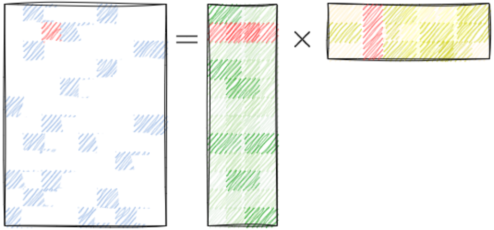
Personalized Machine Learning

Deep Learning Methods

Rodrigo Alves

October 29, 2024.

Matrix Factorization



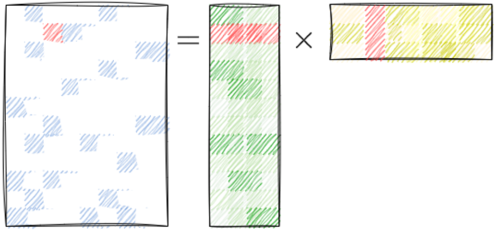
$$R = U \times V^\top$$

$$r_{ij} = u_i^\top \times v_j$$

$$\mathcal{L}_{\text{Exp}} = \sum_{i,j \in \Omega} (r_{ij} - u_i^\top v_j)^2 \qquad \mathcal{L}_{\text{Imp}_1} = \sum_{i,j} c_{ij} (r_{ij} - u_i^\top v_j)^2$$

$$\mathcal{L}_{\text{Imp}_2} = -\sum_{i,j} r_{ij} \log(\sigma(u_i^\top v_j)) + ((1 - r_{ij}) \log(1 - \sigma(u_i^\top v_j)))$$

Matrix Factorization (Sampled)



$$R = U \times V^\top$$

$$r_{ij} = u_i^\top \times v_j$$

$$\mathcal{L}_{\text{Exp}} = \sum_{i,j \in \Omega} (r_{ij} - u_i^\top v_j)^2 \qquad \mathcal{L}_{\text{Imp}_1} = \sum_{i,j \in \Omega^*} c_{ij}(r_{ij} - u_i^\top v_j)^2$$

$$\mathcal{L}_{\text{Imp}_2} = -\sum_{i,j \in \Omega^*} r_{ij} \log(\sigma(u_i^\top v_j)) + ((1 - r_{ij}) \log(\sigma(1 - u_i^\top v_j)))$$

CF as a Supervised Learning Problem

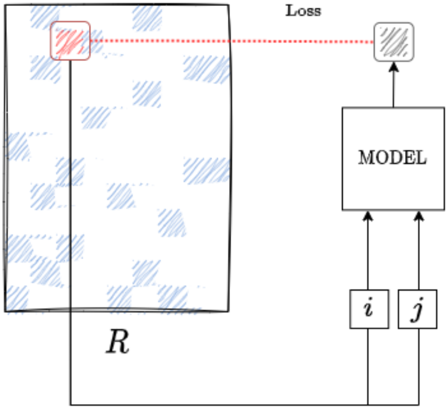
- Matrix Factorization can be seen as a supervised learning problem.

Supervised Learning Data

- Inputs: raw data instances $x_1, x_2, \ldots, x_n$, where $x_l \in \mathbb{R}^p$.
- Labels: annotations of the inputs $y_1, y_2, \ldots, y_N$, where $y_l \in \mathbb{R}$.

What are the xs and ys in CF-based matrix factorization?

CF as a Supervised Learning Problem



CF as a Supervised Learning Problem

- Thus, $x_l = \{i, j\} \in \mathbb{R}^2$ and $y_l \in \mathbb{R}$.
  - For explicit feedback, for example, $y_l \in \{1, 2, 3, 4, 5\}$.
  - For implicit feedback, for example, $y_l \in \{0, 1\}$.
- Therefore, we can see an MF optimization problem (without regularization)

$$\sum_{i,j \in \Omega} (r_{ij} - u_i^\top v_j)^2$$

as

$$\sum_{i,j \in \Omega} (r_{ij} - g(i, j))^2$$

- So far, we see $g(i, j)$ as a linear function.
- If we replace $g(i, j)$ with a deep architecture, we would have a deep CF problem.

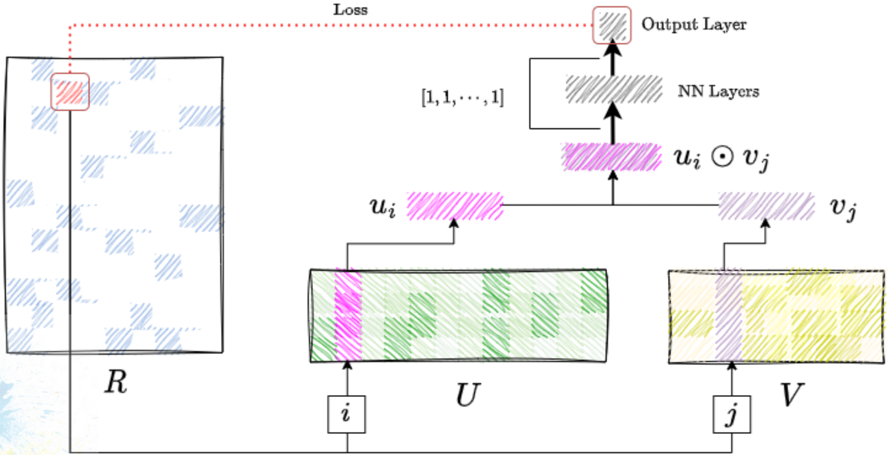
Generalized Matrix Factorization(GMF)

- Many deep learning methods can generalize their shallow equivalents.
- For example, in our previous toy example, the shallow method is equivalent to linear regression with the predictor $y = \omega_{\text{LR}} x$.
- An equivalent version of this shallow model in the context of a deep model could be represented as follows:

$$\begin{aligned} y &= \text{ReLU}(x \times \omega_{11} + \beta_{11}) \times \omega_{21} + \text{ReLU}(x \times \omega_{12} + \beta_{12}) \times \omega_{22} \\ &= \text{ReLU}(x \times 1 + 0) \times \omega_{\text{LR}} + \text{ReLU}(x \times -1 + 0) \times -\omega_{\text{LR}} \\ &= \omega_{\text{LR}} x \end{aligned}$$

- Similarly, we could have a deep matrix factorization method that can generalize shallow matrix factorization methods.

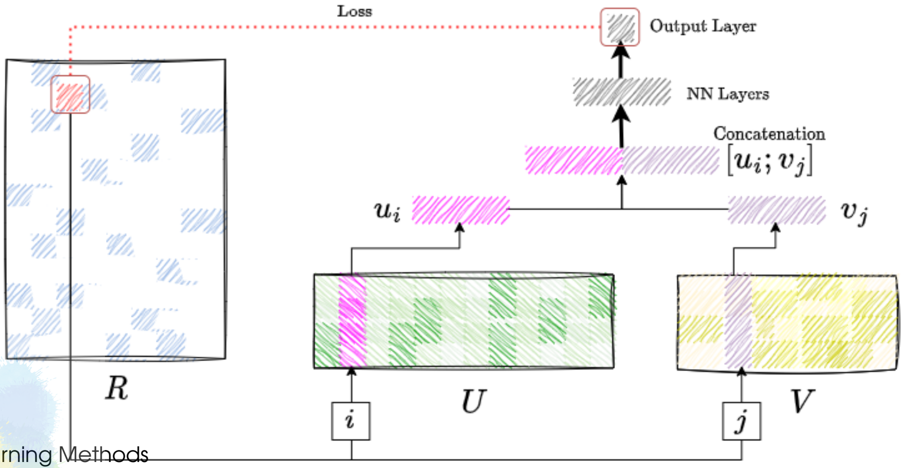
GMF



Multilayer Perceptron (MLP)

- Another natural view of matrix factorization is as a deep model using a multilayer perceptron.
- This model is known for its fully connected dense layers.
- Although the model can also represent simple models, such as shallow matrix factorization, it is more powerful and can represent a wider range of functions.
- Powerful models are more susceptible to overfitting.
- Architectural design and regularization, as is typical in deep models, are best evaluated through validation procedures.

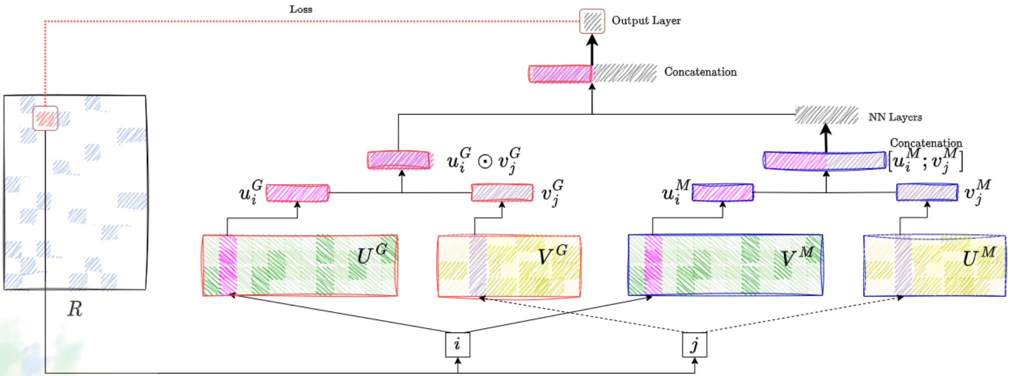
MLP



Neural Collaborative Filtering (NCF)

- One of the early approaches to using deep methods in collaborative filtering.
- It consists of two modules:
  - GMF (Generalized Matrix Factorization).
  - MLP (Multilayer Perceptron).
- Traditionally used for implicit feedback.
  - Utilizes sampling to balance the negative feedback.

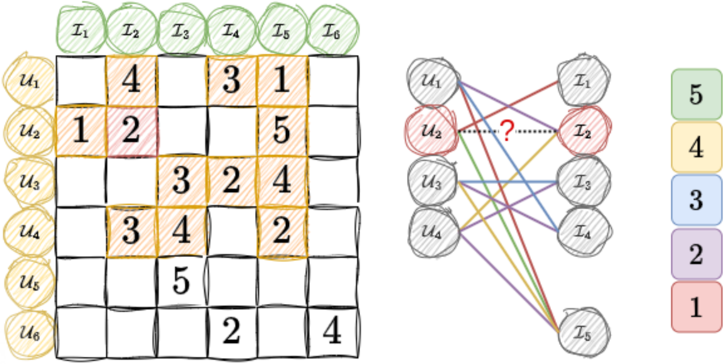
NCF


Loss
Output Layer
Concatenation
NN Layers
Concatenation
$[u_i^M; v_j^M]$
$u_i^G \odot v_j^G$
u_i^G
v_j^G
u_i^M
v_j^M
U^G
V^G
V^M
U^M
R
i
j

Graph Neural Networks (GNNs) for CF

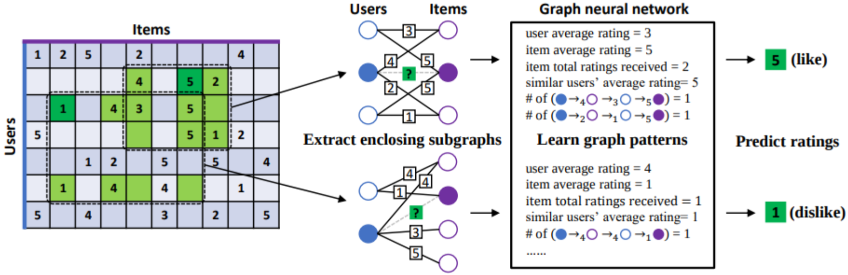
- Traditional collaborative filtering methods often ignore the rich structural information present in user-item interaction data.
- GNNs are a powerful approach to leverage this structure for recommendation systems.
- Key components:
  - Graph Representation: Model user-item interactions as a bipartite graph where users and items are nodes, and interactions are edges.
  - Message Passing: Propagate information along graph edges to capture collaborative patterns.
- Advantages of GNNs:
  - Ability to handle sparsity.
  - Capture complex relationships beyond traditional matrix factorization.
  - Improved recommendation accuracy.

GNNs



I_1 I_2 I_3 I_4 I_5 I_6
U_1 U_2 U_3 U_4 U_5 U_6
4 3 1
1 2 5
3 2 4
3 4 2
5
2 4
U_1 U_2 U_3 U_4
?
I_1 I_2 I_3 I_4 I_5
5
4
3
2
1

GNNs

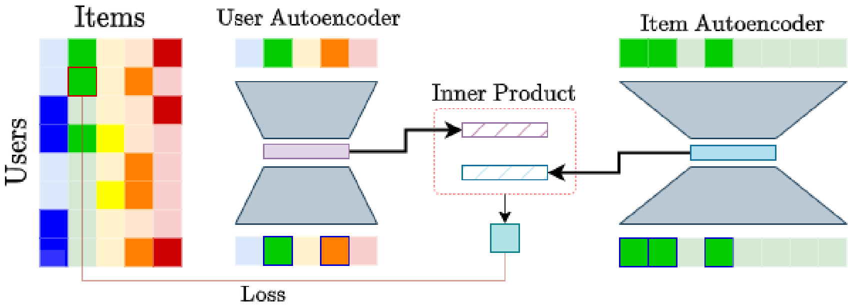


Items
Users
Users
Items
Extract enclosing subgraphs
Graph neural network
user average rating = 3
item average rating = 5
item total ratings received = 2
similar users' average rating= 5
Learn graph patterns
user average rating = 4
item average rating = 1
item total ratings received = 1
similar users' average rating= 1
......
5 (like)
Predict ratings
1 (dislike)

Deep Autoencoders for Collaborative Filtering

- We've explored how autoencoders provide a versatile approach for collaborative filtering tasks.
- Deep autoencoders are neural networks explicitly designed to acquire efficient representations of user-item interactions.
- Autoencoders are frequently employed to extract embeddings from input and output data.
- They can serve as a powerful technique to complement deep collaborative approaches.

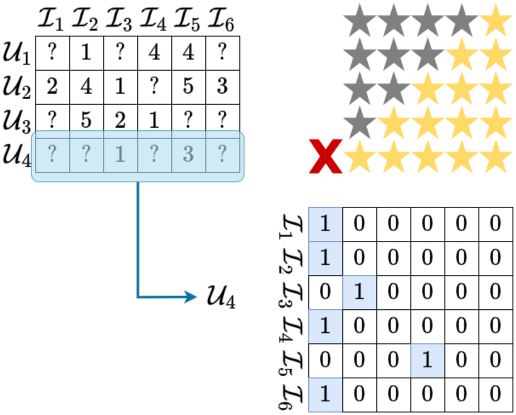
Deep Autoencoders



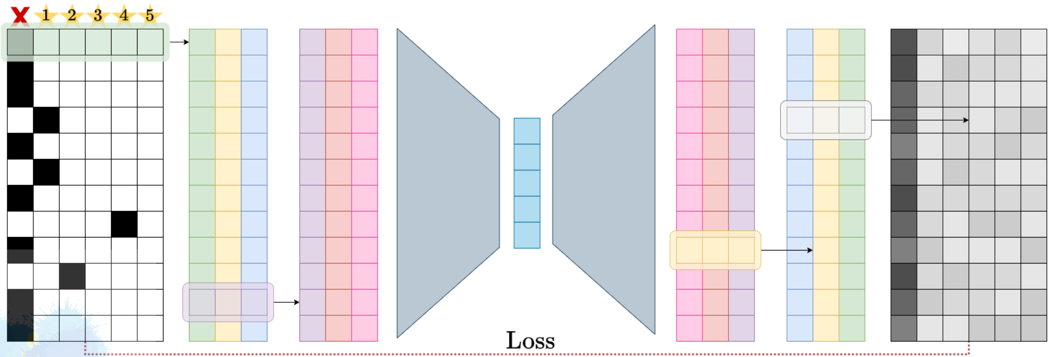
Mixing Implicit and Explicit Feedback

- Some methods incorporate a mixture of implicit and explicit feedback concepts.
- Consider the following:
  - Explicit feedback: Ratings from 1 to 5 stars;
  - Implicit feedback: Whether the user interacts (1) or not (0) with the same item.
- We will introduce the 1-by-1 convolutional autoencoder to combine implicit and explicit feedback.

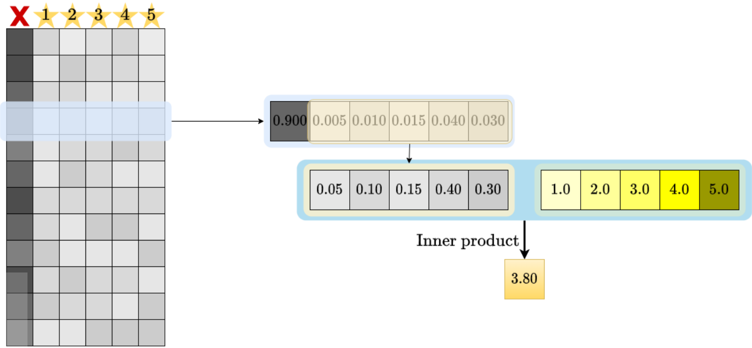
Mixing Implicit and Explicit Feedback



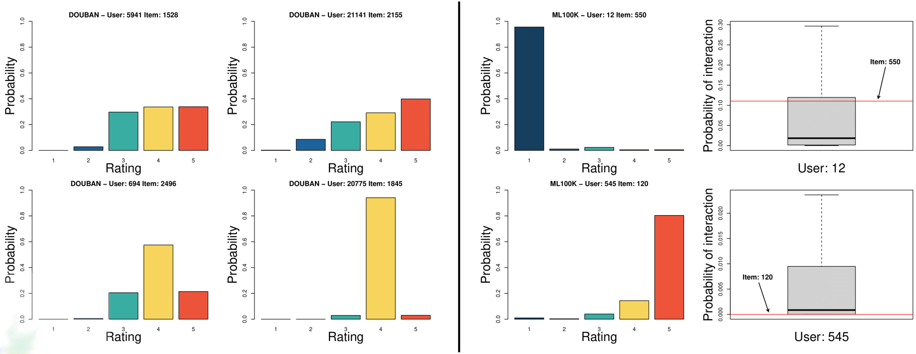
Mixing Implicit and Explicit Feedback



Mixing Implicit and Explicit Feedback



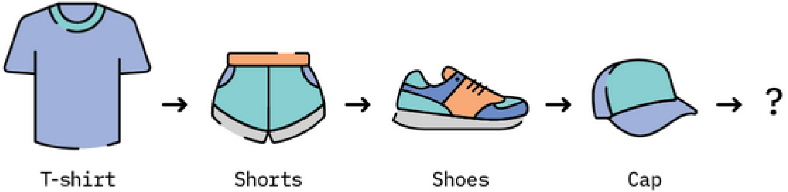
Mixing Implicit and Explicit Feedback



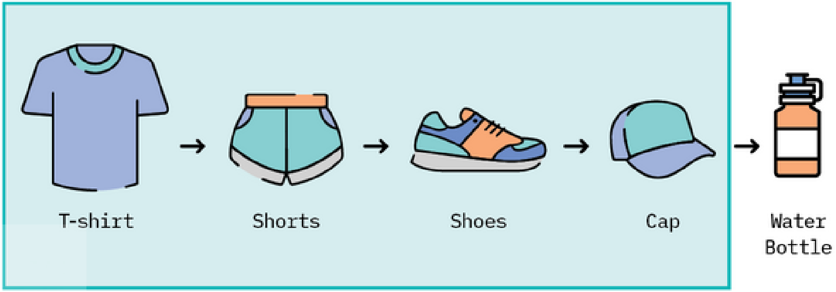
Sequential-based Recommendation

- In traditional recommendation systems, user-item interactions are treated independently.
- However, many real-world scenarios involve sequences of user actions or events.
- Sequential-based recommendation models take into account the order and timing of user interactions.
- This is especially important for applications like:
  - Recommending products in an e-commerce session.
  - Suggesting the next movie or video in a user's watch history.
  - Personalizing content in news and article recommendation systems.
- Sequential recommendation models leverage the sequential patterns, temporal dynamics, and user behavior to provide more accurate and context-aware recommendations.

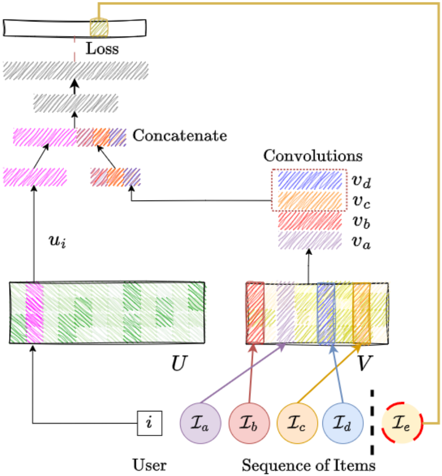
Sequential-based Recommendation



Sequential-based Recommendation



Mixing Implicit and Explicit Feedback



Obrigado :) - Faculty of Information Technology