh C được tích hợp trực tiếp vào mô hình dự đoán.

Kết quả: Tạo ra các gợi ý ngữ cảnh $i_1, i_2, i_3, \ldots$.

1.5 So sánh giữa ba phương pháp

Contextual Prefiltering: Sử dụng ngữ cảnh để lọc dữ liệu trước khi áp dụng hệ thống gợi ý. Phương pháp này có thể dễ dàng triển khai và tận dụng được các thuật toán gợi ý hiện có, nhưng có thể gặp khó khăn nếu ngữ cảnh quá phức tạp hoặc không đầy đủ dữ liệu.

Contextual Postfiltering: Không sử dụng ngữ cảnh trong quá trình gợi ý ban đầu, mà chỉ điều chỉnh kết quả sau đó. Phương pháp này có thể linh hoạt hơn trong việc xử lý nhiều loại ngữ cảnh khác

nhau nhưng có thể không tận dụng được hết thông tin ngữ cảnh ngay từ đầu.

Contextual Modeling: Kết hợp ngữ cảnh trực tiếp vào mô hình gợi ý. Đây là cách tiếp cận phức tạp và yêu cầu phát triển các mô hình mới, nhưng có thể mang lại kết quả gợi ý tốt nhất do việc sử dụng ngữ cảnh một cách toàn diện.

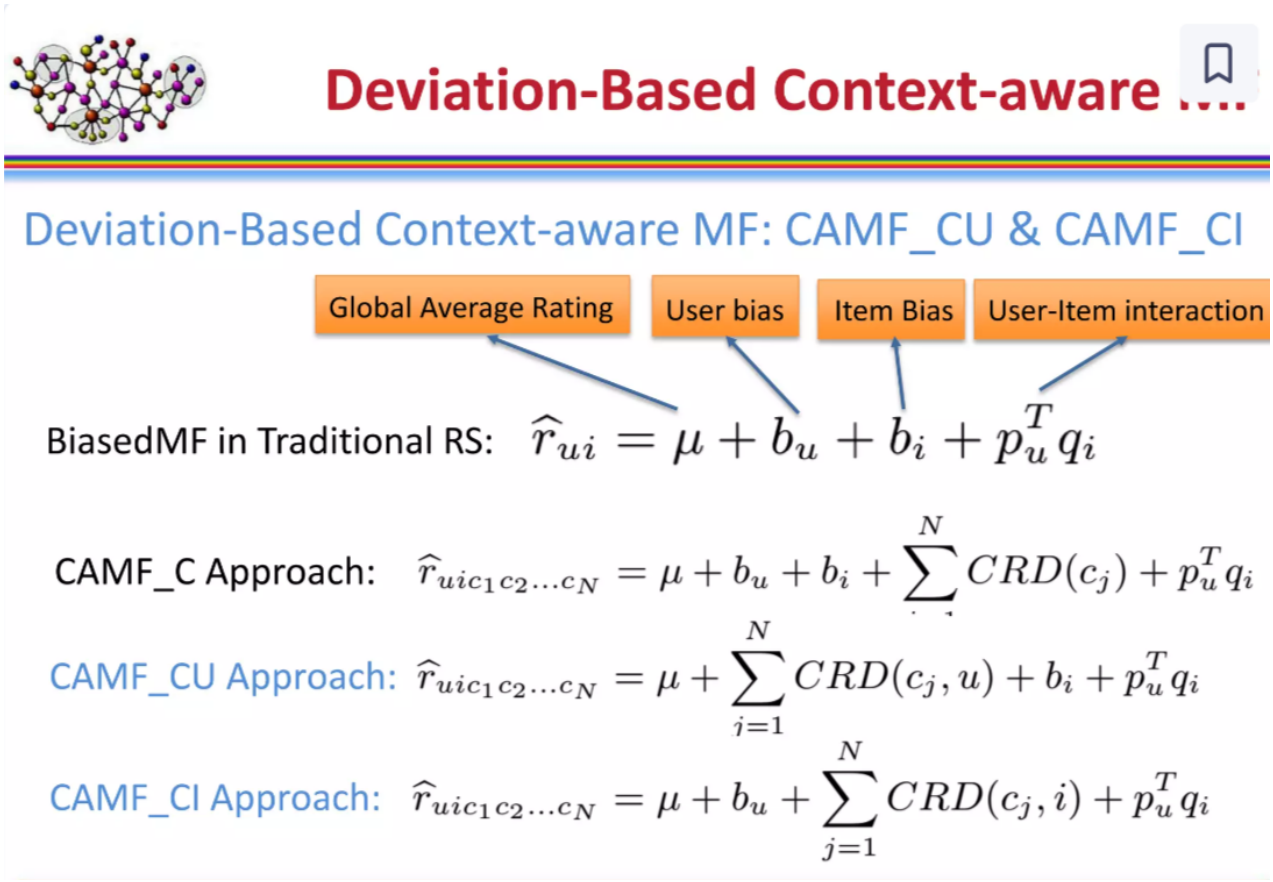
Mỗi phương pháp có ưu và nhược điểm riêng, tùy thuộc vào tình huống và loại dữ liệu mà bạn có thể chọn phương pháp phù hợp nhất.

‘‘‘

2 CARSKit - Các phương pháp chưa ứng dụng Deep learning

2.1 Contextual Modeling (Mô hình hóa ngữ cảnh)

2.1.1 Tổng quan



Hình 2: Biểu đồ mô tả kết quả

https://arxiv.org/pdf/1511.03780 Các thuật toán được sử dụng trong mô hình hóa ngữ cảnh (Contextual Modeling) trong hệ thống gợi ý dựa trên ngữ cảnh (CARS). Dưới đây là danh sách các thuật toán cùng với năm xuất hiện của chúng:

Tensor Factorization (2010): Đây là một phương pháp sử dụng phép phân tích tensor để xử lý dữ liệu đa chiều. Tensor factorization cho phép xử lý đồng thời nhiều ngữ cảnh khác nhau, làm tăng độ chính xác của các dự đoán trong hệ thống gợi ý.

Factorization Machines (2011): Đây là một thuật toán mạnh mẽ cho các bài toán dự đoán với dữ liệu có tương tác cao. Factorization Machines có thể xử lý cả dữ liệu sparse (thưa) và dense (dày đặc), và nó có thể mô hình hóa sự tương tác giữa các yếu tố khác nhau (như người dùng, sản phẩm, ngữ cảnh).

Deviation-Based Context-aware Matrix Factorization (2011): Phương pháp này mở rộng ma trận factorization truyền thống bằng cách tính đến các sự lệch (deviations) trong các yếu tố ngữ cảnh. Điều này cho phép mô hình dự đoán chính xác hơn bằng cách điều chỉnh các giá trị dựa trên ngữ cảnh cụ thể.

Deviation-Based Contextual Sparse Linear Method (2014): Phương pháp này sử dụng ma trận sparse (thưa) và tính đến sự lệch của các yếu tố ngữ cảnh để tạo ra các mô hình tuyến tính, phù hợp cho dữ liệu có độ phân tán cao và ít thông tin.

Similarity-Based Context-aware Matrix Factorization (2015): Phương pháp này sử dụng sự tương đồng giữa các ngữ cảnh để cải thiện việc dự đoán trong quá trình factorization ma trận. Nó giúp tận dụng tốt hơn các thông tin liên quan giữa các ngữ cảnh khác nhau.

Similarity-Based Contextual Sparse Linear Method (2015): Đây là sự kết hợp giữa phương pháp tuyến tính thưa và sự tương đồng ngữ cảnh, giúp cải thiện việc dự đoán trong các trường hợp dữ liệu ngữ cảnh có sự tương đồng cao nhưng không đủ thông tin.

2.1.2 Contextual Rating Deviation (CRD)

Contextual Rating Deviation (CRD) trong phương pháp phân tích ma trận dựa trên ngữ cảnh (CAMF)

*Contextual Rating Deviation (CRD) là gì? CRD là độ lệch đánh giá của người dùng dựa trên các ngữ cảnh cụ thể. Nó biểu thị sự thay đổi trong đánh giá của người dùng khi ngữ cảnh thay đổi.

*Ví dụ cụ thể trong hình:

- **Ngữ cảnh c1:** Được mô tả bởi hai chiều ngữ cảnh:
  - **D1: Time** (Weekday): Ngày trong tuần
  - **D2: Location** (Home): Ở nhà

  CRD(1) = 0.5 nghĩa là điểm đánh giá của người dùng vào các ngày trong tuần (Weekday) thường cao hơn điểm đánh giá vào cuối tuần (Weekend) với mức chênh lệch là 0.5 điểm.

- **Ngữ cảnh c2:** Được mô tả bởi hai chiều ngữ cảnh:
  - **D1: Time** (Weekend): Cuối tuần
  - **D2: Location** (Cinema): Rạp chiếu phim

  CRD(2) = -0.1 nghĩa là điểm đánh giá của người dùng khi ở rạp chiếu phim (Cinema) thường thấp hơn khi ở nhà (Home) với mức chênh lệch là 0.1 điểm.

*Phân tích chi tiết: CRD cho thấy rằng ngữ cảnh có thể ảnh hưởng đến cách người dùng đánh giá một sản phẩm hoặc dịch vụ. Ví dụ, người dùng có thể đánh giá cao hơn vào những ngày trong tuần so với cuối tuần, hoặc khi ở nhà thì họ có thể đánh giá cao hơn so với khi ở rạp chiếu phim.

- **CRD(1) = 0.5:** Nếu một người dùng thường đánh giá cao hơn vào các ngày trong tuần, điều này có thể do trạng thái tâm lý của họ khác nhau giữa ngày trong tuần và cuối tuần, ảnh hưởng đến điểm số đánh giá.

- **CRD(2) = -0.1:** Nếu người dùng đánh giá thấp hơn khi ở rạp chiếu phim, có thể do môi trường hoặc trải nghiệm tại rạp chiếu phim không tốt bằng khi họ ở nhà.

Tóm lại, CRD giúp hệ thống gợi ý hiểu rõ hơn về sự thay đổi trong hành vi đánh giá của người dùng dựa trên các yếu tố ngữ cảnh khác nhau, từ đó cải thiện độ chính xác của các dự đoán và gợi ý.

2.1.3 Deviation-Based Context-aware Matrix Factorization (CAMF_C)

Hình ảnh này minh họa công thức của phương pháp phân tích ma trận với sai lệch dựa trên ngữ cảnh (Deviation-Based Context-aware Matrix Factorization), cụ thể là CAMF_C.

*1. Công thức truyền thống BiasedMF (Biased Matrix Factorization): **Công thức:**

$$\hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p}_u^T \mathbf{q}_i$$

- μ: Giá trị đánh giá trung bình toàn cầu (Global Average Rating).
- b_u: Sai lệch người dùng (User bias), biểu thị xu hướng của người dùng khi đánh giá.
- b_i: Sai lệch của sản phẩm (Item bias), biểu thị mức độ ưa thích tổng quát của một sản phẩm.
- $\mathbf{p}_u^T \mathbf{q}_i$: Tương tác giữa người dùng và sản phẩm (User-Item interaction). Nó được tính dựa trên sự tương tác giữa vector ẩn của người dùng $\mathbf{p}_u$ và vector ẩn của sản phẩm $\mathbf{q}_i$.

Công thức này dự đoán điểm đánh giá $\hat{r}_{ui}$ dựa trên trung bình toàn cầu và sự sai lệch của người dùng cũng như sản phẩm, cùng với sự tương tác giữa người dùng và sản phẩm.

*2. Công thức CAMF_C (Context-aware Matrix Factorization with Contextual Deviations): **Công thức:**

$$\hat{r}_{uic_1c_2...c_N} = \mu + b_u + b_i + \sum_{j=1}^{N} CRD(c_j) + \mathbf{p}_u^T \mathbf{q}_i$$

- μ: Vẫn là giá trị đánh giá trung bình toàn cầu (Global Average Rating).
- b_u: Sai lệch của người dùng (User bias).
- b_i: Sai lệch của sản phẩm (Item bias).
- $\sum_{j=1}^{N} CRD(c_j)$: Tổng các sai lệch dựa trên ngữ cảnh (Contextual Rating Deviations). Ở đây, $CRD(c_j)$ biểu thị sự lệch trong đánh giá dựa trên một ngữ cảnh cụ thể c_j. Ví dụ, sự khác biệt trong đánh giá có thể dựa trên thời gian, địa điểm, hoặc bất kỳ ngữ cảnh nào khác.
- $\mathbf{p}_u^T \mathbf{q}_i$: Tương tác giữa người dùng và sản phẩm (User-Item interaction) giống như trong công thức truyền thống.

*Giải thích chi tiết: **Ngữ cảnh (Context):** Phương pháp CAMF_C mở rộng công thức truyền thống bằng cách thêm một thành phần mới $\sum_{j=1}^{N} CRD(c_j)$, biểu thị tổng các sai lệch dựa trên các yếu tố ngữ cảnh cụ thể. Điều này có nghĩa là điểm dự đoán sẽ không chỉ phụ thuộc vào người dùng và sản phẩm mà còn phụ thuộc vào các ngữ cảnh khác nhau trong đó sự tương tác xảy ra.

Sai lệch ngữ cảnh (Contextual Deviation): Thành phần này được sử dụng để điều chỉnh dự đoán dựa trên các điều kiện ngữ cảnh cụ thể. Nếu ngữ cảnh thay đổi, giá trị của $CRD(c_j)$ sẽ điều chỉnh dự đoán để phù hợp với tình huống mới, làm cho dự đoán trở nên chính xác hơn.

Tóm lại, CAMF_C nâng cao công thức dự đoán truyền thống bằng cách tính đến ảnh hưởng của ngữ cảnh, giúp dự đoán kết quả chính xác hơn trong các tình huống thực tế.

2.1.4 Hai biến thể của phương pháp CAMF: CAMF_CU và CAMF_CI

Hình ảnh này trình bày hai biến thể của phương pháp CAMF (Context-Aware Matrix Factorization) là **CAMF_CU** và **CAMF_CI**. Cả hai đều mở rộng công thức cơ bản bằng cách thêm các thành phần liên quan đến ngữ cảnh (CRD - Contextual Rating Deviation).

*1. CAMF_CU Approach **Công thức:**

$$\hat{r}_{uic_1c_2...c_N} = \mu + \sum_{j=1}^{N} CRD(c_j, u) + b_i + \mathbf{p}_u^T \mathbf{q}_i$$

- μ: Giá trị đánh giá trung bình toàn cầu.
- $\sum_{j=1}^{N} CRD(c_j, u)$: Tổng các sai lệch ngữ cảnh nhưng liên quan đến người dùng u. Điều này có nghĩa là sai lệch ngữ cảnh được điều chỉnh dựa trên người dùng cụ thể.
- b_i: Sai lệch của sản phẩm.
- $\mathbf{p}_u^T \mathbf{q}_i$: Tương tác giữa người dùng và sản phẩm.

Phân tích: CAMF_CU tập trung vào ảnh hưởng của ngữ cảnh lên người dùng. Ví dụ, cách một người dùng đánh giá một sản phẩm có thể thay đổi tùy thuộc vào thời gian, địa điểm, và các yếu tố ngữ cảnh khác.

*2. CAMF_CI Approach **Công thức:**

$$\hat{r}_{uic_1c_2...c_N} = \mu + b_u + \sum_{j=1}^{N} CRD(c_j, i) + \mathbf{p}_u^T \mathbf{q}_i$$

- μ: Giá trị đánh giá trung bình toàn cầu.
- b_u: Sai lệch của người dùng.
- $\sum_{j=1}^{N} CRD(c_j, i)$: Tổng các sai lệch ngữ cảnh nhưng liên quan đến sản phẩm i. Điều này có nghĩa là sai lệch ngữ cảnh được điều chỉnh dựa trên sản phẩm cụ thể.
- $\mathbf{p}_u^T \mathbf{q}_i$: Tương tác giữa người dùng và sản phẩm.

Phân tích: CAMF_CI tập trung vào ảnh hưởng của ngữ cảnh lên sản phẩm. Điều này có nghĩa là cách một sản phẩm được đánh giá có thể thay đổi dựa trên ngữ cảnh cụ thể, như thời gian hoặc địa điểm mà sản phẩm được sử dụng.

*So sánh giữa CAMF_CU và CAMF_CI

- **CAMF_CU**: Điều chỉnh các sai lệch ngữ cảnh dựa trên người dùng, cho phép mô hình xem xét tác động của ngữ cảnh cụ thể đến hành vi đánh giá của từng người dùng.
- **CAMF_CI**: Điều chỉnh các sai lệch ngữ cảnh dựa trên sản phẩm, cho phép mô hình xem xét tác động của ngữ cảnh cụ thể đến cách mỗi sản phẩm được đánh giá.

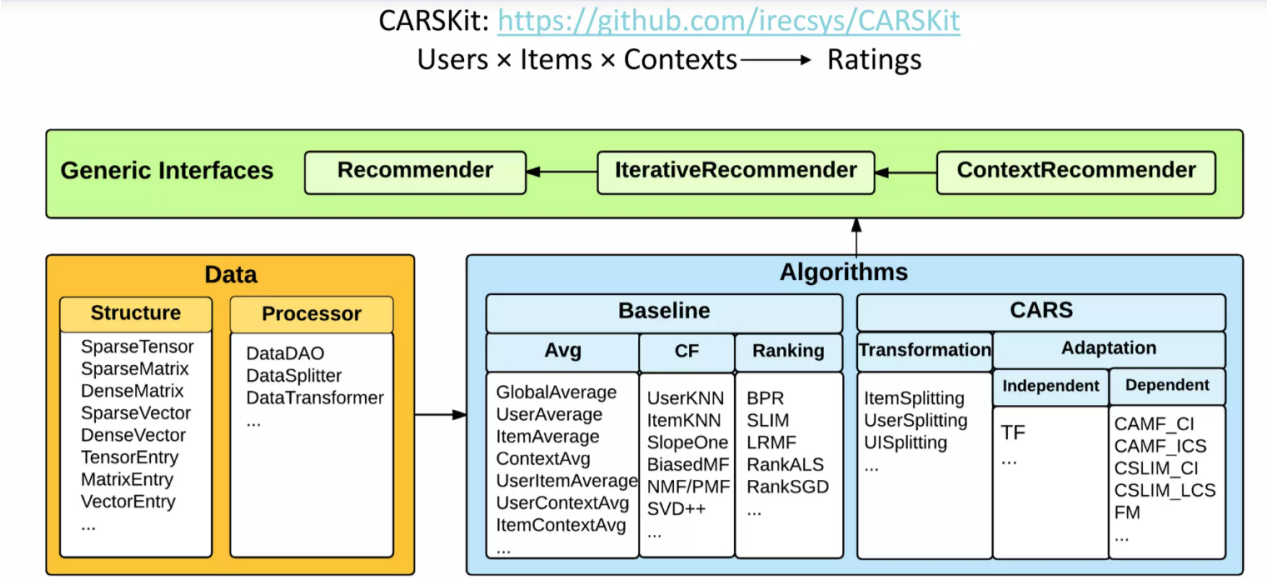
Tóm lại:

- **CAMF_CU** phù hợp trong các tình huống mà ngữ cảnh tác động mạnh đến cách người dùng đưa ra đánh giá.
- **CAMF_CI** phù hợp hơn khi ngữ cảnh chủ yếu ảnh hưởng đến cách sản phẩm được người dùng đánh giá.

Cả hai phương pháp đều bổ sung yếu tố ngữ cảnh vào mô hình, nhưng với trọng tâm khác nhau, giúp mô hình hóa các ảnh hưởng của ngữ cảnh một cách chi tiết và phù hợp hơn với từng tình huống cụ thể.

2.2 CARSKit

CARSKit là một bộ công cụ dành cho nghiên cứu và phát triển các hệ thống gợi ý dựa trên ngữ cảnh (Context-Aware Recommender Systems - CARS). Đây là một framework mã nguồn mở giúp các nhà nghiên cứu và lập trình viên dễ dàng triển khai và thử nghiệm các thuật toán gợi ý có sử dụng thông tin ngữ cảnh.



Hình 3: Biểu đồ mô tả kết quả

*1. Cấu trúc tổng quan của CARSKit

- **Generic Interfaces (Giao diện tổng quát):** Đây là các giao diện cơ bản cung cấp nền tảng cho việc phát triển các hệ thống gợi ý trong CARSKit. Những giao diện này có thể được mở rộng để tạo ra các mô hình gợi ý cụ thể.
- **ContextRecommender (Gợi ý theo ngữ cảnh):** Đây là lớp cụ thể hóa cho việc tích hợp ngữ cảnh vào quá trình gợi ý, cung cấp khả năng xử lý và gợi ý có tính đến các yếu tố ngữ cảnh khác nhau.
- **IterativeRecommender (Gợi ý lặp):** Thành phần này xử lý các thuật toán gợi ý dựa trên quy trình lặp lại, giúp mô hình học từ dữ liệu dần dần thông qua nhiều bước lặp.

- **Recommender (Gợi ý):** Đây là thành phần chính thực hiện việc gợi ý các mục (items) cho người dùng (users) dựa trên dữ liệu có sẵn.

*2. Data (Dữ liệu)

- **Structure (Cấu trúc):**
  - Bao gồm các loại dữ liệu khác nhau như SparseTensor (Tensor thưa), SparseMatrix (Ma trận thưa), DenseMatrix (Ma trận dày đặc), SparseVector (Vector thưa), DenseVector (Vector dày đặc), TensorEntry, MatrixEntry, và VectorEntry.
  - Những cấu trúc này giúp lưu trữ dữ liệu một cách hiệu quả, đặc biệt là trong các trường hợp dữ liệu ngữ cảnh có kích thước lớn và thưa.
- **Processor (Bộ xử lý):**
  - Các thành phần xử lý dữ liệu như DataDAO, DataSplitter, DataTransformer, v.v. giúp xử lý và biến đổi dữ liệu trước khi đưa vào các thuật toán gợi ý.
  - Chúng thực hiện các tác vụ như chia dữ liệu, chuẩn hóa dữ liệu, và biến đổi dữ liệu để phù hợp với các mô hình gợi ý.

*3. Algorithms (Thuật toán)

- **Baseline (Cơ sở):** Bao gồm các thuật toán cơ bản trong hệ thống gợi ý truyền thống, chẳng hạn như:
  - **Avg:** GlobalAverage, UserAverage, ItemAverage, ContextAvg, v.v.
  - **CF (Collaborative Filtering - Lọc cộng tác):** UserKNN, ItemKNN, SlopeOne, BiasedMF, NMF/PMF, SVD++, v.v.
  - **Ranking (Xếp hạng):** BPR, SLIM, LRMF, RankALS, RankSGD, v.v.
- **CARS (Context-Aware Recommender Systems):** Bao gồm các thuật toán dành riêng cho hệ thống gợi ý dựa trên ngữ cảnh:
  - **Transformation (Biến đổi):** Các phương pháp như ItemSplitting, UserSplitting, UISplitting giúp biến đổi dữ liệu trước khi áp dụng các mô hình gợi ý.
  - **Adaptation (Thích ứng):** Bao gồm các thuật toán thích ứng dựa trên ngữ cảnh:
    - **Independent (Độc lập):** TF (Tensor Factorization).
    - **Dependent (Phụ thuộc):** CAMF_CI, CAMF_ICS, CSLIM_CI, CSLIM_LCS, FM (Factorization Machines).

*4. Tính năng nổi bật

- **Khả năng mở rộng:** CARSKit có khả năng mở rộng với nhiều thuật toán khác nhau, cho phép người dùng dễ dàng thử nghiệm và so sánh các phương pháp khác nhau trong cùng một môi trường.
- **Hỗ trợ nhiều loại dữ liệu:** Bộ công cụ này hỗ trợ cả dữ liệu sparse (thưa) và dense (dày đặc), giúp xử lý dữ liệu có ngữ cảnh phức tạp và đa chiều.

- **Tích hợp ngữ cảnh:** Một trong những tính năng chính của CARSKit là khả năng tích hợp thông tin ngữ cảnh vào quy trình gợi ý, từ đó giúp cải thiện độ chính xác của các gợi ý.

*5. Tóm lại

CARSKit là một công cụ mạnh mẽ và linh hoạt cho nghiên cứu và phát triển các hệ thống gợi ý có tích hợp ngữ cảnh, phù hợp với các nhà nghiên cứu và lập trình viên muốn khám phá các thuật toán tiên tiến trong lĩnh vực này. Bộ công cụ này không chỉ hỗ trợ các thuật toán gợi ý truyền thống mà còn cung cấp các phương pháp tối ưu cho các mô hình gợi ý dựa trên ngữ cảnh, giúp nâng cao chất lượng gợi ý trong các ứng dụng thực tế.

3 DeepCarskit

3.1 Context-Aware Recommendations Based on Deep Learning Frameworks

3.1.1 Giới thiệu

https://carskit.github.io/DeepCARSKit.html https://dl.acm.org/doi/10.1145/3511047.3536404 Các tác giả đề xuất một khung làm việc mới về đề xuất dựa trên học sâu, trong đó tích hợp thông tin ngữ cảnh vào các phương pháp lọc cộng tác (collaborative filtering) dựa trên neural network. Khung làm việc này hướng tới việc cải thiện độ chính xác của các mô hình đề xuất thông qua việc sử dụng các biểu diễn ngữ cảnh khác nhau như ngữ cảnh rõ ràng (explicit context), ngữ cảnh tiềm ẩn không cấu trúc (unstructured latent context), và ngữ cảnh tiềm ẩn cấu trúc (structured latent context).

Kết quả của các thí nghiệm trên ba bộ dữ liệu khác nhau cho thấy rằng các mô hình đề xuất nhận biết ngữ cảnh dựa trên học sâu vượt trội hơn so với các phương pháp hiện đại khác trong việc dự đoán xếp hạng, tạo ra top-k đề xuất, và phân loại phản hồi của người dùng. Đặc biệt, việc sử dụng các ngữ cảnh tiềm ẩn cấu trúc trong mô hình đề xuất này đã mang lại hiệu suất tốt hơn so với các mô hình khác.

3.1.2 Deep Context-Aware Recommendation Framework

Hình ảnh này mô tả một khung làm việc cho hệ thống đề xuất nhận biết ngữ cảnh dựa trên học sâu (Deep Context-Aware Recommendation Framework). Khung làm việc này bao gồm ba bước chính:

Step 1: Context Representation (Đại diện Ngữ cảnh)

Đây là bước đầu tiên, nơi ngữ cảnh (context) được thu thập từ các nguồn khác nhau như thời tiết, vị trí, mức pin, tình trạng cuộc gọi, v.v. Ngữ cảnh này sau đó được chuyển đổi thành một vector ngữ cảnh (Context Vector) dưới dạng một biểu diễn số học.

Step 2: User-Item Data Collection (Thu thập Dữ liệu Người dùng-Sản phẩm)

Ở bước này, hệ thống thu thập dữ liệu về người dùng và sản phẩm. Dữ liệu này được đại diện dưới dạng các vector, bao gồm vector người dùng (User Vector) và vector sản phẩm (Item Vector).

Step 3: Deep Context-Aware Recommendation (Đề xuất Nhận biết Ngữ cảnh Dựa trên Học sâu)

Trong bước này, các vector người dùng, sản phẩm và ngữ cảnh được đưa vào mô hình đề xuất nhận biết ngữ cảnh dựa trên học sâu. Mô hình này sử dụng các dữ liệu này để huấn luyện và đưa ra dự đoán (Prediction Task), nhằm tính toán điểm đề xuất (score) $\hat{y}_{u,i,c}$ cho một người dùng cụ thể với một sản phẩm cụ thể trong một ngữ cảnh cụ thể. Kết quả của dự đoán này được so sánh với mục tiêu $y_{u,i,c}$ để cải thiện độ chính xác của mô hình.

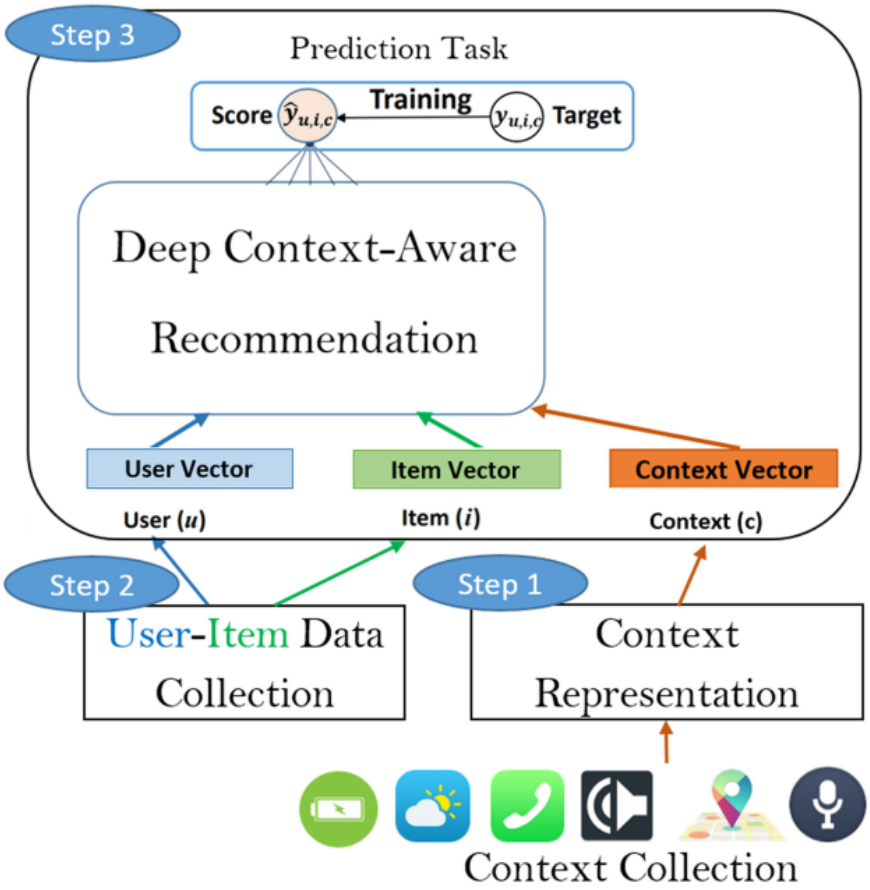


Fig. 1. Deep context-aware recommendation framework.

Hình 4: Biểu đồ mô tả kết quả

Mục tiêu của khung làm việc này là tối ưu hóa khả năng dự đoán của mô hình đề xuất bằng cách tích hợp thông tin ngữ cảnh vào quy trình học máy, giúp hệ thống đưa ra các đề xuất phù hợp hơn với người dùng dựa trên ngữ cảnh hiện tại của họ.

3.1.3 Mở rộng mô hình Neural Collaborative Filtering (NCF) với thông tin ngữ cảnh

Hình ảnh này minh họa cách mở rộng của mô hình Neural Collaborative Filtering (NCF) với thông tin ngữ cảnh.

Các thành phần trong hình:

User Latent Vector (Vector tiềm ẩn của người dùng):

Đây là biểu diễn tiềm ẩn của người dùng, được lưu trữ trong một ma trận P với kích thước $M \times K$, trong đó:

- M là số lượng người dùng.
- K là số chiều của vector tiềm ẩn.

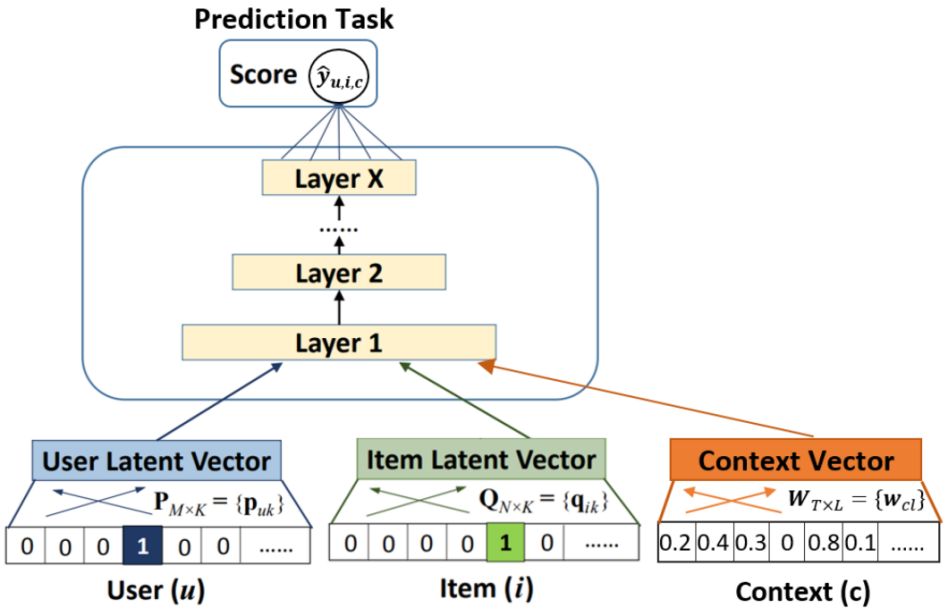


Fig. 3. Neural-CF (NCF) extension with contextual information.

Hình 5: Biểu đồ mô tả kết quả

Mỗi người dùng u sẽ được biểu diễn bằng một vector tiềm ẩn p_{uk}, thể hiện sở thích của người dùng đối với các sản phẩm.

Item Latent Vector (Vector tiềm ẩn của sản phẩm):

Đây là biểu diễn tiềm ẩn của sản phẩm, được lưu trữ trong một ma trận Q với kích thước $N \times K$, trong đó:

- N là số lượng sản phẩm.
- K là số chiều của vector tiềm ẩn.

Mỗi sản phẩm i sẽ được biểu diễn bằng một vector tiềm ẩn q_{ik}, thể hiện các đặc điểm của sản phẩm.

Context Vector (Vector ngữ cảnh):

Đây là biểu diễn ngữ cảnh dưới dạng một vector w_{cl}, lưu trữ trong ma trận W với kích thước $T \times L$, trong đó:

- T là số lượng ngữ cảnh.
- L là số chiều của vector ngữ cảnh.

Ngữ cảnh c có thể bao gồm các yếu tố như thời gian, địa điểm, thời tiết, v.v., và chúng được biểu diễn dưới dạng số liệu.

Quá trình xử lý:

Các vector tiềm ẩn của người dùng, sản phẩm, và ngữ cảnh được đưa vào Layer 1 của mô hình học sâu. Layer 1 và các lớp tiếp theo (Layer 2, Layer X...) trong mô hình học sâu chịu trách nhiệm học mối quan hệ phi tuyến tính giữa người dùng, sản phẩm và ngữ cảnh. Kết quả cuối cùng là một giá trị điểm $\hat{y}_{u,i,c}$, biểu diễn dự đoán của mô hình về mức độ phù hợp giữa người dùng u, sản phẩm i, trong ngữ cảnh c.

Ý nghĩa của mô hình:

Mô hình này được sử dụng để tối ưu hóa việc đề xuất sản phẩm bằng cách không chỉ xem xét sở thích của người dùng và đặc điểm của sản phẩm, mà còn tích hợp thêm thông tin ngữ cảnh. Điều này giúp các hệ thống đề xuất có thể cung cấp những gợi ý chính xác hơn, phù hợp với tình huống cụ thể mà người dùng đang gặp phải.

3.1.4 Mở rộng mô hình Neural Matrix Factorization (Neural-MF hoặc NeuMF) với thông tin ngữ cảnh

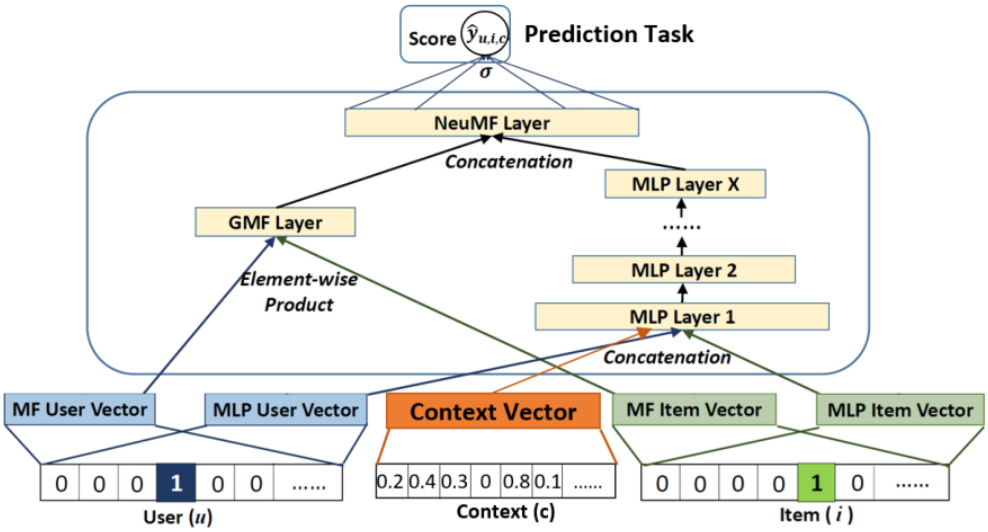


Fig. 4. Neural-MF (NeuMF) extension with contextual information.

Hình 6: Biểu đồ mô tả kết quả

Hình ảnh này minh họa cách mở rộng của mô hình **Neural Matrix Factorization (Neural-MF hoặc NeuMF)** với thông tin ngữ cảnh. NeuMF là một mô hình kết hợp giữa hai phương pháp: **Generalized Matrix Factorization (GMF)** và **Multilayer Perceptron (MLP)**.

Các thành phần chính trong hình:

1. **MF User Vector và MLP User Vector (Vector người dùng trong MF và MLP):**
   - **MF User Vector:** Biểu diễn người dùng trong không gian tiềm ẩn, tương tự như trong mô hình Matrix Factorization truyền thống. Mỗi người dùng được biểu diễn bởi một vector tiềm ẩn trong không gian này.
   - **MLP User Vector:** Biểu diễn người dùng trong không gian tiềm ẩn nhưng được học thông qua mạng neural (MLP). Đây là cách biểu diễn phi tuyến tính và có khả năng học phức tạp hơn.
2. **MF Item Vector và MLP Item Vector (Vector sản phẩm trong MF và MLP):**
   - **MF Item Vector:** Biểu diễn sản phẩm trong không gian tiềm ẩn tương tự như trong Matrix Factorization.

- **MLP Item Vector:** Biểu diễn sản phẩm trong không gian tiềm ẩn thông qua mạng neural (MLP).

3. **Context Vector (Vector ngữ cảnh):**

   - Đây là vector đại diện cho thông tin ngữ cảnh (như thời gian, vị trí, thời tiết, v.v.), được sử dụng để làm phong phú thêm các biểu diễn của người dùng và sản phẩm.

Quá trình xử lý trong mô hình:

1. **Element-wise Product (Nhân từng phần tử):**

   - Vector người dùng MF và vector sản phẩm MF được nhân từng phần tử với nhau, tạo ra một vector kết quả thể hiện mối tương quan giữa người dùng và sản phẩm theo cách tuyến tính. Kết quả này được đưa vào **GMF Layer** để tính toán.

2. **Concatenation (Kết hợp):**

   - Vector người dùng MLP, vector sản phẩm MLP, và vector ngữ cảnh được kết hợp lại với nhau bằng phép nối (concatenation). Kết quả này sau đó được đưa qua nhiều lớp của mạng neural (MLP Layer 1, 2, ..., X) để học mối quan hệ phi tuyến tính phức tạp hơn giữa người dùng, sản phẩm, và ngữ cảnh.

3. **NeuMF Layer (Lớp NeuMF):**

   - Kết quả từ GMF Layer và MLP Layer được kết hợp lại trong lớp NeuMF, nơi cả hai loại thông tin (MF và MLP) được hòa trộn để tạo ra một dự đoán chính xác hơn.

4. **Score $\hat{y}_{u,i,c}$:**

   - Đây là điểm dự đoán cuối cùng mà mô hình tạo ra, biểu diễn mức độ phù hợp giữa người dùng u, sản phẩm i, trong ngữ cảnh c. Điểm này có thể được sử dụng cho các tác vụ dự đoán như dự đoán xếp hạng, dự đoán sở thích người dùng, v.v.

Ý nghĩa của mô hình:

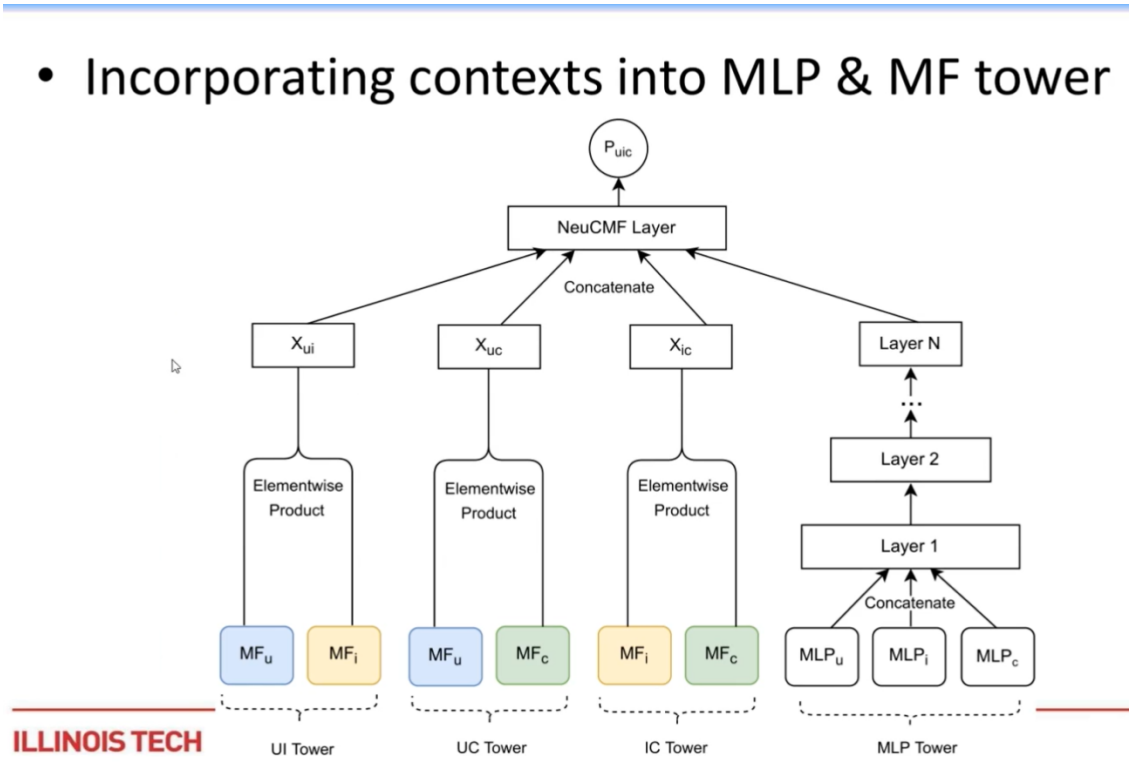
Mô hình NeuMF với thông tin ngữ cảnh được thiết kế để kết hợp các ưu điểm của Matrix Factorization và Multilayer Perceptron, đồng thời tích hợp thêm thông tin ngữ cảnh. Điều này giúp cải thiện độ chính xác của các đề xuất bằng cách tận dụng cả mối quan hệ tuyến tính và phi tuyến tính giữa người dùng, sản phẩm, và ngữ cảnh, giúp mô hình đề xuất chính xác hơn trong các tình huống thực tế phức tạp.

3.1.5 Tích hợp thông tin ngữ cảnh trong mô hình NeuMF

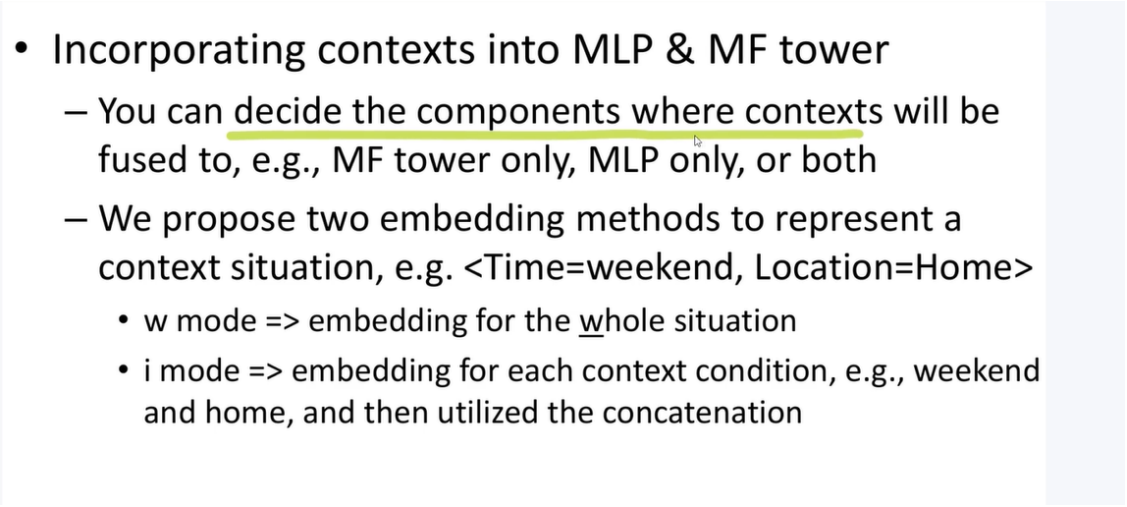
Hình ảnh này minh họa cách tích hợp thông tin ngữ cảnh vào hai phương pháp chính là **Multilayer Perceptron (MLP)** và **Matrix Factorization (MF)** trong mô hình NeuMF (Neural Collaborative Filtering).

Các thành phần chính trong hình:

1. **UI Tower (User-Item Tower):**



Hình 7: Biểu đồ mô tả kết quả



Hình 8: Biểu đồ mô tả kết quả

- **MF_u** và **MF_i:** Đây là các vector tiềm ẩn (latent vectors) cho người dùng (User) và sản phẩm (Item), tương ứng, trong phương pháp Matrix Factorization. Các vector này sau đó được nhân từng phần tử (element-wise product) để tạo ra đầu vào X_{ui} cho các bước xử lý tiếp theo.

2. **UC Tower (User-Context Tower):**

- **MF_u** và **MF_c:** Tương tự như UI Tower, nhưng thay vì chỉ sử dụng vector tiềm ẩn cho

người dùng và sản phẩm, UC Tower sử dụng vector tiềm ẩn cho người dùng (User) và ngữ cảnh (Context). Kết quả của việc nhân từng phần tử là đầu vào X_{uc}.

3. **IC Tower (Item-Context Tower):**

   - **MF_i** và **MF_c:** Đây là các vector tiềm ẩn cho sản phẩm (Item) và ngữ cảnh (Context). Sau khi nhân từng phần tử, kết quả là đầu vào X_{ic}.

4. **MLP Tower:**

   - **MLP_u, MLP_i, MLP_c:** Đây là các vector tiềm ẩn cho người dùng (User), sản phẩm (Item), và ngữ cảnh (Context) trong mô hình Multilayer Perceptron. Các vector này được kết hợp (concatenation) và đưa qua nhiều lớp của mạng neural (Layer 1, 2, ..., N) để học mối quan hệ phi tuyến tính phức tạp giữa các thành phần.

Quá trình xử lý trong mô hình:

1. **Element-wise Product (Nhân từng phần tử):**

   - Trong UI Tower, UC Tower, và IC Tower, các vector tiềm ẩn được nhân từng phần tử với nhau để tạo ra các đặc trưng kết hợp giữa người dùng, sản phẩm, và ngữ cảnh.

2. **Concatenation (Kết hợp):**

   - Sau khi tạo ra các đầu vào từ các tower khác nhau (UI Tower, UC Tower, IC Tower), các đầu vào này được kết hợp (concatenation) trong lớp NeuCMF Layer.

3. **NeuCMF Layer:**

   - Kết quả từ các phép kết hợp trong các tower và kết quả từ MLP Tower được đưa vào lớp NeuCMF Layer, nơi các thông tin này được xử lý để tạo ra dự đoán cuối cùng P_{uic}, biểu diễn khả năng người dùng u sẽ tương tác với sản phẩm i trong ngữ cảnh c.

Ý nghĩa của mô hình:
Mô hình này cho phép tích hợp thông tin ngữ cảnh vào cả hai phương pháp MF và MLP, cung cấp một cách tiếp cận toàn diện để xử lý dữ liệu đa chiều và phức tạp. Thông qua việc kết hợp thông tin từ nhiều nguồn (người dùng, sản phẩm, ngữ cảnh), mô hình có thể tạo ra các dự đoán chính xác và đáng tin cậy hơn, phục vụ cho các hệ thống đề xuất trong các tình huống đa dạng và phức tạp.

Các điểm chính trong hình:

- **Lựa chọn thành phần tích hợp ngữ cảnh:** Bạn có thể chọn tích hợp ngữ cảnh vào **MF Tower** hoặc **MLP Tower**, hoặc cả hai. Sự linh hoạt này cho phép bạn xác định cụ thể nơi mà ngữ cảnh có thể tác động đến mô hình, từ đó cải thiện hiệu quả của dự đoán.

- **Phương pháp nhúng (embedding) ngữ cảnh:**

  – **w mode:** Đây là phương pháp nhúng toàn bộ tình huống ngữ cảnh như một thực thể duy nhất. Ví dụ, nếu ngữ cảnh bao gồm cả "Thời gian = cuối tuần"và "Địa điểm = nhà", thì phương pháp này sẽ tạo ra một vector nhúng duy nhất để đại diện cho toàn bộ tình huống này.

– **i mode:** Với phương pháp này, mỗi điều kiện ngữ cảnh như “cuối tuần"và “nhà"sẽ được nhúng riêng lẻ thành các vector. Sau đó, các vector này sẽ được kết hợp lại bằng phép nối (concatenation) để tạo ra một vector tổng hợp đại diện cho ngữ cảnh.

Ý nghĩa:

- **Quyết định nơi tích hợp ngữ cảnh:** Việc lựa chọn nơi tích hợp ngữ cảnh vào mô hình có thể ảnh hưởng mạnh mẽ đến cách mô hình dự đoán. Bạn có thể chọn tích hợp ngữ cảnh vào một trong hai phương pháp hoặc cả hai, tùy vào mục đích cụ thể, để tận dụng tối đa các thông tin ngữ cảnh có sẵn.
- **Phương pháp nhúng ngữ cảnh:** Hai phương pháp này cho phép mô hình xử lý ngữ cảnh một cách linh hoạt và chi tiết, giúp mô hình có khả năng hiểu và phản ánh tốt hơn các tình huống phức tạp mà người dùng gặp phải, từ đó đưa ra các dự đoán chính xác hơn dựa trên ngữ cảnh cụ thể.

3.1.6 Hình ảnh này cung cấp một bảng tóm tắt về các cách kết hợp ngữ cảnh vào các tầng MLP và MF trong mô hình NeuCMF, với ba tùy chọn chính

- Incorporating contexts into MLP & MF tower

	Option	Towers in NeuCMF	Mode for Context Embedding
NeuCMF_{0i}	1	UI + MLP Towers with MLP_c	i mode
NeuCMF_{0w}			w mode
NeuCMF_{i0}	2	All 4 towers without MLP_c	i mode
NeuCMF_{w0}			w mode
NeuCMF_{ii}	3	All 4 towers with MLP_c	i mode
NeuCMF_{ww}			w mode

Hình 9: Biểu đồ mô tả kết quả

Các cột trong bảng:

1. **Option (Tùy chọn):** Đánh số các tùy chọn khác nhau cho mô hình NeuCMF.
2. **Towers in NeuCMF (Các tầng trong NeuCMF):**
   - Mô tả các tầng (towers) sẽ được sử dụng trong NeuCMF.

- **UI + MLP Towers với MLPc:** Chỉ kết hợp ngữ cảnh vào UI Tower và MLP Tower, có sử dụng ngữ cảnh trong MLP Tower.
- **All 4 towers without MLPc:** Sử dụng tất cả bốn tầng, nhưng không sử dụng ngữ cảnh trong MLP Tower.
- **All 4 towers with MLPc:** Sử dụng tất cả bốn tầng và kết hợp ngữ cảnh vào MLP Tower.

3. **Mode for Context Embedding (Chế độ nhúng ngữ cảnh):** Chỉ ra cách ngữ cảnh được nhúng vào trong mô hình.
   - **i mode:** Nhúng ngữ cảnh vào từng thành phần của ngữ cảnh (embedding for each context condition).
   - **w mode:** Nhúng ngữ cảnh cho toàn bộ tình huống (embedding for the whole situation).

Các tùy chọn cụ thể:

- **NeuCMF$_{0i}$ và NeuCMF$_{0w}$ (Option 1):** Sử dụng UI và MLP Towers với MLPc. Trong đó NeuCMF$_{0i}$ sử dụng **i mode** và NeuCMF$_{0w}$ sử dụng **w mode**.
- **NeuCMF$_{i0}$ và NeuCMF$_{w0}$ (Option 2):** Sử dụng cả bốn tầng (UI, UC, IC, MLP), nhưng không có ngữ cảnh trong MLPc. NeuCMF$_{i0}$ sử dụng **i mode** và NeuCMF$_{w0}$ sử dụng **w mode**.
- **NeuCMF$_{ii}$ và NeuCMF$_{ww}$ (Option 3):** Sử dụng tất cả bốn tầng và kết hợp ngữ cảnh vào MLPc. NeuCMF$_{ii}$ sử dụng **i mode** và NeuCMF$_{ww}$ sử dụng **w mode**.

Ý nghĩa của bảng:

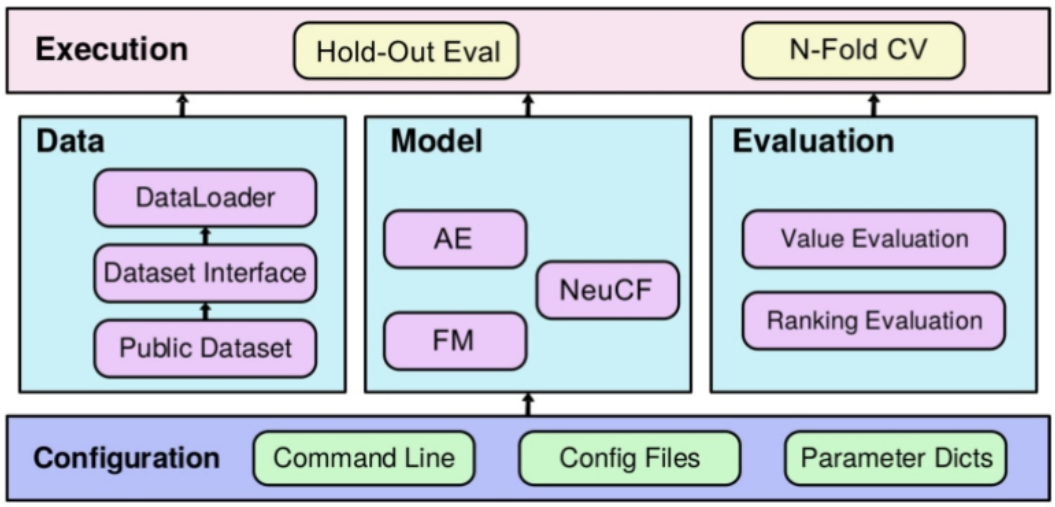
- **Linh hoạt trong lựa chọn mô hình:** Bảng này thể hiện sự linh hoạt trong việc tích hợp ngữ cảnh vào các thành phần khác nhau của mô hình, cho phép người dùng tùy chỉnh mô hình NeuCMF dựa trên yêu cầu cụ thể của họ.
- **Chế độ nhúng ngữ cảnh:** Việc sử dụng các chế độ nhúng khác nhau (i mode hoặc w mode) cho phép mô hình hiểu ngữ cảnh ở mức độ chi tiết khác nhau, giúp tối ưu hóa khả năng dự đoán của mô hình trong các tình huống khác nhau.

Bảng này tóm tắt các lựa chọn khả thi để cấu hình mô hình NeuCMF tùy thuộc vào cách mà người dùng muốn kết hợp ngữ cảnh vào quá trình dự đoán.

3.2 Chi tiết về DeepCarskit

Giới thiệu một ví dụ cụ thể và giải thích cách áp dụng phương pháp đã đề xuất.

Hình ảnh này mô tả cấu trúc và các thành phần chính của **DeepCARSKit**, một công cụ được sử dụng để đánh giá các hệ thống đề xuất ngữ cảnh dựa trên các mô hình học sâu.



Hình 10: Biểu đồ mô tả kết quả

Các thành phần chính của DeepCARSKit:

1. **Execution (Thực thi):**
   - **Hold-Out Eval:** Đây là phương pháp đánh giá giữ lại (Hold-Out Evaluation), trong đó một phần dữ liệu được tách ra để làm dữ liệu kiểm thử, và phần còn lại được sử dụng để huấn luyện mô hình.
   - **N-Fold CV:** Đây là phương pháp đánh giá chéo k lần (N-Fold Cross Validation), trong đó dữ liệu được chia thành k phần, và quá trình huấn luyện và kiểm thử được thực hiện k lần, mỗi lần sử dụng một phần khác nhau làm dữ liệu kiểm thử và phần còn lại làm dữ liệu huấn luyện.
2. **Data (Dữ liệu):**
   - **Public Dataset (Tập dữ liệu công khai):** Đây là nguồn dữ liệu được sử dụng, có thể là các tập dữ liệu công khai từ các nghiên cứu trước.
   - **Dataset Interface (Giao diện tập dữ liệu):** Cung cấp cách tiếp cận và tương tác với các tập dữ liệu.
   - **DataLoader:** Công cụ chịu trách nhiệm tải và quản lý dữ liệu từ các nguồn tập dữ liệu công khai.
3. **Model (Mô hình):**
   - **AE (Autoencoder):** Một loại mô hình học sâu tự mã hóa được sử dụng để học biểu diễn của dữ liệu bằng cách nén thông tin.
   - **NeuCF (Neural Collaborative Filtering):** Mô hình lọc cộng tác dựa trên học sâu, được sử dụng để tạo các hệ thống đề xuất cá nhân hóa.

- **FM (Factorization Machine):** Một loại mô hình sử dụng các phép phân tích ma trận để dự đoán tương tác giữa các biến số, thường được sử dụng trong các hệ thống đề xuất.

4. **Evaluation (Đánh giá):**

   - **Value Evaluation (Đánh giá giá trị):** Phương pháp đánh giá dựa trên các chỉ số giá trị, chẳng hạn như độ chính xác, MSE (Mean Squared Error), hoặc RMSE (Root Mean Squared Error).
   - **Ranking Evaluation (Đánh giá xếp hạng):** Phương pháp đánh giá dựa trên xếp hạng, chẳng hạn như tỷ lệ chính xác tại k (Precision@k), MAP (Mean Average Precision), hoặc NDCG (Normalized Discounted Cumulative Gain).

5. **Configuration (Cấu hình):**

   - **Command Line:** Các cấu hình và tham số có thể được thiết lập qua dòng lệnh.
   - **Config Files (Tập tin cấu hình):** Các cấu hình được lưu trữ và quản lý trong các tập tin cấu hình.
   - **Parameter Dicts:** Sử dụng các từ điển tham số để quản lý và truyền các tham số cần thiết cho mô hình.

Ý nghĩa của DeepCARSKit: **DeepCARSKit** là một bộ công cụ mạnh mẽ và linh hoạt, cho phép các nhà nghiên cứu và nhà phát triển có thể đánh giá và thử nghiệm các hệ thống đề xuất dựa trên nhiều mô hình học sâu khác nhau. Với khả năng cấu hình đa dạng và hỗ trợ nhiều phương pháp đánh giá, DeepCARSKit giúp tối ưu hóa và so sánh hiệu suất của các mô hình trong các tình huống thực tế, đặc biệt là trong việc xử lý ngữ cảnh và cá nhân hóa đề xuất cho người dùng.

4 Tài liệu tham khảo

Tài liệu

[1] Donald E. Knuth, *The TeXbook*, Addison-Wesley, 1990.

[2] ex, *ex*, https://fr.slideshare.net/irecsys/contextaware-recommendation-a-quick-view

[3] ex, *ex*, https://github.com/irecsys/DeepCARSKit

[4] 2022, *A Family of Neural Contextual Matrix Factorization Models for Context-Aware Recommendations*, https://dl.acm.org/doi/10.1145/3511047.3536404

[5] 2022. *DeepCARSKit: A Deep Learning Based Context-Aware Recommendation Library*. Yong Zheng. Software Impacts.

[6] 2022. *DeepCARSKit: A Demo and User Guide*. Yong Zheng. In Adjunct Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization. ACM. https://doi.org/10.1145/3511047.3536417

[7] 2020. *Research directions in session-based and sequential recommendation*. Dietmar Jannach, Bamshad Mobasher, and Shlomo Berkovsky. User Modeling and User-Adapted Interaction, 30(4), pp. 609–616.

[8] 2020. *A Systematic Review on Context-Aware Recommender Systems using Deep Learning and Embeddings*. Igor André Pegoraro Santana and Marcos Aurelio Domingues. arXiv preprint arXiv:2007.04782.

[9] 2020. *Context-Aware Recommendations Based on Deep Learning Frameworks*. Moshe Unger, Alexander Tuzhilin, and Amit Livne. ACM Transactions on Management Information Systems (TMIS), 11(2), pp. 1–15.

[10] 2019. *Deep learning based recommender system: A survey and new perspectives*. Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. ACM Computing Surveys (CSUR), 52(1), pp. 1–38.

[11] 2018. *Context-Aware Recommendations*. Yong Zheng and Bamshad Mobasher. World Scientific Publishing, pp. 173–202.

[12] 2017. *Neural collaborative filtering*. Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. In Proceedings of the 26th International Conference on World Wide Web, pp. 173–182.

[13] 2017. *DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction*. Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. In Proceedings of the 26th International Joint Conference on Artificial Intelligence. AAAI Press, pp. 1725–1731.

[14] 2015. *Session-based recommendations with recurrent neural networks*. Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. arXiv preprint arXiv:1511.06939.

[15] 2014. *Splitting approaches for context-aware recommendation: An empirical study*. Yong Zheng, Robin Burke, and Bamshad Mobasher. In Proceedings of the 29th Annual ACM Symposium on Applied Computing, ACM, pp. 274–279.

[16] 2011. *Matrix factorization techniques for context aware recommendation*. Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. In Proceedings of the Fifth ACM Conference on Recommender Systems. ACM, pp. 301–304.

[17] 2011. *Context-Aware Recommender Systems*. Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alexander Tuzhilin. AI Magazine, 32(3), pp. 67–80.