Recommender System

Xanh Ho
University of Science – Ho Chi Minh

June 21, 2018

Why Recommender System?

Introduction to Recommender System

Terminology Explanation

- User
- Item: movies, books, songs, …

- n_u : number of users
- n_m : number of items

=> Rating matrix/Utility matrix: $(n_u \; x \; n_m)$

- Rating information
  - Implicit feedback: number of times website visited, read a book, view a book, heard a song, …
  - Explicit feedback: user rating for a movie

Problem Formulation

	I1	I2	I3	I4
U1	4	?	3	?
U2	?	3	?	?
U3	5	?	?	2
U4	?	1	?	?
U5	?	?	4	5

Item descriptions

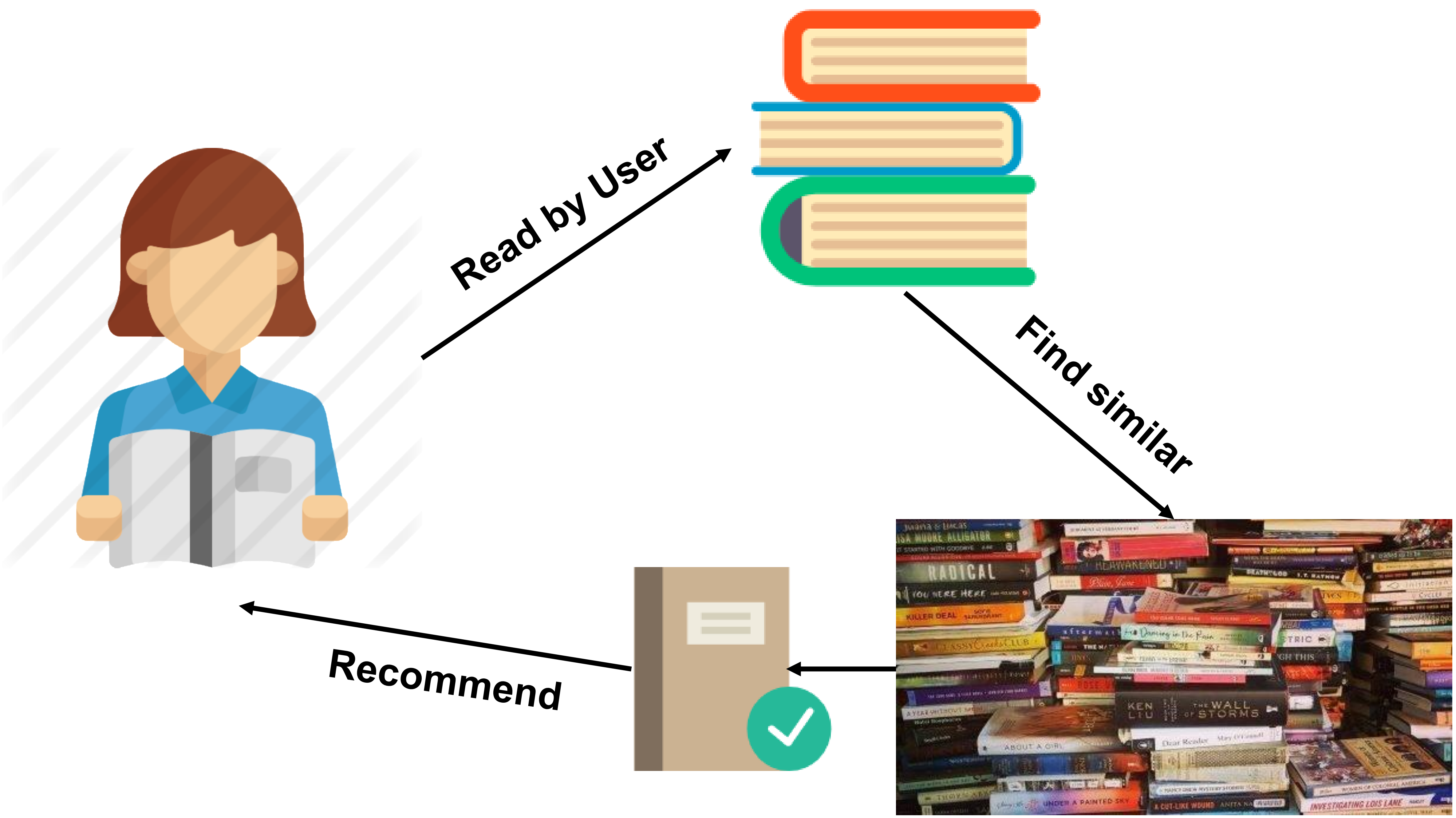
	I1	I2	I3	I4
U1	4	2	3	5
U2	3	3	2	4
U3	5	1	5	2
U4	1	1	3	2
U5	4	3	4	5

Recommender System Approaches

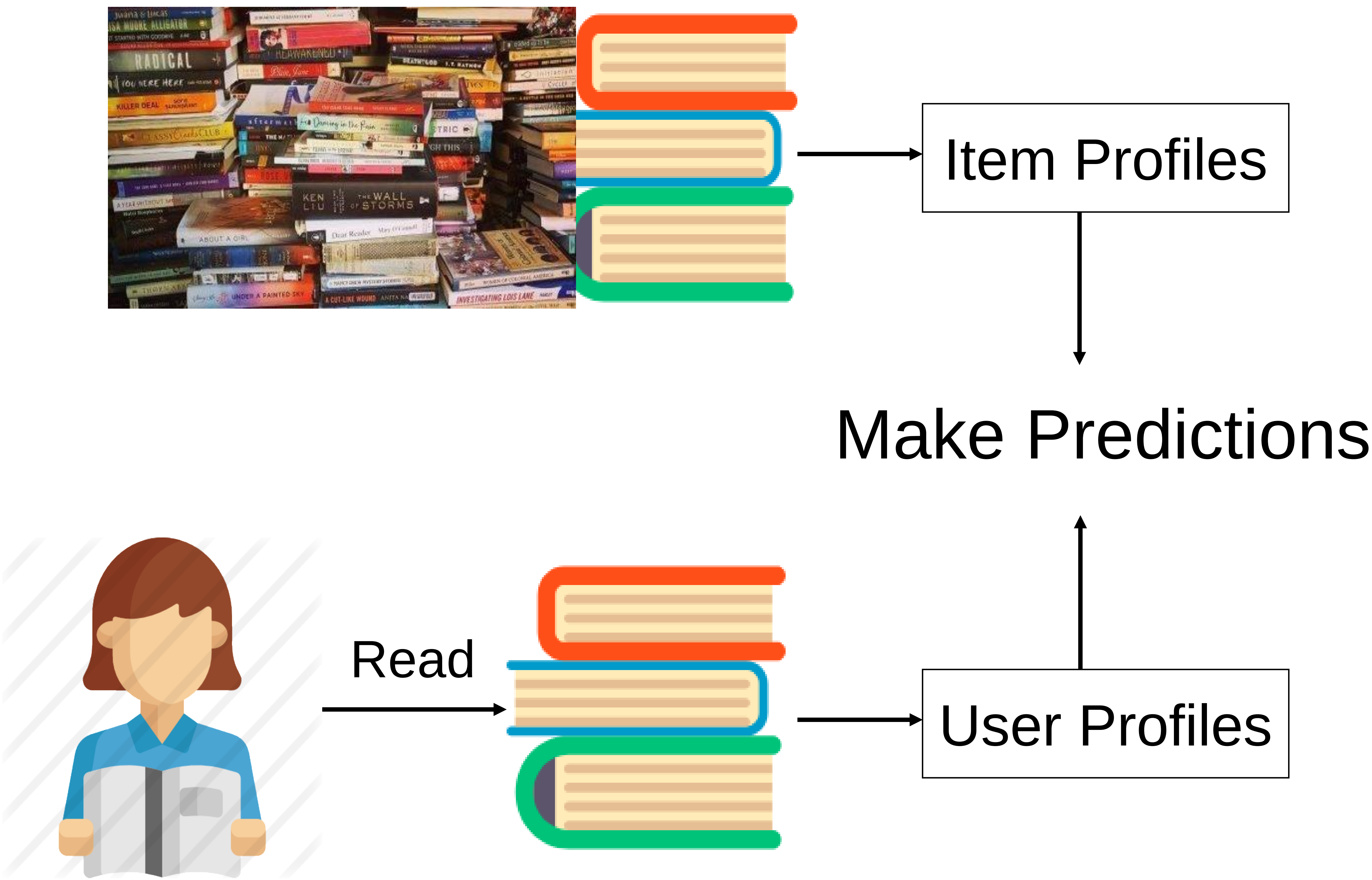
- Content-based
- Collaborative Filtering
- Hybrid methods

Content-based

- **Main idea:** "Recommend items that are similar to those the user liked in the past"



Content-based



Item Profiles

- List of features
  - Movies: actor, title, director, genre, …
  - Songs: singer, year release, musician, …
  - Books: Title, author, genre, …
- Text features
  - List of important words
  - Use TF-IDF

User Profiles and Making Predictions

- User Profiles:
  - User has rated items with profiles $x_1, \ldots, x_n$
  - Simple: average of rated item profiles

- Making Predictions:
  - Item profiles: $x_1, \ldots, x_k$
  - User profile: u_1
  - Prediction: Cosine similarity $\cos(x_q, u_1), q \in [1, k]$

Learning a User Model

- Regression/Classification task
- Training data: item profile and ratings

	I1	I2	I3	I4
U1	4	?	3	?
U2	?	3	?	?
U3	5	?	?	2
U4	?	1	?	?
U5	?	?	4	5

	x1	x2
I1	0.9	0.1
I2	0.4	0.6
I3	0.5	0.5
I4	0.3	0.7

Learning a User Model

- $r(i, j) = 1$ if user i rated movie j (0 otherwise)
- $y^{(i,j)}$ = rating by user i on movie j
- $w^{(i)}$: parameter vector for user i
- $x^{(j)}$: feature vector for movie j
- For user i, movie j, predicted rating: $(w^{(i)})^T x^{(j)}$
- n: number of features in movie

- To learn $w^{(i)}$ parameter for user i:

$$\min_{w^{(i)}} \frac{1}{2} \sum_{j:r(i,j)=1} ((w^{(i)})^T x^{(j)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (w_k{}^{(i)})^2$$

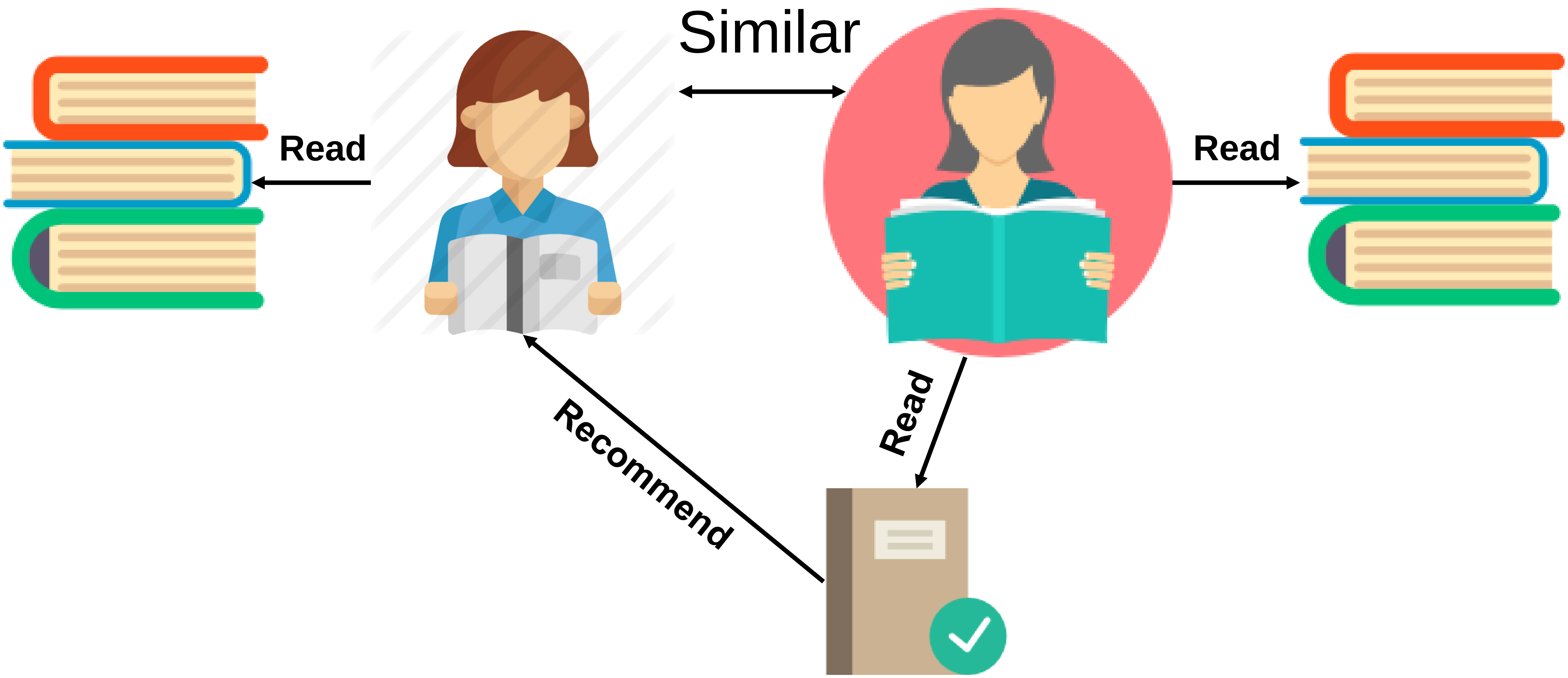
Comments

- Advantages:
  - No need data of other users
  - Able to recommend new and unpopular items
  - Explanations for recommended items

- Disadvantages:
  - Never recommends items outside user's content profile
  - Unable to exploit quality judgments of other users
  - Cold-start problem for new users

Collaborative Filtering

- **Main idea:** “Recommend items that similar users liked”



Collaborative Filtering

- Neighborhood-based
  - **User-user collaborative filtering**
  - Item-item collaborative filtering
- Matrix Factorization

User-user Collaborative Filtering

Step 1: Normalized data

	U1	U2	U3	U4	U5	U6	U7
I1	5	5	2	0	1	?	?
I2	4	?	?	0	?	2	?
I3	?	4	1	?	?	1	1
I4	2	2	3	4	4	?	4
I5	2	0	4	?	?	?	5

3.25	**2.75**	**2.5**	**1.33**	**2.5**	**1.5**	**3.33**

Mean user ratings

	U1	U2	U3	U4	U5	U6	U7
I1	1.75	2.25	-0.5	-1.3	-1.5	0	0
I2	0.75	0	0	-1.3	0	1.5	0
I3	0	1.25	-1.5	0	0	-0.5	-2.3
I4	-1.2	-0.7	0.5	2.67	1.5	?	0.67
I5	-1.2	-2.7	1.5	0	0	0	1.67

Normalized rating matrix

User-user Collaborative Filtering

Step 2: Calculate user similarity: cosine similarity

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
u_0	1	0.83	-0.58	-0.79	-0.82	0.2	-0.38
u_1	0.83	1	-0.87	-0.40	-0.55	-0.23	-0.71
u_2	-0.58	-0.87	1	0.27	0.32	0.47	0.96
u_3	-0.79	-0.40	0.27	1	0.87	-0.29	0.18
u_4	-0.82	-0.55	0.32	0.87	1	0	0.16
u_5	0.2	-0.23	0.47	-0.29	0	1	0.56
u_6	-0.38	-0.71	0.96	0.18	0.16	0.56	1

User similarity matrix

User-user Collaborative Filtering

Step 3: Rating prediction

$$\hat{y}_{i,u} = \frac{\sum_{u_j \in \mathcal{N}(u,i)} \bar{y}_{i,u_j} \mathbf{sim}(u, u_j)}{\sum_{u_j \in \mathcal{N}(u,i)} |\mathbf{sim}(u, u_j)|}$$

	U1	U2	U3	U4	U5	U6	U7
I1	1.75	2.25	-0.5	-1.3	-1.5	0.18	-0.63
I2	0.75	0.48	-0.17	-1.3	-1.33	1.5	0.05
I3	0.91	1.25	-1.5	-1.84	-1.78	-0.5	-2.3
I4	-1.2	-0.7	0.5	2.67	1.5	0.59	0.67
I5	-1.2	-2.7	1.5	1.57	1.56	01.59	1.67

User-user Collaborative Filtering

Step 5: Denormalized

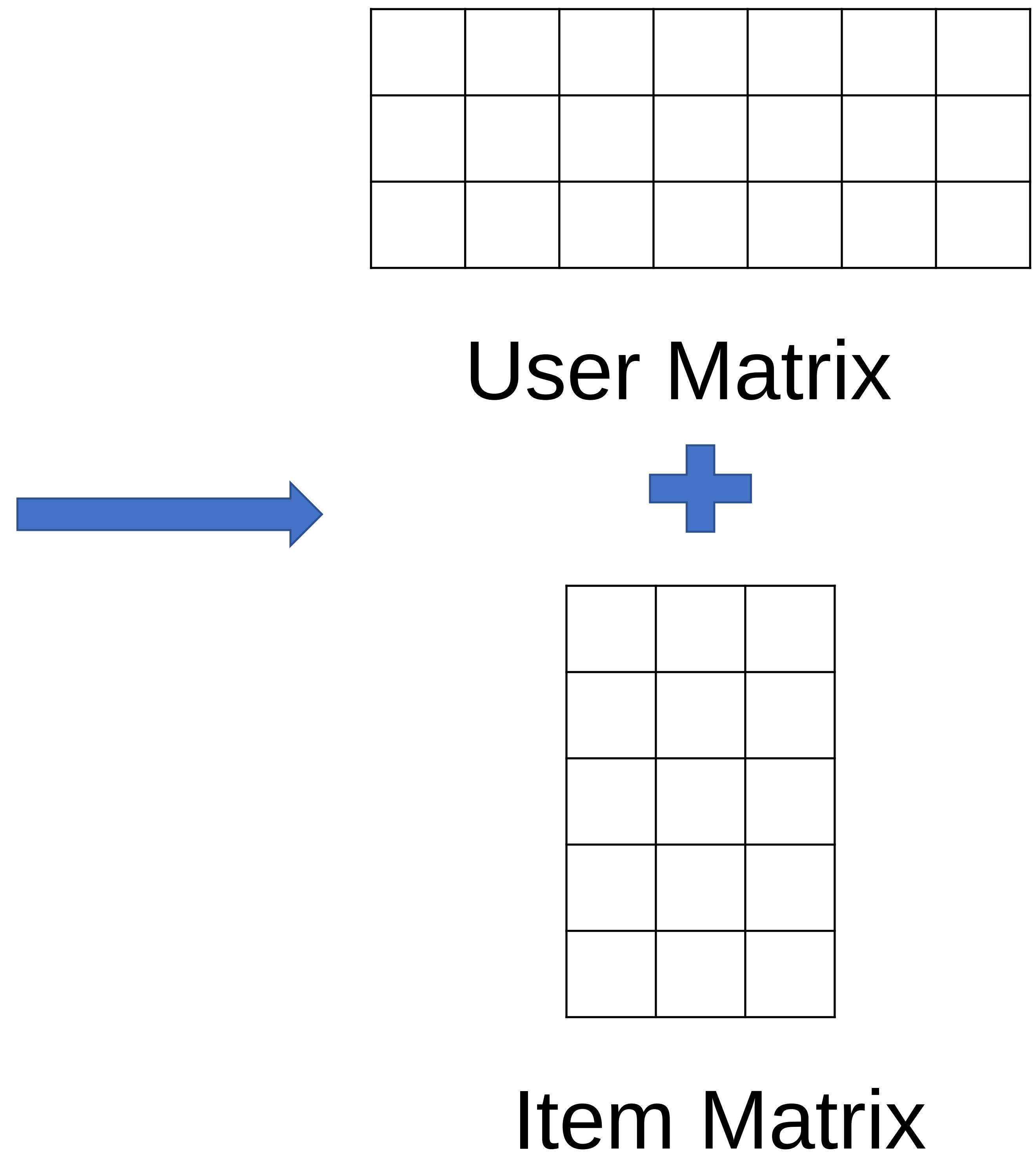
	U1	U2	U3	U4	U5	U6	U7
I1	5	5	2	0	1	1.68	2.70
I2	4	3.23	2.33	0	1.67	2	3.38
I3	4.15	4	1	-0.5	0.71	1	1
I4	2	2	3	4	4	2.10	4
I5	2	0	4	2.9	4.06	3.10	5

Matrix Factorization

- n_u : number of users
- n_m : number of items
- n_u = 7, n_m = 5, K = 3

	U1	U2	U3	U4	U5	U6	U7
I1	5	5	2	0	1	?	?
I2	4	?	?	0	?	2	?
I3	?	4	1	?	?	1	1
I4	2	2	3	4	4	?	4
I5	2	0	4	?	?	?	5

Rating Matrix



Matrix Factorization

- Content-based, for user i:

$$\min_{w^{(i)}} \frac{1}{2} \sum_{j:r(i,j)=1} ((w^{(i)})^T x^{(j)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (w_k{}^{(i)})^2$$

- For n_u users:

$$\min_{w^{(i)},\ldots,w^{(n_u)}} \frac{1}{2} \sum_{i=1}^{n_u} \sum_{j:r(i,j)=1} ((w^{(i)})^T x^{(j)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_u} \sum_{k=1}^{n} (w_k{}^{(i)})^2$$

Matrix Factorization

- For find features vector of items:

$$\min_{x^{(j)},\dots,x^{(n_m)}} \frac{1}{2}\sum_{j=1}^{n_m}\sum_{i:r(i,j)=1}((w^{(i)})^T x^{(j)} - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_m}\sum_{k=1}^{n}(x_k{}^{(j)})^2$$

- For find both features vector of items and users:

$$J(x,w) = \frac{1}{2}\sum_{(i,j):r(i,j)=1}((w^{(i)})^T x^{(j)} - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{i=1}^{n_u}\sum_{k=1}^{n}(w_k{}^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_m}\sum_{k=1}^{n}(x_k{}^{(j)})^2$$

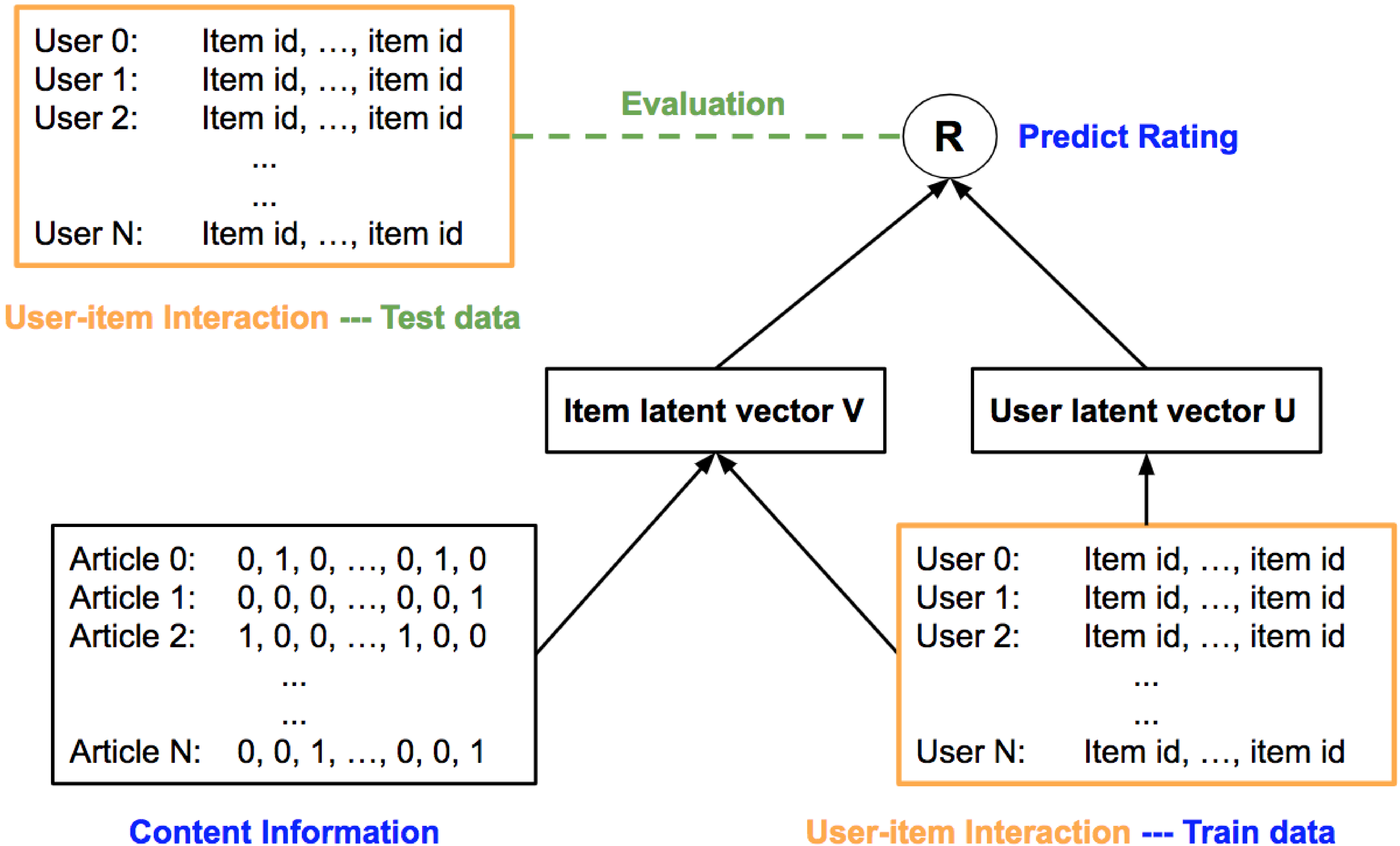
Comment

- Collaborative filtering result > Content-based result

- Issues of Collaborative filtering:
  - Cold-start problems
  - Sparsity problems

=> Hybrid methods: combine both Collaborative filtering and Content-based

Current State-of-the-art Models



Datasets

- Movie
  - Movielens: MovieID, Title, Genre, Rating, Tag
  - Netflix
- Scientific paper
  - CiteULike: Title, abstract, User-item interaction, tag…
  - CiteSeer
- Amazon product data

1. https://movielens.org/
2. http://www.citeulike.org/

Evaluation

- **Rating Prediction Task**
  - Mean Absolute Error (MAE)
  - Mean Square Error (MSE)
  - Root Mean Square Error (RMSE)
- **Top-N Recommendation Task**
  - Precision and Recall
  - Mean Average Precision (MAP)
  - Mean Reciprocal Rank (MRR)
  - Normalized Discounted Cumulative Gain (nDCG)

References

1. J. Bobadilla, F. Ortega, A. Hernando, and A. GutiéRrez. Recommender Systems Survey, 2013.
2. Ken Lang. NewsWeeder: Learning to Filter Netnews. In Proceedings of the 12th International Machine Learning Conference (ML95), 1995.
3. Ruslan Salakhutdinov andAndriy Mnih. Probabilistic Matrix Factorization. In Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07), 2007.
4. https://www.coursera.org/learn/machine-learning/lecture/uG59z/content-based-recommendations
5. https://machinelearningcoban.com

thank you
danke
謝謝
ngiyabonga
рахмат
teşekkür ederim
Баярлалаа
спасибо
faafetai lava
mersi
kia ora
barka
welalin
tack
spas
mahalo
tapadh leat
vinaka
спасибі
blagodaram
dank je
misaotra
matondo
paldies
grazzi
gracias
хвала
asante
manana
obrigada
tenki
kiitos
dankie
nanni
nandri
dhanyavad
mauruuru
köszönöm
bedankt
bayarlalaa
gràcie
hvala
enkosi
akun
dankon
aciū
djiere dieuf
tau
mochchakkeram
murakoze
дякую
mamnun
chokrane
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
go raibh maith agat
sobodi
děkuji
obrigado
mèsi
sagolun
sukriya
kop khun krap
ありがとう
arigatô
takk
dakujem
trugarez
didi madloba
najis tuke
tanemirt
grazie
rahmet
kam sah hamnida
rahmat
terima kasih
ευχαριστώ
diolch
dhanyavadagalu
shukriya
mercê
мерси
তোমাকে ধন্যবাদ
감사합니다
xièxie
merci