



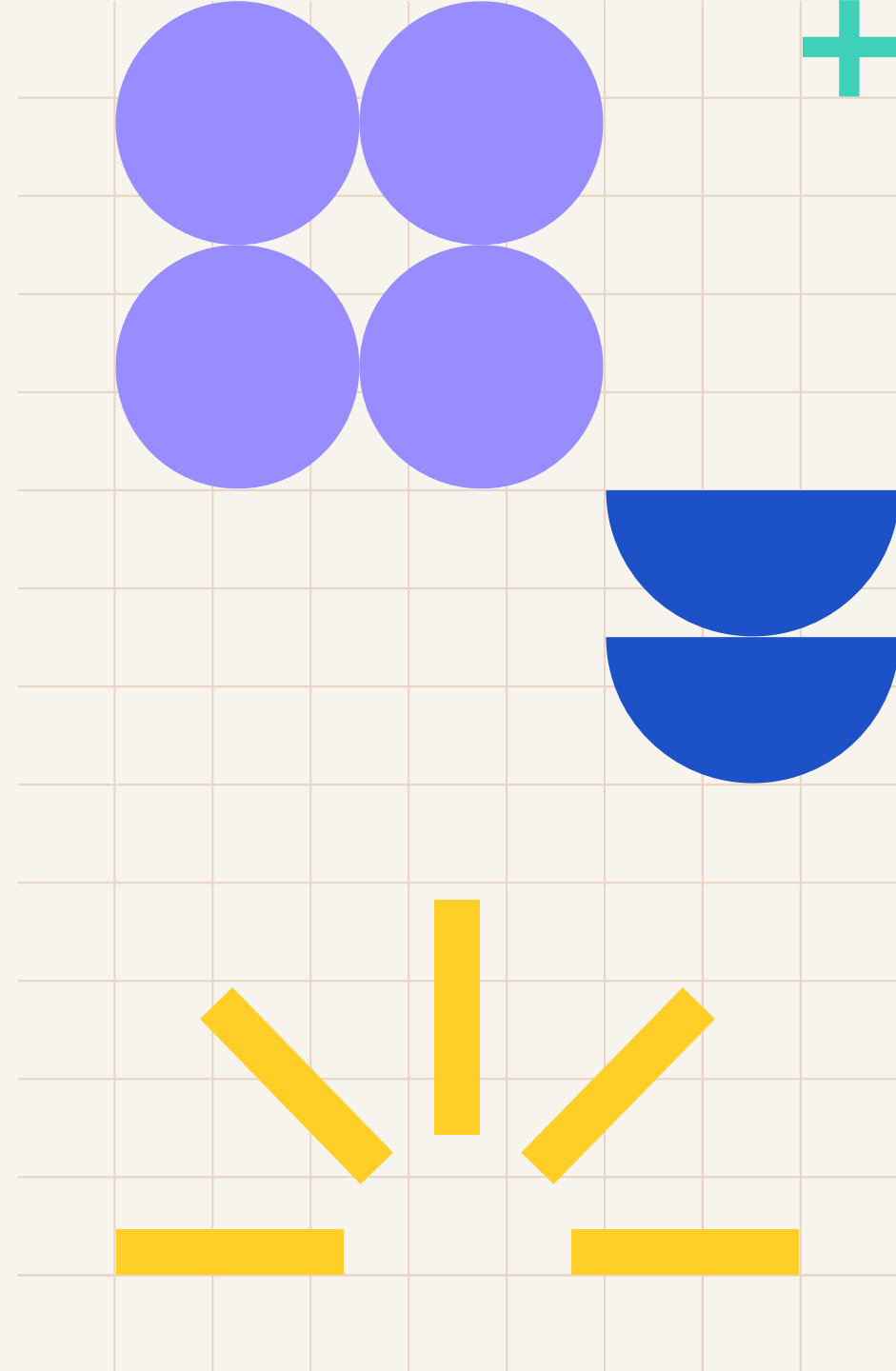
×



Practical algorithms for Recommender Systems

Evgeny Frolov

Research Scientist, Skoltech



About me

PhD in Data Science

<https://www.skoltech.ru/en/2018/09/phd-thesis-defense-evgeny-frolov/>

Currently:

- Research Scientist at Computational Intelligence Lab, Skoltech;
- Recommender Systems Research group lead

Previously:

- Sberbank AI Laboratory, head of recsys department
- Graduated from MSU, Chair of Polymer and Crystal Physics
- 5 years of professional experience in IT

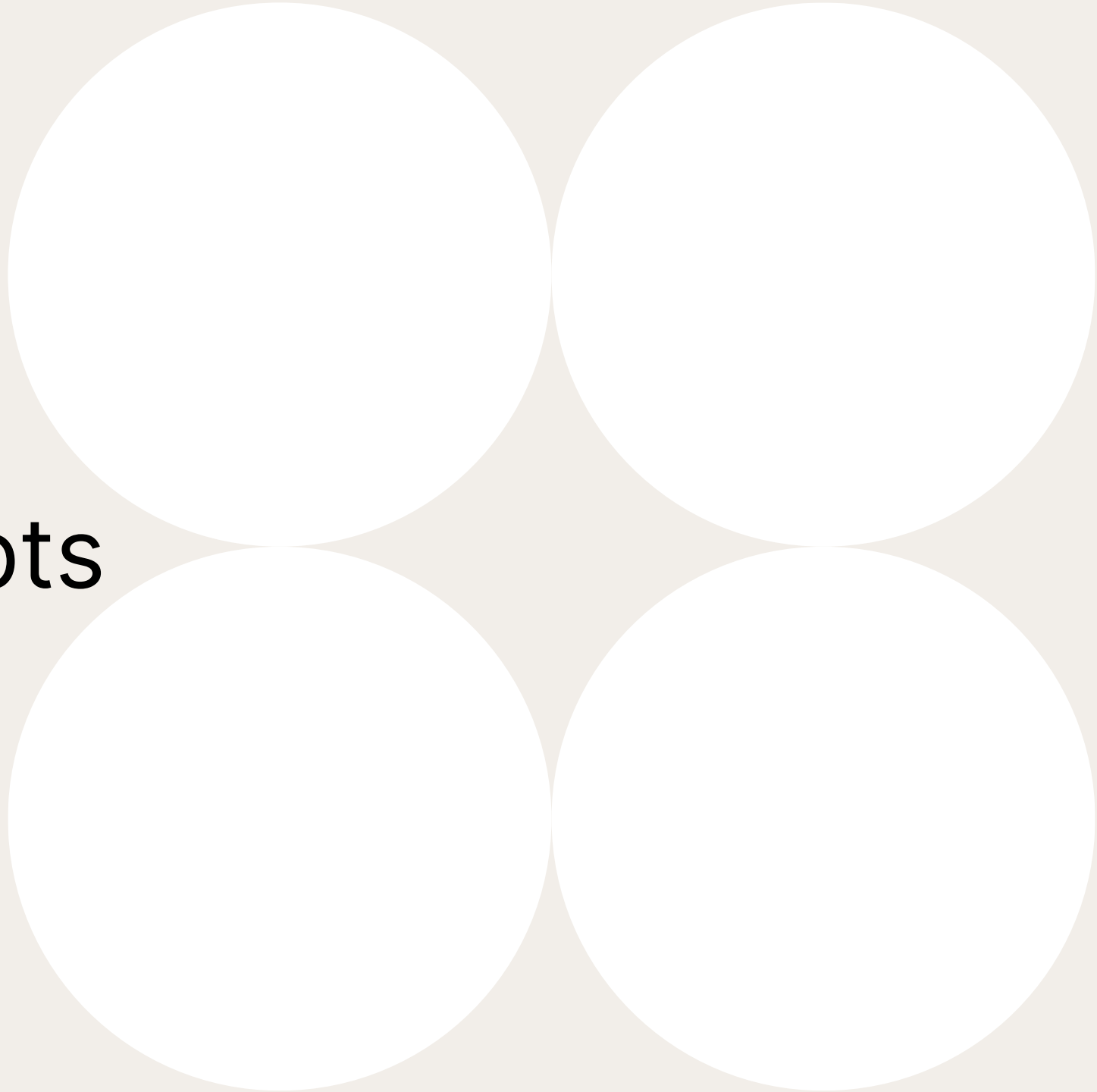
Today

- 01 General Concepts
- 02 Deep Learning in RecSys
- 03 Collaborative Filtering with Matrix Factorization

01



General Concepts



What is a recommender system?



Examples:

- Amazon
- Netflix
- Pandora
- Spotify
- Social platforms
- etc.

Many different areas: e-commerce, news, tourism, entertainment, education...

Goal: predict user preferences given some prior information on user behavior.

In a more general sense

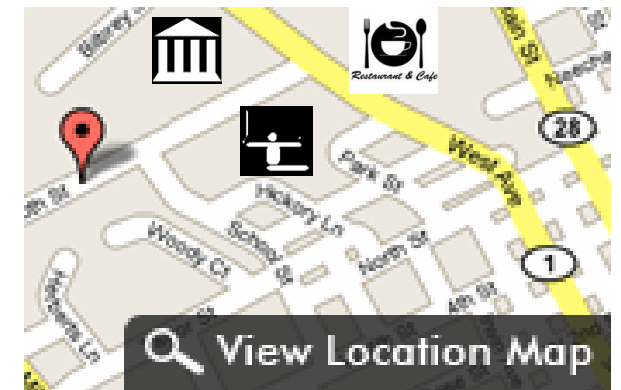
Recommender Systems aim to recover partially observed relations between two or more entities.

Sequential data:
item → next item
(order matters)

Social Networks:
user ↔ user



Ternary relations:
user → action → location



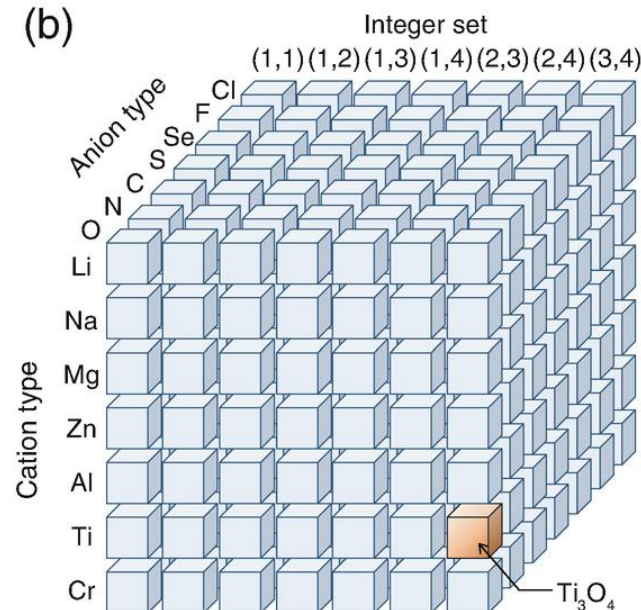
Material Discovery and Recommender Systems

Finding *chemically relevant compositions* and atomic arrangements of *inorganic compounds* using information from inorganic crystal structure databases.

(a)

$$\mathbf{X} = \begin{pmatrix}
 \text{Li}_2\text{O} & \text{Li}_2\text{O} & \text{Li}_2\text{O} & \text{Li}_2\text{O} & \text{Li}_2\text{O} & \dots & \text{In}_2\text{O}_3 & \text{In}_2\text{O}_3 & \text{In}_2\text{O}_3 & \text{In}_2\text{O}_3 & \text{In}_2\text{O}_3 & \text{Na} \\
 (1,1,1) & (1,1,2) & (1,2,2) & (1,3,3) & (1,4,3) & & (1,1,2) & (1,1,3) & (1,2,4) & (2,2,5) & (3,3,8) & \\
 0 & 0 & 0 & 0 & 0 & \dots & 1 & 0 & 0 & 0 & 0 & \\
 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 1 & 0 & 0 & \text{Mg} \\
 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & \text{Al} \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \\
 1 & 1 & 1 & 1 & 0 & \dots & 1 & 0 & 1 & 1 & 1 & \text{Cu} \\
 0 & 0 & 0 & 0 & 1 & \dots & 0 & 0 & 1 & 0 & 0 & \text{Zn} \\
 0 & 1 & 0 & 0 & 0 & \dots & 0 & 1 & 0 & 0 & 0 & \text{Ga}
 \end{pmatrix}$$

(b)



Discovered compounds and their stability

Composition	Predicted rating	Stability
RbInO ₂	1.01	○
Rb ₃ InO ₃	0.64	○
RbIn ₅ O ₈	0.20	—
NaGaS ₂	0.98	○
NaGa ₅ S ₈	0.21	—
KPbCl ₃	0.97	—
Ca ₂ TiO ₄	0.95	○
CaTi ₃ O ₇	0.29	—
Ca ₃ TiO ₅	0.21	—
BaAs ₂ O ₆	0.93	○
CsZnCl ₃	0.92	○
CsZn ₂ Cl ₅	0.25	○
RbInF ₄	0.91	○
Rb ₃ SbO ₃	0.91	○
CsY ₂ F ₇	0.87	○
Cs ₂ YF ₅	0.44	○
CsYF ₄	0.38	—
CsYS ₂	0.87	○
Ba ₂ Ga ₂ O ₅	0.85	○
BaGa ₄ O ₇	0.55	—
RbZnCl ₃	0.85	○
RbZn ₂ Cl ₅	0.23	—

Image Source

Seko, Atsuto, Hiroyuki Hayashi, Hisashi Kashima, and Isao Tanaka. "Recommender Systems for Materials Discovery." In *Machine Learning Meets Quantum Physics*, pp. 427-443. Springer, Cham, 2020.

Drug Discovery and Recommender Systems

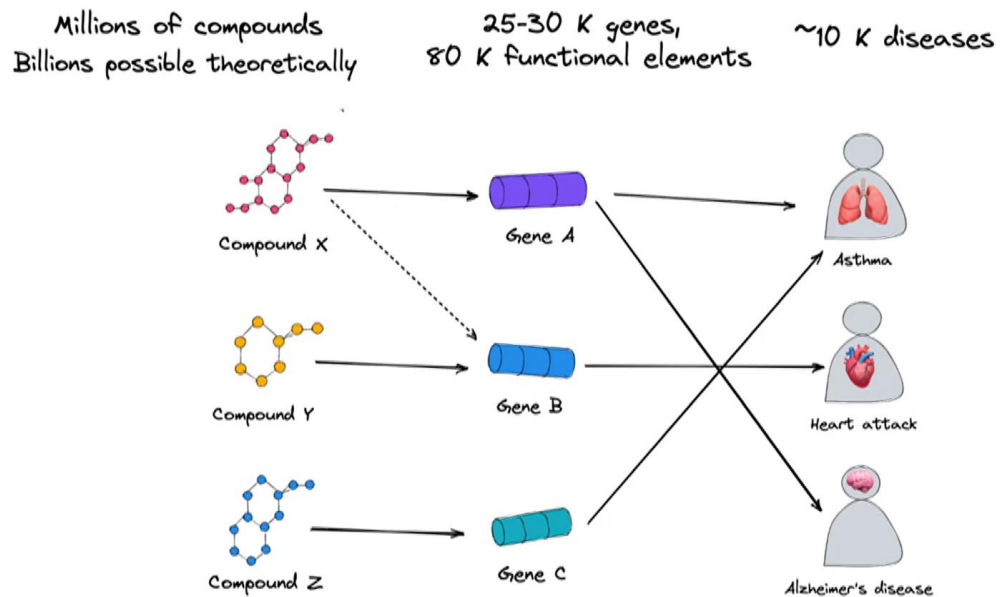


Image Source:

Gogleva, Anna, Eliseo Papa, Erik Jansson, and Greet De Baets. "Drug Discovery as a Recommendation Problem: Challenges and Complexities in Biological Decisions." In *Fifteenth ACM Conference on Recommender Systems*, pp. 548-550. 2021.

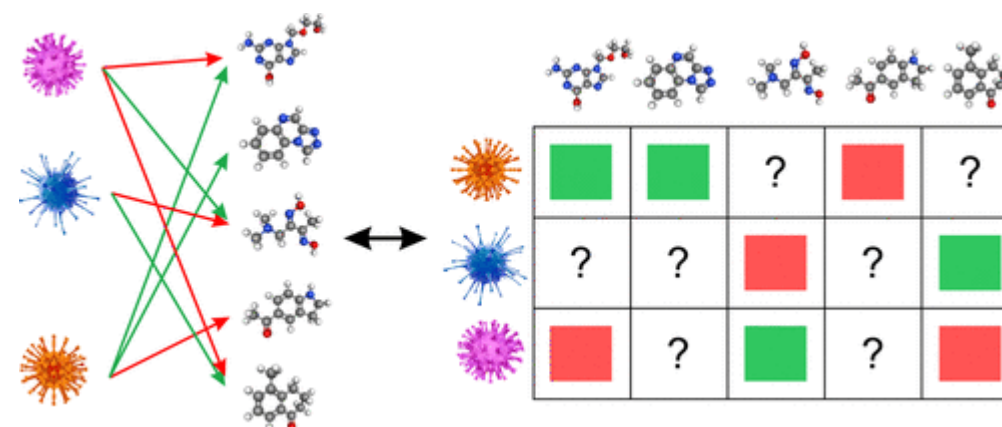
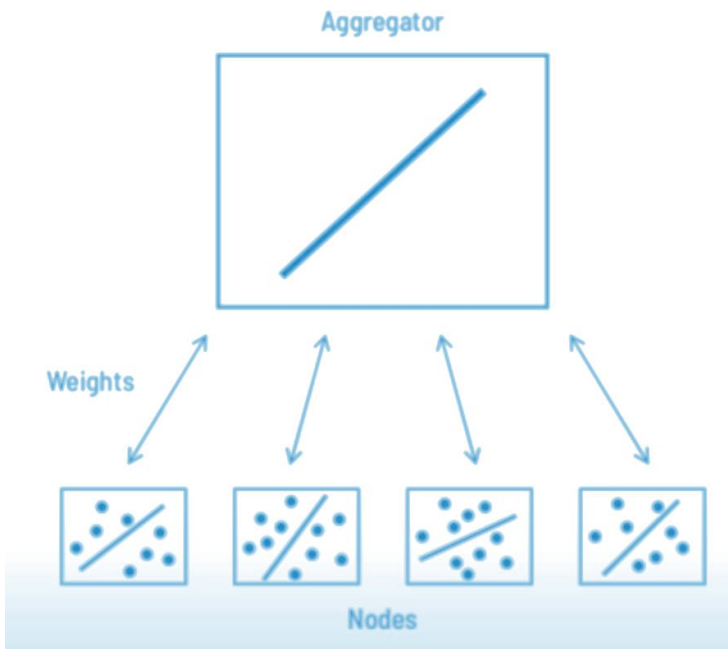


Image Source:

Sosnina, Ekaterina A., Sergey Sosnin, Anastasia A. Nikitina, Ivan Nazarov, Dmitry I. Osolodkin, and Maxim V. Fedorov. "Recommender systems in antiviral drug discovery." *ACS omega* 5, no. 25 (2020): 15039-15051.

Some of our Lab's projects

Federated Collaborative Filtering with Privacy



Hyperbolic Geometry in Recommender Systems



(a) M. C. Escher's Circle Limit III, 1959

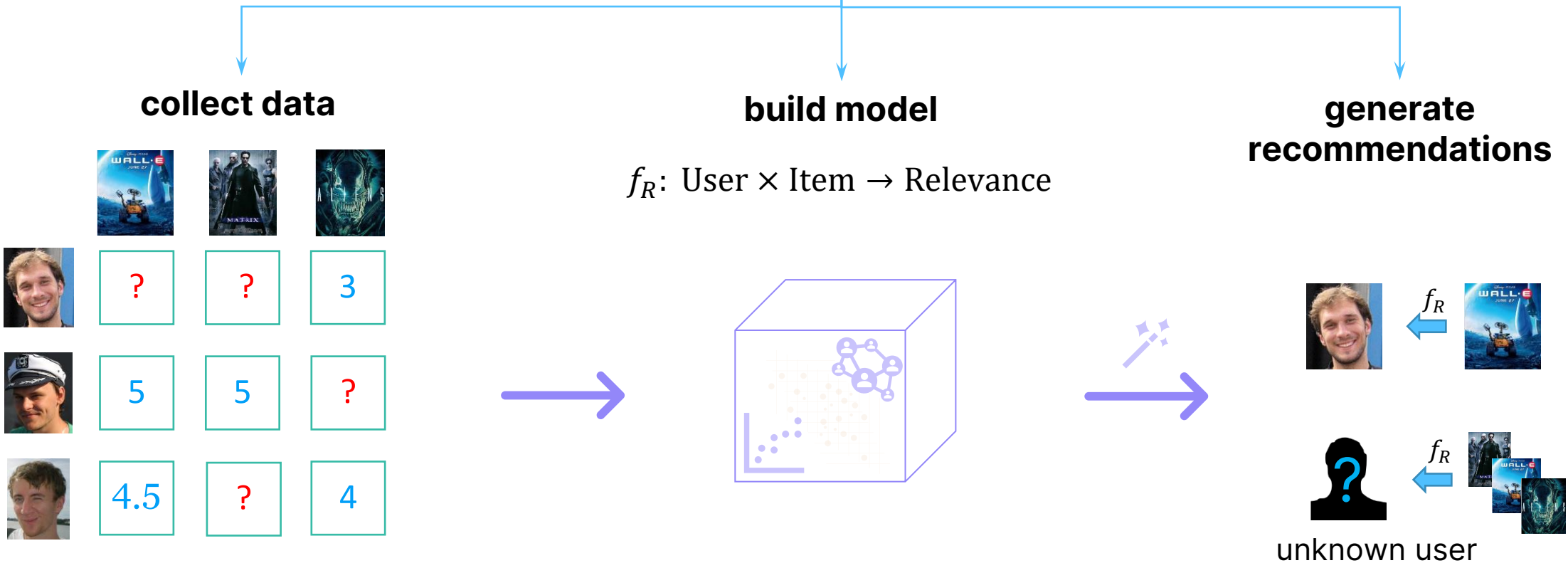
Fake Reviews detection

Q: get, shop, unfriendly, disrespect, tuxedo, disappoint, working, location, ridicule, shirt, new

A: I've been **shopping** for **tuxedos** here for years and I've **never** been treated with the **courtesy and respect** that I got from the staff here. The sales clerk has never even remembered my name and they have **no patience** for my inquiries. I've only ever had a good experience at the **other stores on the Pike**.

General workflow

Goal: predict user preferences based on prior user feedback and collective user behavior.



user-movie matrix A of size $M \times N$

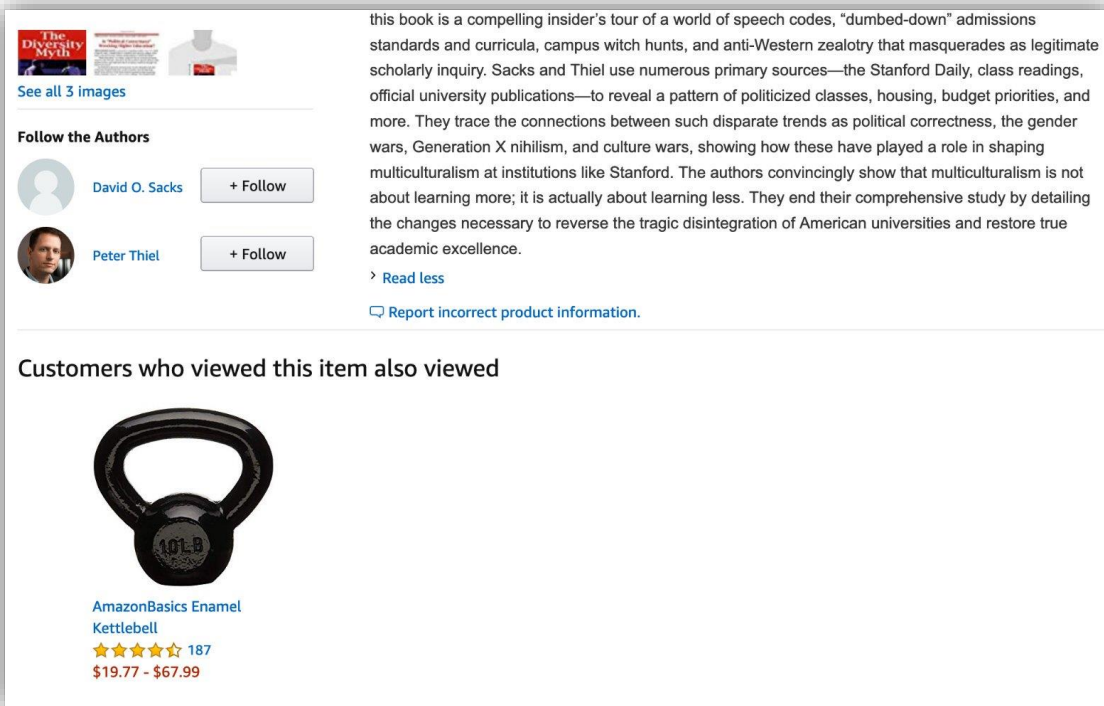
a_{ij} is a rating of i^{th} user for j^{th} movie

? - missing (unknown) values

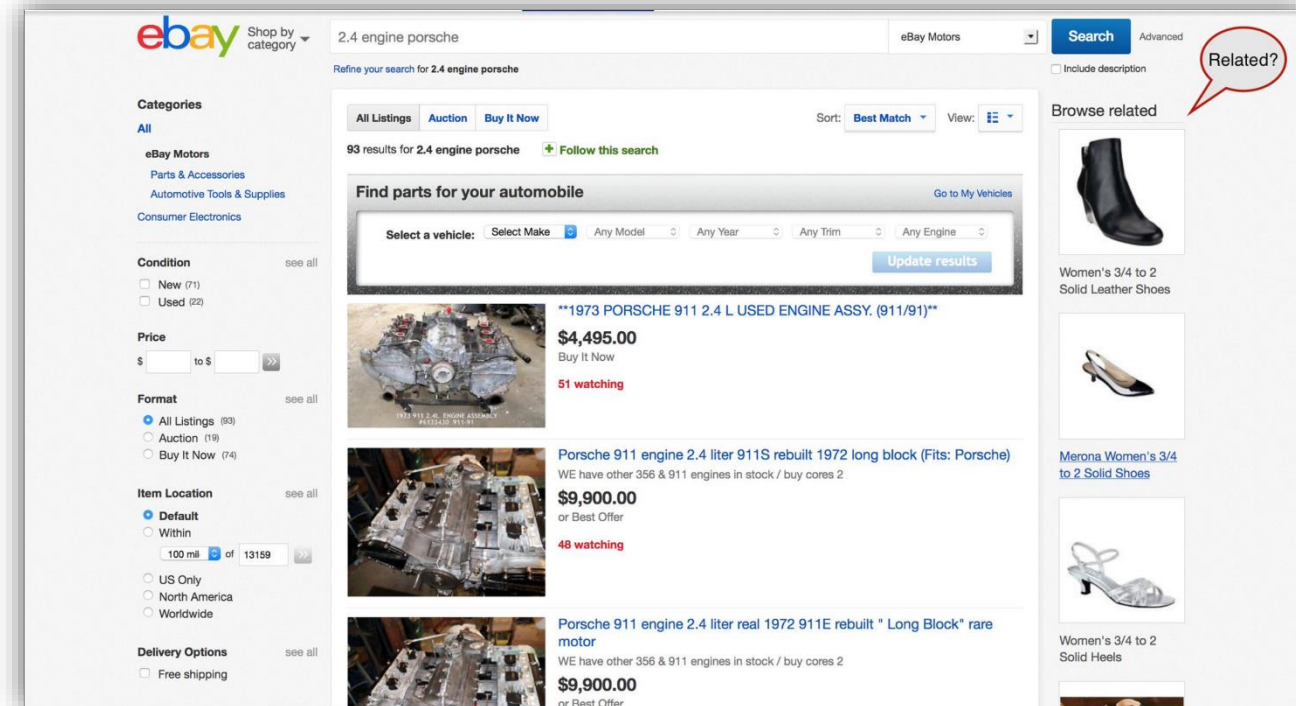
Building a good recommender system is difficult

“If you like *The Diversity Myth* book by Peter Thiel you may also like a kettlebell.”

“Need engine parts? You may find these women shoes relevant.”



Amazon product page for the book *The Diversity Myth* by Peter Thiel and David O. Sacks. The page shows the book cover, authors' profiles, and a description: "this book is a compelling insider's tour of a world of speech codes, 'dumbed-down' admissions standards and curricula, campus witch hunts, and anti-Western zealotry that masquerades as legitimate scholarly inquiry. Sacks and Thiel use numerous primary sources—the Stanford Daily, class readings, official university publications—to reveal a pattern of politicized classes, housing, budget priorities, and more. They trace the connections between such disparate trends as political correctness, the gender wars, Generation X nihilism, and culture wars, showing how these have played a role in shaping multiculturalism at institutions like Stanford. The authors convincingly show that multiculturalism is not about learning more; it is actually about learning less. They end their comprehensive study by detailing the changes necessary to reverse the tragic disintegration of American universities and restore true academic excellence." Below the description is a "Customers who viewed this item also viewed" section featuring an AmazonBasics Enamel Kettlebell with a price range of \$19.77 - \$67.99.



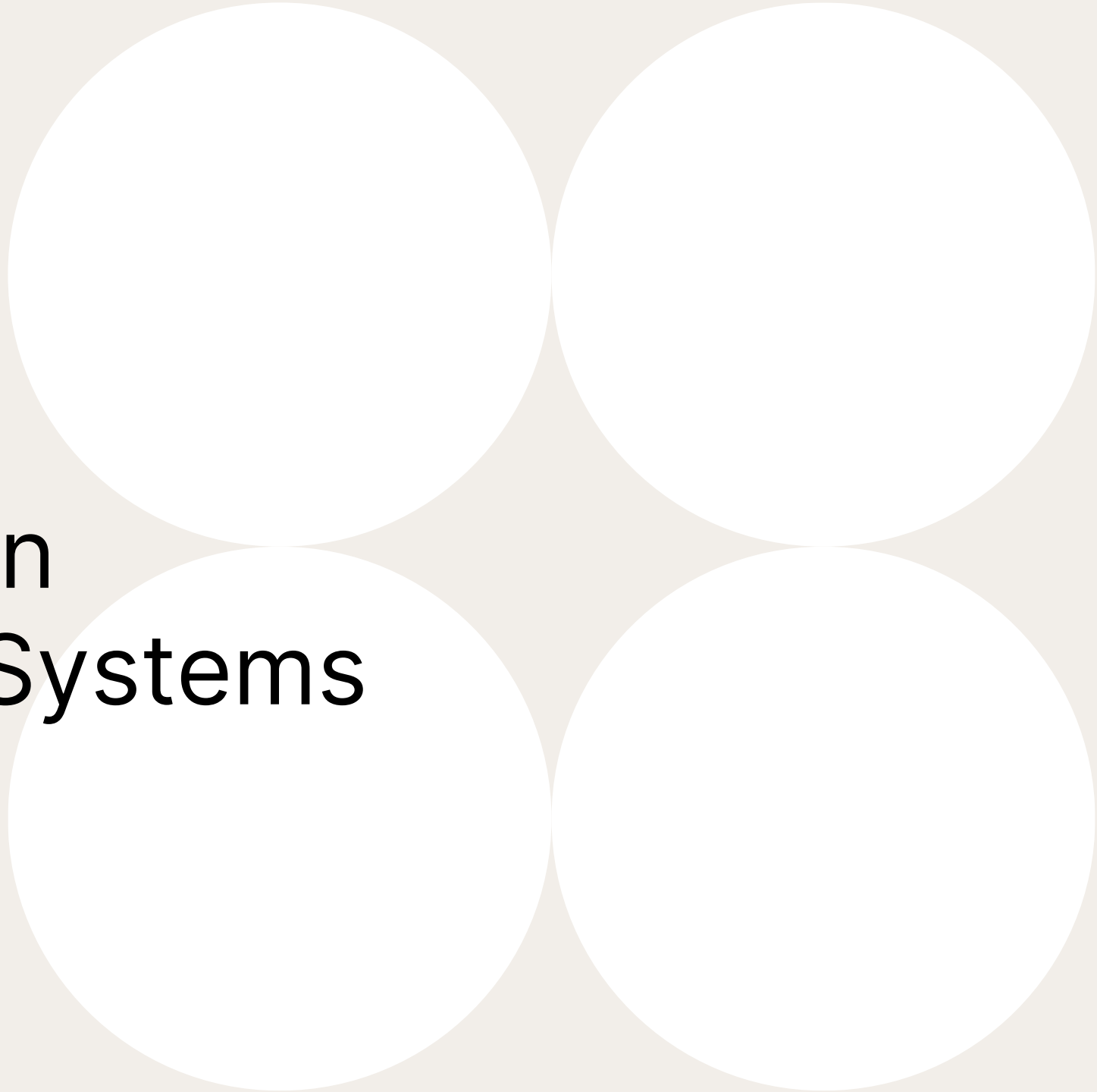
eBay search results for "2.4 engine porsche". The page shows search filters, categories, and a list of engine parts. A "Find parts for your automobile" section is visible. On the right, a "Browse related" section shows women's shoes, with a "Related?" callout bubble pointing to the shoes. The main search results include:

- **1973 PORSCHE 911 2.4 L USED ENGINE ASSY. (911/91)**: \$4,495.00, Buy It Now, 51 watching
- Porsche 911 engine 2.4 liter 911S rebuilt 1972 long block (Fits: Porsche): \$9,900.00 or Best Offer, 48 watching
- Porsche 911 engine 2.4 liter real 1972 911E rebuilt " Long Block" rare motor: \$9,900.00 or Best Offer

02



Deep Learning in Recommender Systems



FC feed-forward networks vs discrete relationships

Experiment:

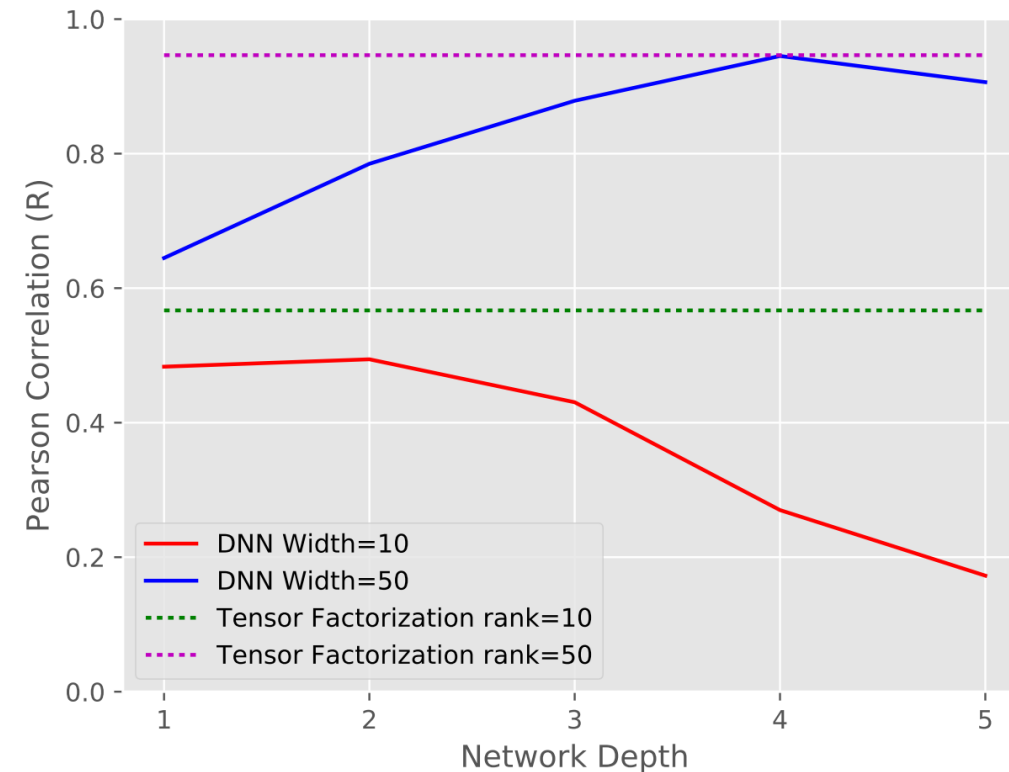
- approximate tensor \mathcal{X} of the form

$$x_{uic} = \langle v_{(u,c)}, v_i \rangle$$

u, i, c are (user, item, context) triplets

- TF baseline model is learned in the form

$$\langle v_u, v_i, v_c \rangle$$



FC feed-forward networks vs multiplicative relationships

Experiment:

- model multiplicative relations (“feature crosses”) between

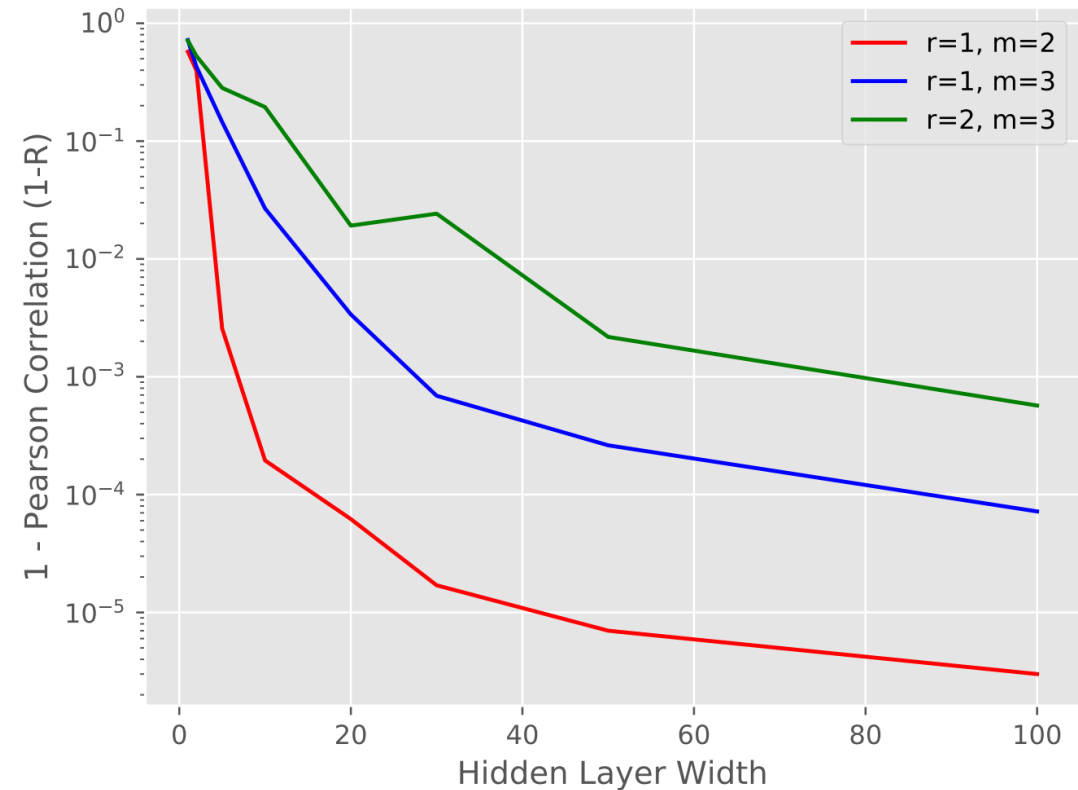
$$v_i \sim \mathcal{N}\left(0, \frac{1}{r^{0.5 \cdot m}} I\right)$$

r – dimensionality of hidden space

m – # of tensor modes in data

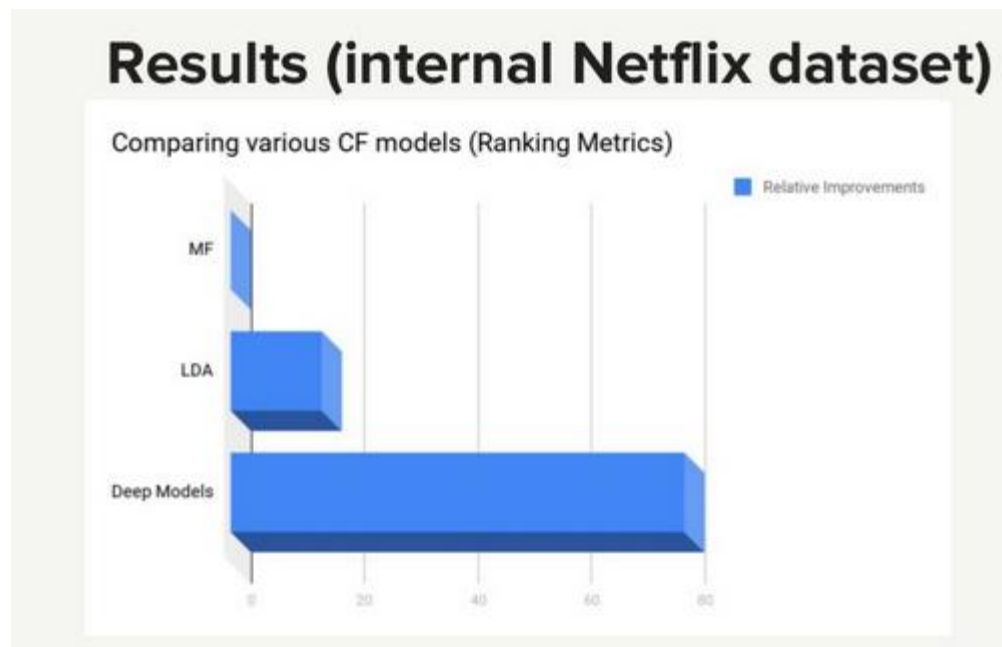
- network architecture:

$$h_\tau = g(W_\tau h_{\tau-1} + b_\tau)$$



About “the hype”

How it is presented

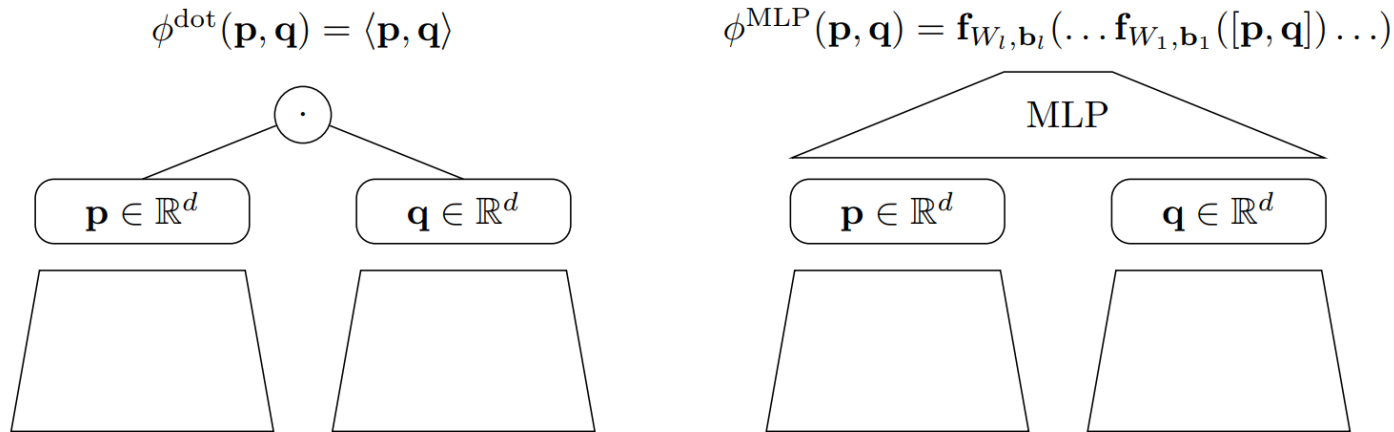


How it works in practice

Method	movielens			yahoo			pinterest		
	HR [%]	ARHR [%]	NDCG [%]	HR [%]	ARHR [%]	NDCG [%]	HR [%]	ARHR [%]	NDCG [%]
EigenRec	45.21	20.44	26.35	48.12	23.30	29.23	33.81	13.51	18.41
PureSVD	44.14	19.33	25.36	38.68	18.30	22.62	30.97	11.85	16.30
RP3b	34.87	15.02	19.66	41.51	17.82	22.94	27.01	8.07	12.45
SLIM	46.34	21.39	27.28	52.44	26.15	32.35	34.17	13.63	18.57
Mult-DAE	44.06	18.97	24.83	45.37	21.46	27.07	35.03	13.79	18.77
Mult-VAE	44.35	19.50	25.31	45.09	21.22	26.80	35.13	13.73	18.71

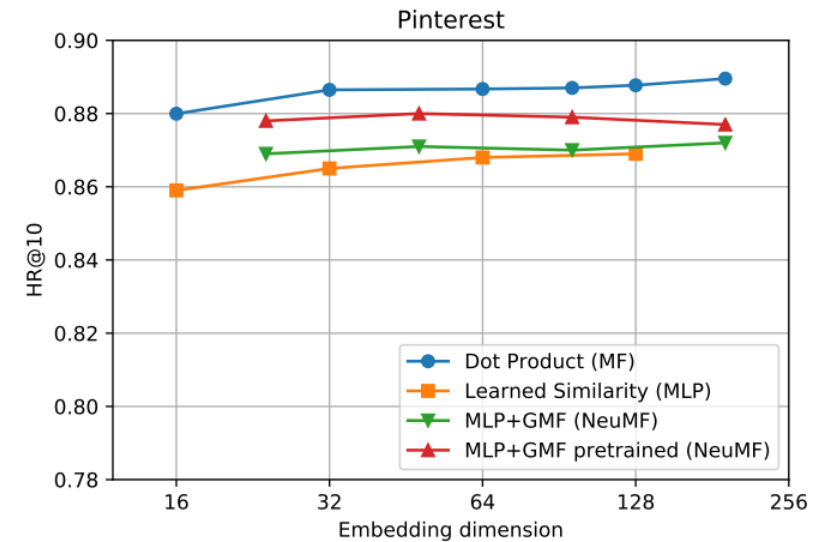
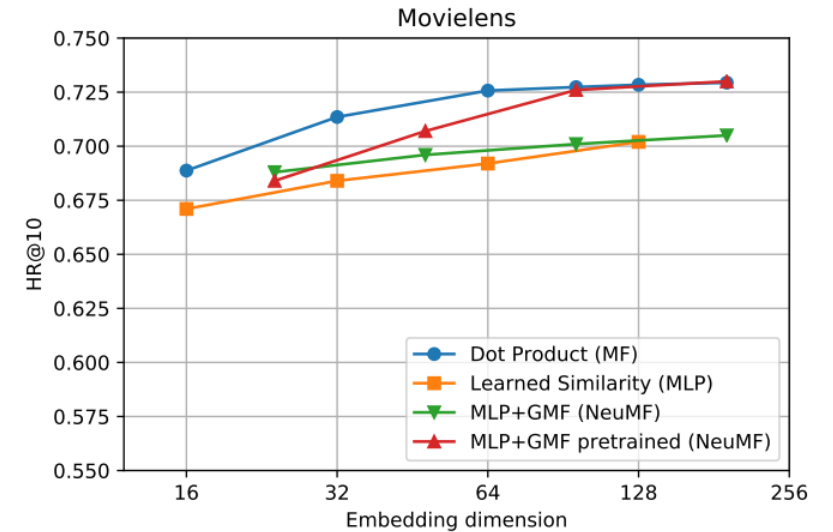
MF

MLP vs dot product



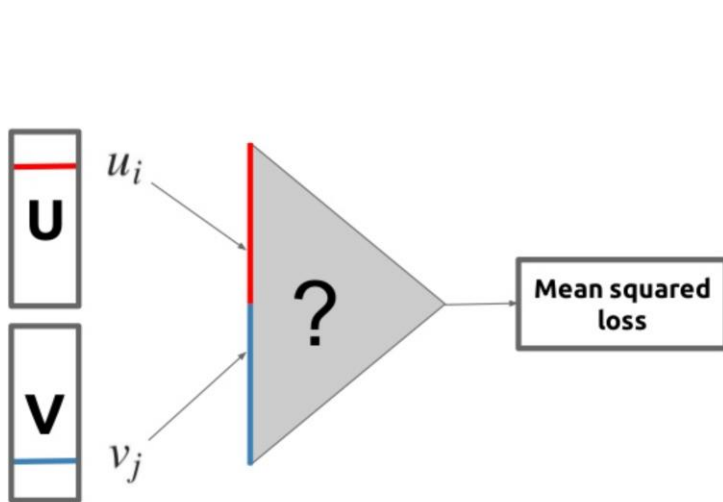
fully connected feed-forward networks seem to be not good at learning multiplicative relationships (like dot products)

Source: Rendle, Steffen, Walid Krichene, Li Zhang, and John Anderson. "Neural collaborative filtering vs. matrix factorization revisited." In *Fourteenth ACM conference on recommender systems*, pp. 240-248. 2020.

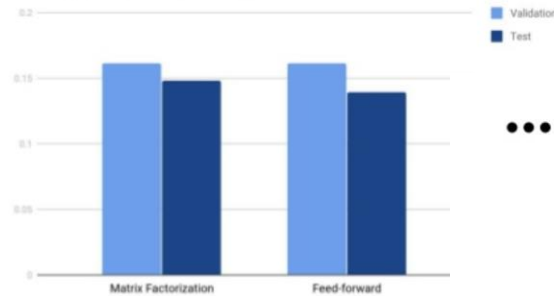


Netflix experience

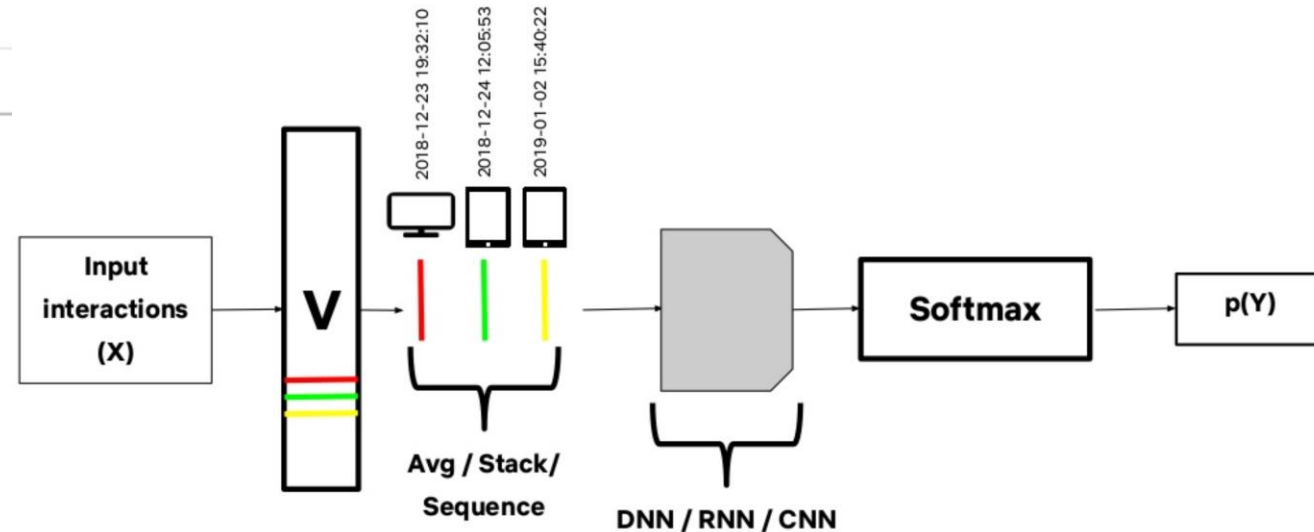
... isn't always the best



Ranking quality of selected model

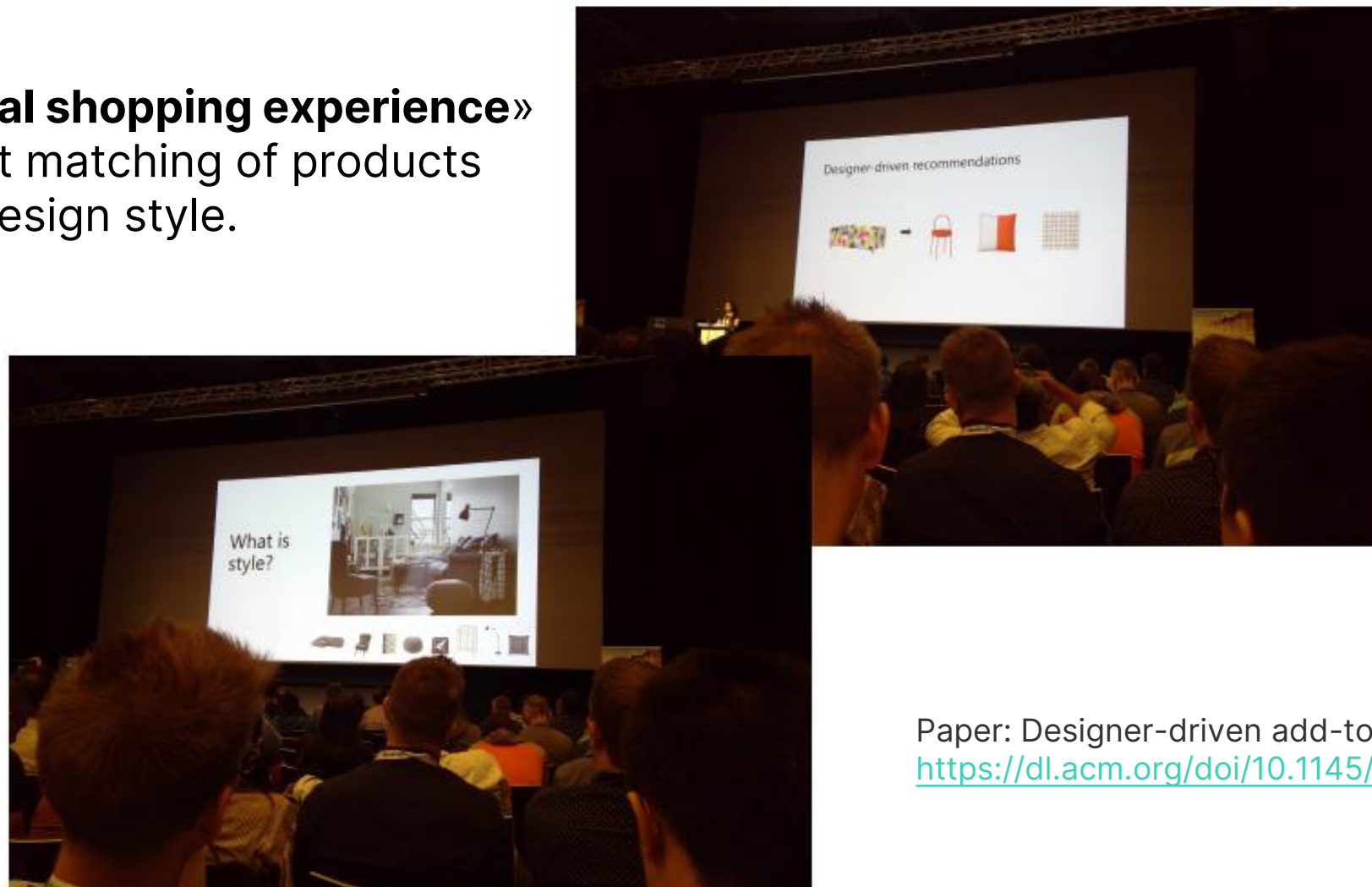


... but opens up many possibilities



Example 3 - IKEA

Selling «**inspirational shopping experience**» based on intelligent matching of products within a common design style.



Paper: Designer-driven add-to-cart recommendations
<https://dl.acm.org/doi/10.1145/3298689.3346959>

DNN's are also good for sequential modeling

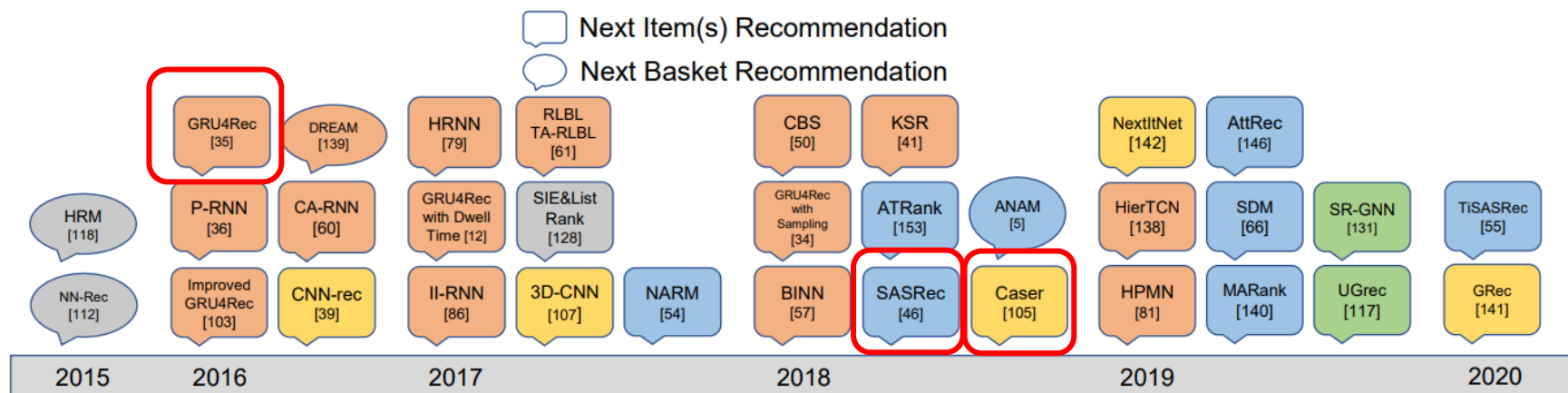
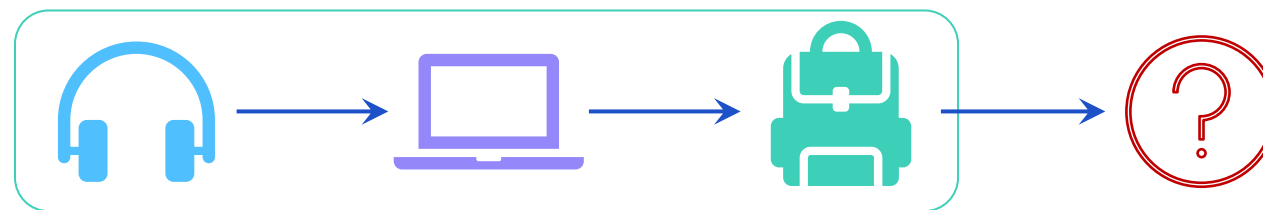


Fig. 8. Some recent and representative DL-based sequential recommendation models. Different colors indicate different DL techniques (grey: MLP; orange: RNN; yellow: CNN; blue: attention mechanism; green: GNN).

[Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations](#)

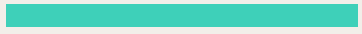
How recsys is different from NLP?

- **BERT for NLP:**
 - vocabulary size is 30K x 1024,
 - compute makes up almost the entire workload.
- **Transformer-based recsys:**
 - equivalent 'vocabulary' is 300M users and 12M items with dimension 64,
 - don't fit on a single GPU, heavily IO-bound.

...inference for recommender systems in production still happens on CPU because GPUs don't offer the same speedups that we see in other domains out of the box...

Even Oldridge, research scientist at NVidia

03



Collaborative Filtering with Matrix Factorization

A general view on latent factors models

- **Task:** find utility (relevance) function f_R :

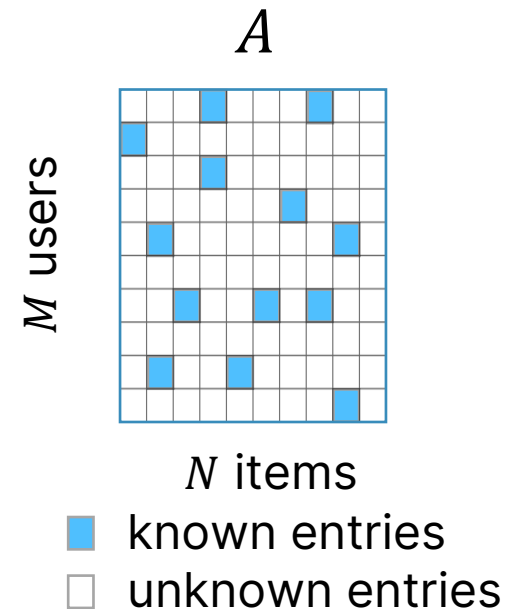
$$f_R: \text{Users} \times \text{Items} \rightarrow \text{Relevance score}$$

- As optimization problem with some *loss function* \mathcal{L} :

$$\mathcal{L}(A, R) \rightarrow \min$$

- **Components of the model:**

- Utility function to generate R
- Optimization objective defined by \mathcal{L}
- Optimization method (algorithm)



Intuition behind MF







Assumption: observed interactions can be explained via

- a *small* number of common patterns in human behavior
- + individual variations (including random and unobservable factors)

This is a recipe for low rank approximation:

$$A_{full} = R + E, \quad R = PQ^T$$

Low rank representation

					
					
					
		?	3	5	5
		4	?	5	5

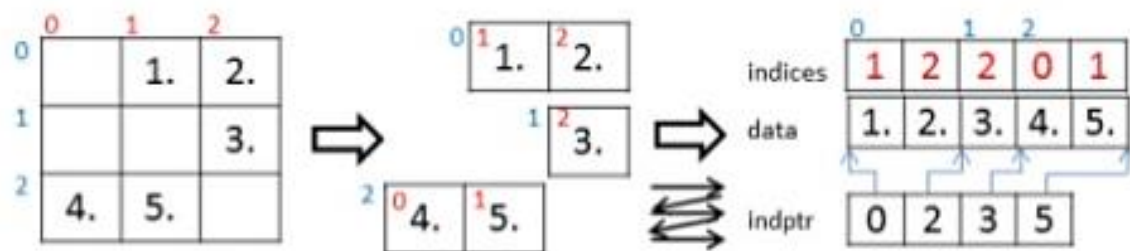
4	3	?		
?	3	5		
4	?	5		
			4	5
			?	5

How to work with large ratings matrix?

$M = 1_000_000$ users, $N = 100_000$ items would require ≈ 745 **Gb of RAM**.

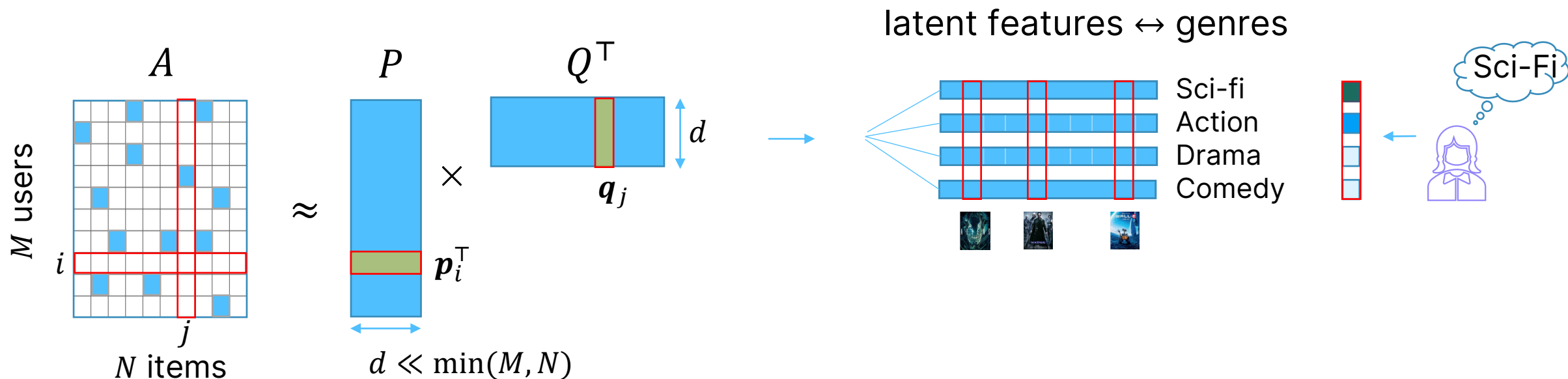
How to avoid explicitly forming it?

Internal structure: `csr_matrix`



Images source: <http://www.slideshare.net/hamukazu/effective-numerical-computation-in-num-py-and-scipy>

Simplistic view on latent features



\mathbf{p}_i – latent factors vector for user i (user embedding)

\mathbf{q}_j – latent factors vector for item j (item embedding)

Predicted relevance score (utility) of item j for user i :

$$r_{ij} \approx \mathbf{p}_i^T \mathbf{q}_j = \sum_{k=1}^d p_{ik} q_{jk}$$


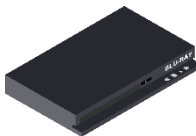

Top- n recommendations task

Expected output: a ranked list of n most relevant items.

$$\text{toprec}(i, n) := \arg \max_j^n r_{ij}$$

r_{ij} - is the predicted relevance score between user i and item j

Example:

r_{ij}	0.73	0.41	0.95
item			

top-2 recommendations



leftsorted list



Suggest a matrix factorization approach

Quick reminder on Singular Value Decomposition

$$A = U\Sigma V^T$$

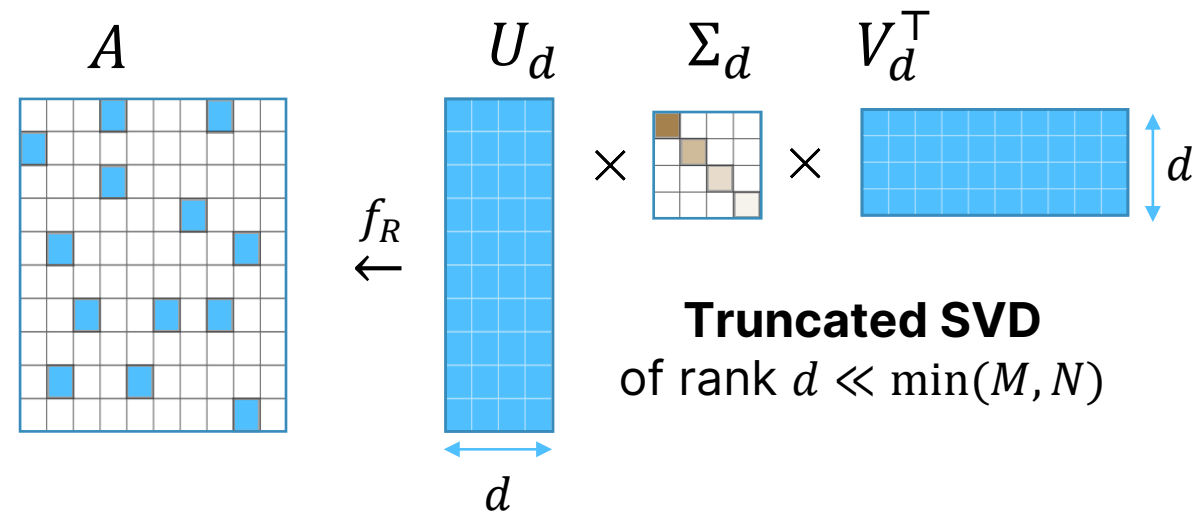
$$U \in \mathbb{R}^{M \times M}, \quad V \in \mathbb{R}^{N \times N}$$

$$U^T U = I_M, \quad V^T V = I_N$$

$\Sigma \in \mathbb{R}^{M \times N}$ - diagonal, with $[\Sigma]_{kk} = \sigma_k$:

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(M,N)} \geq 0$$

$$\sigma_k(A) = \sqrt{\lambda_k(A^T A)} = \sqrt{\lambda_k(AA^T)}$$



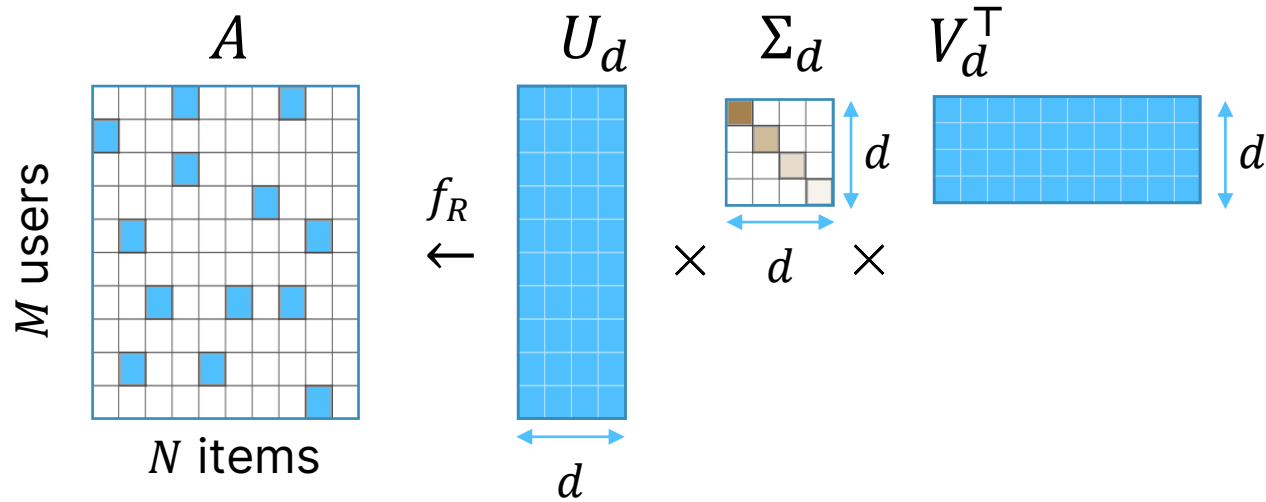
Low-rank approximation task:

$$\|A - R\|_F^2 \rightarrow \min, \text{ s.t. } \text{rank}(R) = d$$

$$R = U_d \Sigma_d V_d^T, \quad \|A - R\|_F^2 =$$

Undefined for incomplete matrix!

PureSVD model for CF



Relevance score prediction:

$$A_0 V_d V_d^T =$$

Let's impute zeros in place of unknowns!

$$A_0 = U \Sigma V^T, \quad [A_0]_{ij} = \begin{cases} a_{ij}, & \text{if known} \\ 0, & \text{otherwise} \end{cases}$$

$$R = U_d \Sigma_d V_d^T$$

PureSVD computation

Efficient computation with Lanczos algorithm:

- iterative process
- requires only sparse matrix-vector (matvec) multiplications (fast with CSR format);
- training complexity $O(nnz \cdot d) + O((M + N) \cdot d^2)$

Efficient implementations in Python:

- SciPy Sparse svds, Scikit-Learn TruncatedSVD.
- core functionality is also implemented in Spark.

```
In [1]: import numpy as np
        from scipy.sparse import csr_matrix
        from scipy.sparse.linalg import svds
```

```
In [2]: # convert sparse matrix into efficient CSR format
        A = csr_matrix([[0, 1, 1, 0, 0, 0],
                        [0, 1, 0, 1, 0, 0],
                        [0, 0, 1, 0, 1, 1],
                        [0, 1, 0, 1, 0, 1]], dtype=np.float64)
        A
```

```
Out[2]: <4x6 sparse matrix of type '<type 'numpy.float64''>'
         with 10 stored elements in Compressed Sparse Row format>
```

```
In [3]: # compute sparse SVD of rank 2
        rank = 2
        U, S, Vt = svds(A, k=rank)
```

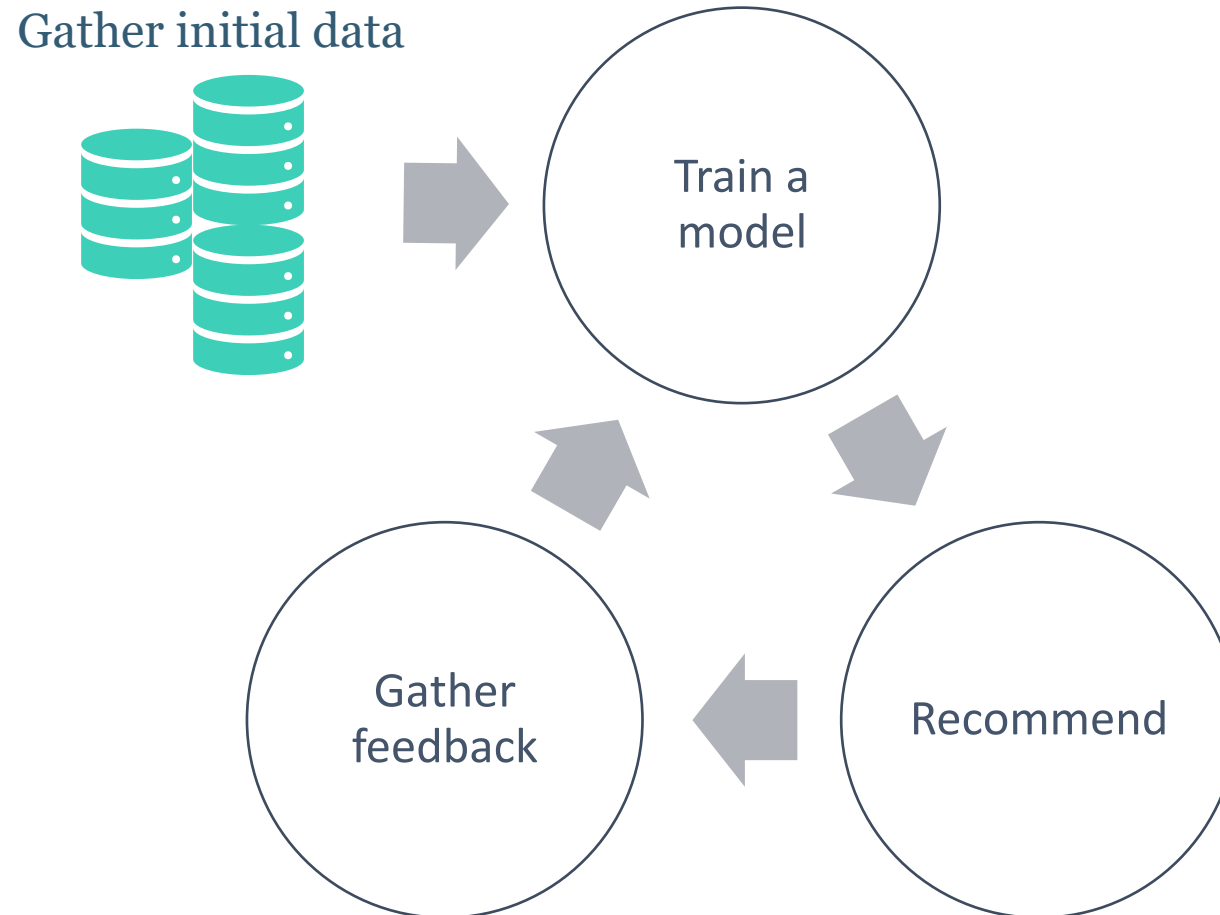
```
In [4]: # check orthogonality
        np.testing.assert_almost_equal(U.T.dot(U), np.eye(rank), decimal=15)
        np.testing.assert_almost_equal(Vt.dot(Vt.T), np.eye(rank), decimal=15)
```

Explaining recommendations

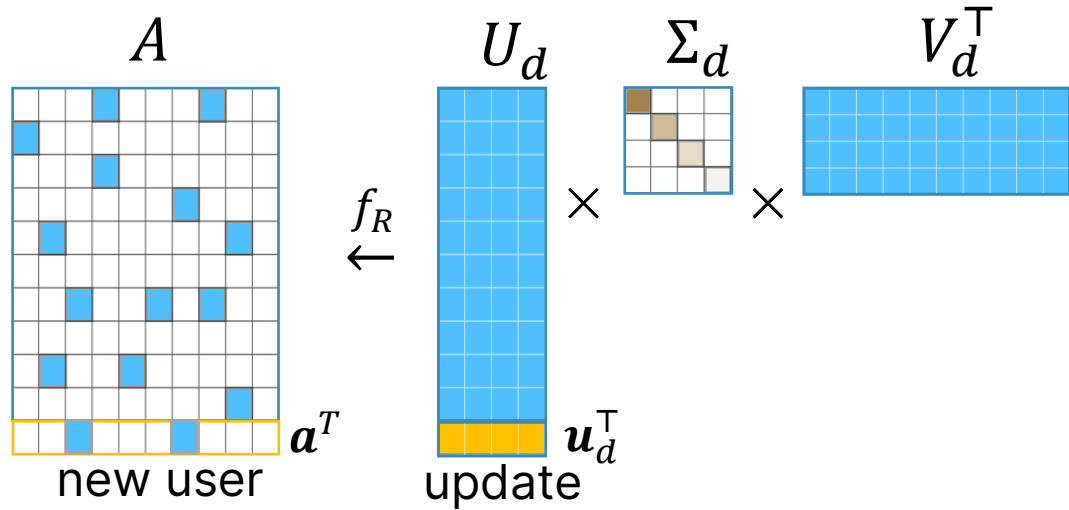
Which one is more explainable?

$$\mathbf{r} = Q\mathbf{p} \quad \text{vs.} \quad \mathbf{r} = VV^T\mathbf{a}$$

Lifecycle of a recsys model



PureSVD – recommending online



*folding-in technique**

*G. Furnas, S. Deerwester, and S. Dumais, "Information Retrieval Using a Singular Value Decomposition Model of Latent Semantic Structure," Proceedings of ACM SIGIR Conference, 1988

Finding a warm-start user representation:

$$\| \mathbf{a}_0^T - \mathbf{u}^T \Sigma V^T \|_2^2 \rightarrow \min$$

new user embedding

$$\mathbf{u}^T = \mathbf{a}_0^T V \Sigma^{-1}$$

Prediction:

$$\mathbf{r}^T = \mathbf{u}^T \Sigma_d V_d^T = \mathbf{a}_0^T V_d V_d^T$$

$$\mathbf{r} = V_d V_d^T \mathbf{a}_0$$

- convenient for evaluation
- complexity $\sim O(Nd)$
- enables real-time recommendations

Approximation error estimates

$$A = \mathbf{1} \cdot \mathbf{a}^\top + \epsilon B$$

$$\sigma_1^2(A) \leq M \|\mathbf{a}\|^2 + \epsilon^2 N, \quad \sigma_i^2(A) \sim \epsilon^2 N, i > 1$$



?	3	5	5
4	?	5	5

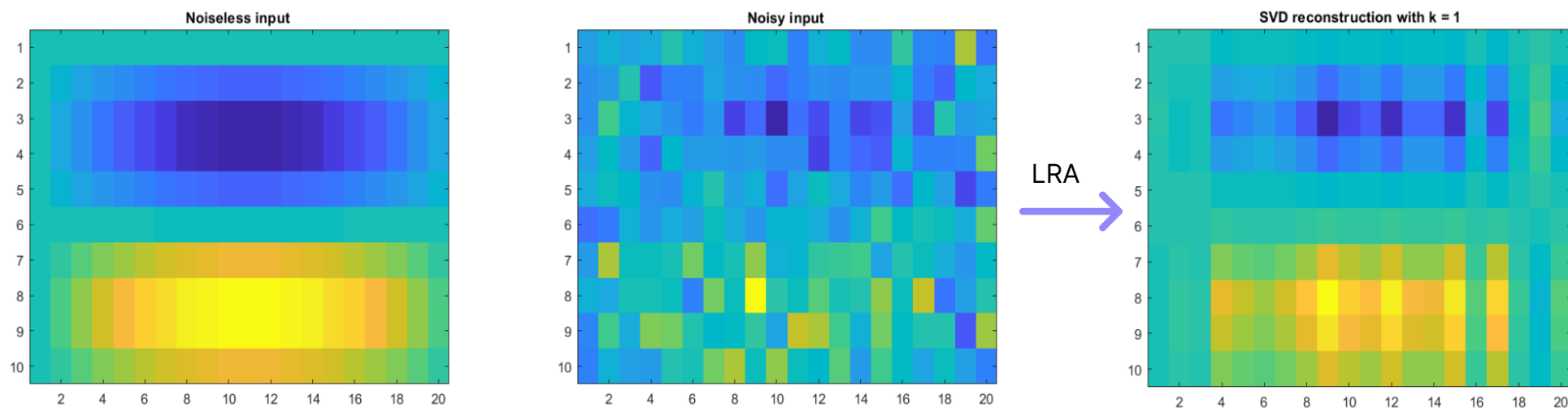
Practical consequences:

- even in the simplest case singular values won't become 0
- no simple choice of the optimal rank of the decomposition

- $\|A - U_d \Sigma_d V_d^\top\|_F^2 = \sqrt{\sigma_{d+1}^2 + \dots + \sigma_{\min(M,N)}^2}$

- doesn't mean you can't get close to zero RMSE on trainset

CF as low-rank approximation task



unknown in reality

Images source: Khoshrou, Abdolrahman, and Eric J. Pauwels. "Regularisation for PCA-and SVD-type matrix factorisations." *arXiv preprint arXiv:2106.12955* (2021).

Data centering (PCA style)

Observations:

- in PureSVD, values are highly biased towards 0
- from rating prediction perspective: a signal is carried mostly by baseline estimators / average values

How does it affect rating prediction?

Strategy:

- value imputation → mean shifted matrix
- akin to data centering in PCA

Mean-shifted ratings matrix spectrum

$$A_{\text{full}} = \tilde{A}_0 + \alpha \mathbb{1} \mathbb{1}^\top$$

$$\sigma_1^2(A_{\text{full}}) = \sigma_1^2(\tilde{A}_0) + \alpha^2 MN, \quad \sigma_i^2(A_{\text{full}}) \approx \sigma_i^2(\tilde{A}_0), i > 1$$

Would it improve quality of recommendations?

Verifying it poses the same scalability problem:

$M = 1_000_000$ users, $N = 100_000$ items would require \approx **745 Gb of RAM.**

How to avoid it?

Practical consequences

What MF for Collaborative Filtering is not:

- pure matrix completion
- pure dimensionality reduction

Common PCA-like preprocessing may spoil data representation!

Rating prediction doesn't make sense:

- recommendations can still be good without accurate rating estimation!
- we can treat rating values more flexibly

Billion-scale computations

In practice, in distributed setups, randomized SVD is used.

Examples:

- Criteo <https://github.com/criteo/Spark-RSVD>
- Facebook's randomized SVD implementation <https://research.fb.com/fast-randomized-svd>

Research:

- “out-of-memory” SVD [Kabir 2017]
- communication-avoiding algebra [Demmel 2008]
- DeepMind’s attempt to adapt to modern hardware (GPU, TPU) via game-theoretic approach <https://www.deepmind.com/blog/game-theory-as-an-engine-for-large-scale-data-analysis>

Streaming and incremental learning

Incremental learning:

- *Adding new users/items:*

- rank-1 updates (see G. Golub, C. Van Loan, “Matrix Computations”)
- M. Brand "Incremental singular value decomposition of uncertain data with missing values.", 2002.

- *Adding new interactions:*

- Can be done via projector splitting approach [Lubich & Oseledets 2013]
- Example in recsys: [Olaleke et al. 2021]

Streaming:

- Method of frequent directions [Ghashami et al. 2016]
- Zoom SVD [Jang et al. 2018]

More general MF optimization scheme

Optimization objective:

$$\mathcal{J}(\Theta) = \mathcal{L}(A, \Theta) + \Omega(\Theta)$$

Model parameters: $\Theta = \{P, Q\}$

$\Omega(\Theta)$ - additional constraints, e.g. L_2 regularization

Typical optimization algorithms:

stochastic gradient descent (SGD)

alternating least squares (ALS)

ALS:

$$\begin{cases} P^* = \arg \min_P \mathcal{J}(\Theta) \\ Q^* = \arg \min_Q \mathcal{J}(\Theta) \end{cases}$$

GD:

$$\begin{cases} \mathbf{p}_i \leftarrow \mathbf{p}_i - \eta \nabla_{\mathbf{p}_i} \mathcal{J} \\ \mathbf{q}_j \leftarrow \mathbf{q}_j - \eta \nabla_{\mathbf{q}_j} \mathcal{J} \end{cases}$$

ALS vs SGD vs SVD

ALS

- More stable
- Fewer hyper-parameters to tune
- Higher complexity, however requires fewer iterations
- Embarrassingly parallel
- Higher communication cost in distributed environment
- Coordinate Descent can be a good alternative (e.g., eALS by [He et al. 2016])

SGD

- Sensitive to hyper-parameters
- Requires special treatment of learning rate
- Lower complexity but slower convergence, using adaptive learning rate schedule (ADAM, Adagard, etc.) helps
- Inherently sequential (parallelization is tricky for RecSys)
- Hogwild! algorithm is not directly applicable in CF settings

Algorithm	Overall complexity	Update complexity	Sensitivity
SVD*	$O(nnz_A \cdot r + (M + N)r^2)$	$O(nnz_a \cdot r)$	Stable
ALS	$O(nnz_A \cdot r^2 + (M + N)r^3)$	$O(nnz_a \cdot r + r^3)$	Stable
CD	$O(nnz_A \cdot r)$	$O(nnz_a \cdot r)$	Stable
SGD	$O(nnz_A \cdot r)$	$O(nnz_a \cdot r)$	Sensitive

* For both standard and randomized implementations [71].

(ALS, SGD) vs SVD:

- **More involved optimization (no rank truncation).**
- **Allow for custom optimization objectives.**

General rule of thumb: avoid distributed setups.

Working with imbalanced data

- SGD:
 - negative sampling
- iALS:
 - confidence weights

$$\mathcal{L} = \sum_{ij} w(a_{ij}) \cdot l(s_{ij} - r_{ij})^2$$

- PureSVD
 - data normalization

$$\tilde{A} = DA_0, \quad [D]_{ii} = \|a_i\|^{f-1}$$

All these methods aim to balance contribution of positive and negative items!

Case study: Yandex Zen

Company manages many different types of media content (news, search, etc.).

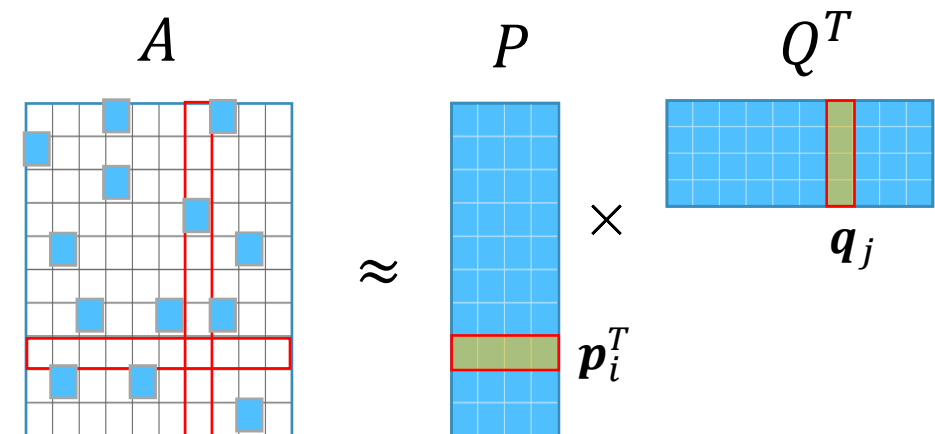
Goal: have a unified user representation across all domains.

Solution:

- A neural network embeds onto a latent space all unstructured content
- Users are updated through the “half”-ALS scheme

Algorithm:






- Get Q from external source (DNN)
- Update P based on most recent Q



Case study: Yandex Music

Составим матрицу оценок пользователей:

- › По строчкам – пользователи
- › По столбцам – треки
- › На пересечении пользователя и трека будет оценка пользователя – понравился трек или нет

<https://academy.yandex.ru/posts/kholodnye-polzovateli-i-mnogorukie-bandity>

Need to take into account different signals! For example:

- number of seconds a user spent listening to a track
- number of skips / fast forwards
- adding to playlist / favorites
- ...

Solved by iALS / WRMF model

$$\mathcal{L} = \sum_{ij} w(a_{ij}) \cdot l(s_{ij} - r_{ij})^2$$

Some recent trends

Multi-task learning (RecSys 2020 best paper)

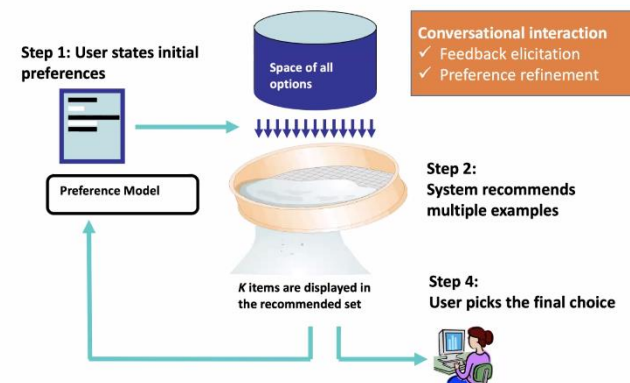
Causality

Fairness and debiasing

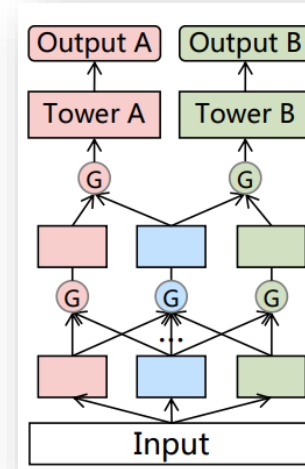
Reinforcement Learning

Conversational Recommenders, critiquing

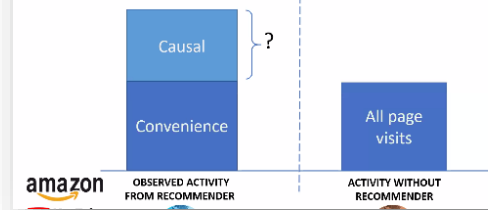
Critiquing-based Recommender Systems



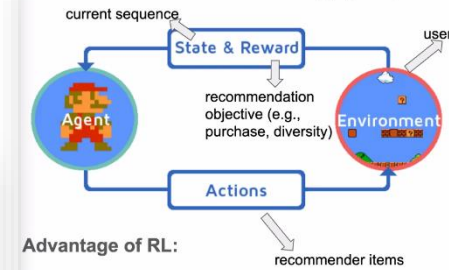
Li Chen and Pearl Pu, Critiquing-based Recommenders: Survey and Emerging Trends, User Modeling and User-Adapted



Observed activity is almost surely an overestimate of the causal effect



Reinforcement Learning (RL)



The RL agent is trained to take actions given the state of the environment with the objective of getting the maximum long-term rewards.

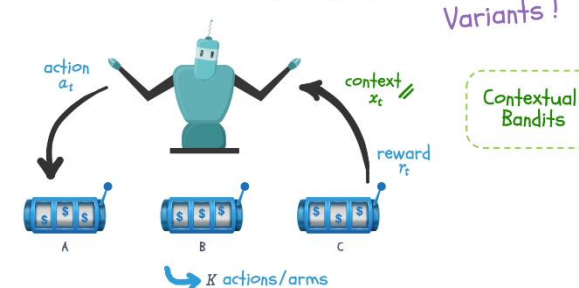
Advantage of RL:

- Flexible reward setting
- Long-term optimization

Casting se recommend problem is

Multi-Armed Bandit Problem

Variants!





Artificial Intelligence Research Institute

airi.net



[airi_research_institute](https://t.me/airi_research_institute)



[AIRI Institute](https://vk.com/AIRI_Institute)



[AIRI Institute](https://www.youtube.com/AIRI_Institute)



[AIRI_inst](https://twitter.com/AIRI_inst)



[artificial-intelligence-research-institute](https://www.linkedin.com/company/artificial-intelligence-research-institute)