Part II: Matching Users and Items with Deep NN

RecSys - Past, Present, and Future (Challenges) Course

Markus Reiter-Haas, Elisabeth Lex

July 2026

Course Website: *https://aisocietylab.github.io/recsys-ppf-course/*

Lecture Structure

- 5min Recap
- 15min Theoretical Introduction
- 45min Hands-on Coding
- 15min RecSys in Practice
- 10min Discussion/Questions

Recap: Introduction

- Ubiquity of RecSys
- RecSys: match users and items (often via latent factors)
- Hybridization of Approaches

Recap: Introduction

- Ubiquity of RecSys
- RecSys: match users and items (often via latent factors)
- Hybridization of Approaches

Ice Breaker Question: How are Recommender Systems different from Search Engines?

Recap: Matrix Completion

User-Item Matrix

Rating matrix (explicit):

$$\begin{bmatrix} 1 & ? & & 4 \\ & 2 & 3 & ? \\ & 4.5 & 3.5 & 5 \end{bmatrix} \quad (1)$$

Interaction matrix (implicit):

$$\begin{bmatrix} 1 & 0 & ? & \\ & ? & 0 & 0 \\ 1 & 1 & 1 & \end{bmatrix} \quad (2)$$

Challenge: Sparsity of $> 99\%$

Recap: Matrix Completion

User-Item Matrix

Rating matrix (explicit):

$$\begin{bmatrix} 1 & ? & & 4 \\ & 2 & 3 & ? \\ & 4.5 & 3.5 & 5 \end{bmatrix} \quad (1)$$

Interaction matrix (implicit):

$$\begin{bmatrix} 1 & 0 & ? & \\ & ? & 0 & 0 \\ 1 & 1 & 1 & \end{bmatrix} \quad (2)$$

Challenge: Sparsity of $> 99\%$

Question 1: strategy to predict the target (or missing) values?

Question 2: factors that influence prediction quality?

The Transition to Deep Learning

It's Matrices all the way down.

Learning Goals

By the end of the session, you should be able to:

- Describe how classical recommendation principles apply to deep learning recommender systems
- Train your own recommender system from scratch
- List various established recommendation training techniques
- Explain how real-world recommender systems are implemented

MF via Machine Learning

Idea: decompose into user matrix P ($|U|x|F|$) and item matrix Q ($|I|x|F|$) via latent features F

Restore rating matrix R ($|U|x|I|$) with $P \times Q^T$

Compression: $\#param = |F| * (|U| + |I|)$ rather than $\#param = |U| * |I|$

MF via SGD: FunkSVD popularized by the Netflix Prize

Predict ratings:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

MF via SGD: FunkSVD popularized by the Netflix Prize

Predict ratings:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

Minimize MSE (with regularization λ):

$$\sum_{r_{ui} \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2)$$

MF via SGD: FunkSVD popularized by the Netflix Prize

Predict ratings:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

Minimize MSE (with regularization λ):

$$\sum_{r_{ui} \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2)$$

Optimization via SDG (with learning rate γ)

$$b_u \leftarrow b_u + \gamma((r_{ui} - \hat{r}_{ui}) - \lambda b_u)$$

$$b_i \leftarrow b_i + \gamma((r_{ui} - \hat{r}_{ui}) - \lambda b_i)$$

$$p_u \leftarrow p_u + \gamma((r_{ui} - \hat{r}_{ui}) \cdot q_i - \lambda p_u)$$

$$q_i \leftarrow q_i + \gamma((r_{ui} - \hat{r}_{ui}) \cdot p_u - \lambda q_i)$$

Modifications

Only Baseline terms (μ, b_u,b_i) to indicate average offsets to the mean:

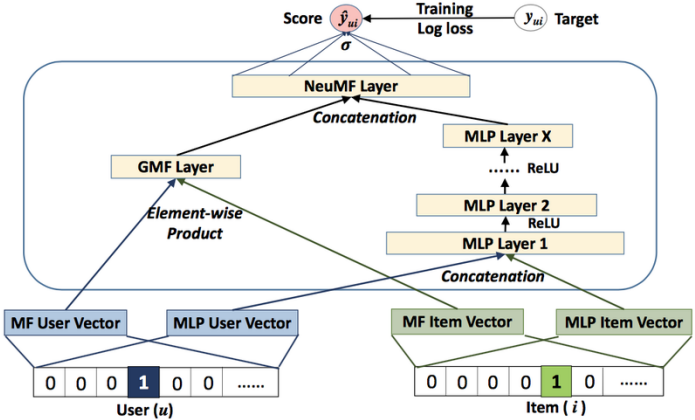
$$\hat{r}_{ui} = \mu + b_u + b_i$$

Unbiased version omits baseline terms (incl. mean μ) leads to probabilistic MF:

$$\hat{r}_{ui} = q_i^T p_u$$

Alternatively NMF, keep the factors positive (adjusts the SGD update rule)

Neural Collaborative Filtering



- Strong ID-based performance
- Similarity to Wide&Deep [4]

Figure 1: NCF combines shallow neural MF and deep MLP [8].

Early Fusion vs. Late Fusion

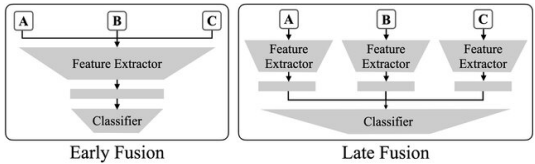
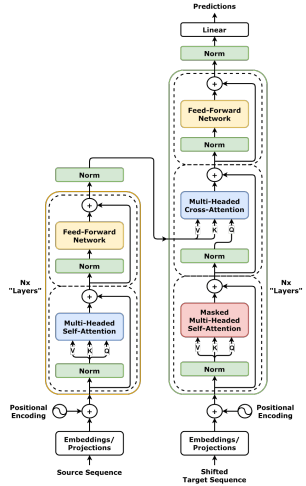


Figure 2: Comparison of fusion strategies.

Relevance for multimodality, e.g., with text data as side information.

Transformers



- Encoder followed by Decoder
- Multiheaded Attention (self, cross)
- Embed features

- Model for sequential or session-based recommendations
- Item IDs as tokens
- Transformers4Rec [16]

Figure 2: Transformer Architecture [20]

From NLP to IR to RecSys

1. NLP: doc(s) -> prediction (class, score, generative)
2. IR: q, docs -> prediction (docs, class, score, generative)
3. RecSys: u, docs -> prediction (docs, class, score, generative)

Two Tower Model

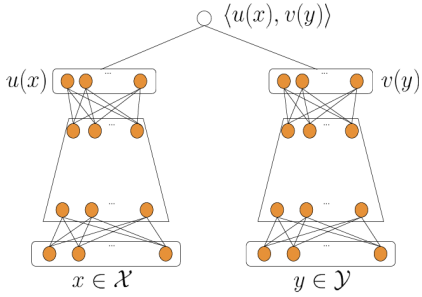


Figure 1: A two-tower DNN model for learning query and candidate representations.

Figure 4: Originally introduced by Google in 2019 for RecSys [22].

Two Tower Model - Common Setup

User Tower

- Encode User Features (e.g., demographic)
- Encode User History (often reusing Item Encoder)
- Combine (e.g., concat) to create User Embedding

Item Tower

- Encode Item Features (e.g., description)
- Leverage pretrained language models
- Combine (e.g., pooling) to create Item Embedding

Scoring, e.g., dot product or cosine similarity (depends on normalization)
See: *https://www.shaped.ai/blog/the-two-tower-model-for-recommendation-systems-a-deep-dive*

Practical Implementation

The Real Thing!

Recommender System Pipeline: Training

Training Pipeline:

1. Data Loading and Preparation
2. **Model Design and Implementation**
3. Training and Hyperparameter Optimization
4. Evaluation
5. (Deployment)

Practical Implementations

Four models considered (on MovieLens for simplicity):

- FunkSVD (plain Python)
- FunkSVD (PyTorch)
- NCF (prototypical NN model)
- Two Tower (naive) + FAISS Retrieval

https://www.kaggle.com/code/markusreiterhaas/essai26-recsys

Exercise: Improve an Algorithm

Challenge: Can you improve the performance with (small) alterations to the code?

Further Steps to Consider for Performance

- Strategies for cold-start users/items
- Integration of other features/knowledge sources
- Scalability, real-time demands, online learning

Beyond Pointwise Relevance Predictions

- **Pointwise Learning** ($\phi(u, i) \rightarrow r_{ui}$) to predict the relevance of user-item pairs, e.g., FunkSVD (ML)

Beyond Pointwise Relevance Predictions

- **Pointwise Learning** ($\phi(u, i) \rightarrow r_{ui}$) to predict the relevance of user-item pairs, e.g., FunkSVD (ML)
- **Negative Sampling** ($\phi(u, i) \rightarrow 1; \phi(u, j) \rightarrow 0$) to learn from uninteracted items, e.g., NCF (DL) [8]

Beyond Pointwise Relevance Predictions

- **Pointwise Learning** ($\phi(u, i) \rightarrow r_{ui}$) to predict the relevance of user-item pairs, e.g., FunkSVD (ML)
- **Negative Sampling** ($\phi(u, i) \rightarrow 1; \phi(u, j) \rightarrow 0$) to learn from uninteracted items, e.g., NCF (DL) [8]
- **Pairwise Learning** ($\phi(u, i) > \phi(u, j)$) to predict the relative ranking, e.g., BPR (loss for latent factors) [12]

Beyond Pointwise Relevance Predictions

- **Pointwise Learning** ($\phi(u, i) \rightarrow r_{ui}$) to predict the relevance of user-item pairs, e.g., FunkSVD (ML)
- **Negative Sampling** ($\phi(u, i) \rightarrow 1; \phi(u, j) \rightarrow 0$) to learn from uninteracted items, e.g., NCF (DL) [8]
- **Pairwise Learning** ($\phi(u, i) > \phi(u, j)$) to predict the relative ranking, e.g., BPR (loss for latent factors) [12]
- **Listwise Learning** ($\phi(u, i_1) > \phi(u, i_2) > \ldots > \phi(u, i_n)$) to predict the likelihood of entire lists (less common in practice) [15]

Beyond Pointwise Relevance Predictions

- **Pointwise Learning** ($\phi(u, i) \rightarrow r_{ui}$) to predict the relevance of user-item pairs, e.g., FunkSVD (ML)
- **Negative Sampling** ($\phi(u, i) \rightarrow 1; \phi(u, j) \rightarrow 0$) to learn from uninteracted items, e.g., NCF (DL) [8]
- **Pairwise Learning** ($\phi(u, i) > \phi(u, j)$) to predict the relative ranking, e.g., BPR (loss for latent factors) [12]
- **Listwise Learning** ($\phi(u, i_1) > \phi(u, i_2) > \ldots > \phi(u, i_n)$) to predict the likelihood of entire lists (less common in practice) [15]
- **Contrastive Learning** ($sim(u, i) \gg sim(u, j)$) to align the embedding space, e.g., SimGCL (self-supervised) [23]

Advanced Tasks and Models

- **Session-Based** models predict the next item(s) in (anonymous) sessions, e.g., GRU4Rec (RNN) [19]

Advanced Tasks and Models

- **Session-Based** models predict the next item(s) in (anonymous) sessions, e.g., GRU4Rec (RNN) [19]
- **Sequential** models predict the (long-term) evolution of user preferences, e.g., SASRec (Transformer) [10]

Advanced Tasks and Models

- **Session-Based** models predict the next item(s) in (anonymous) sessions, e.g., GRU4Rec (RNN) [19]
- **Sequential** models predict the (long-term) evolution of user preferences, e.g., SASRec (Transformer) [10]
- **Graph-based** models predict the higher-order structure of (bipartite) user-item graphs, e.g., LightGCN (GNN) [7]

Advanced Tasks and Models

- **Session-Based** models predict the next item(s) in (anonymous) sessions, e.g., GRU4Rec (RNN) [19]
- **Sequential** models predict the (long-term) evolution of user preferences, e.g., SASRec (Transformer) [10]
- **Graph-based** models predict the higher-order structure of (bipartite) user-item graphs, e.g., LightGCN (GNN) [7]
- **Generative** models predict the reconstruction of user profiles, e.g., MultiVAE (Autoencoder) [11]

Many more tasks/models, like context-aware using metadata to predict, e.g., DeepFM (FM)

Modern Recommender Systems (Research)

The Fragmented Landscape

Two Tower Model within Two Stage Retrieval

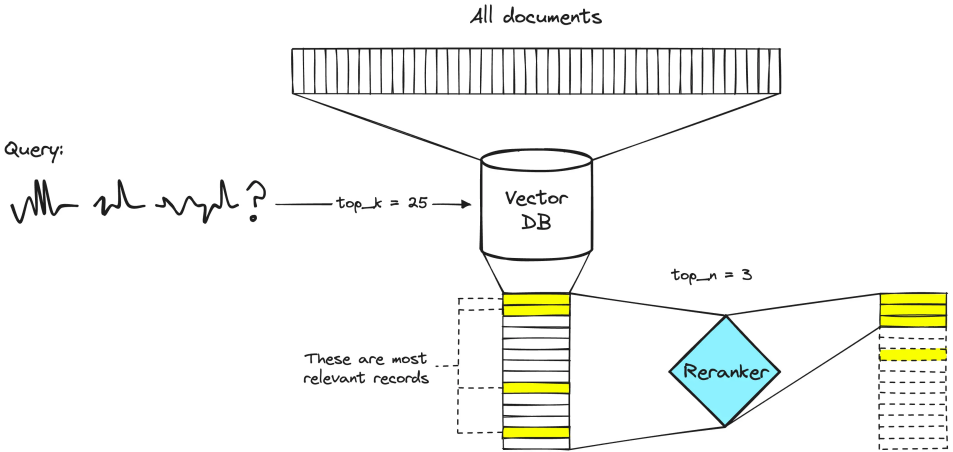


Figure 5: Retrieve candidates efficiently (Vector DB) then accurately rerank.

Recommender System Pipeline: Inference

Inference Pipeline:

1. Retrieval
2. Filtering
3. Scoring
4. Ordering

Still used today: *https://github.com/xai-org/x-algorithm*

Evaluation Protocols

- Test size: holdout ratio, leave-k-out
- Splitting Strategies: random, temporal, timestamp
- Model selection/hyperparameter optimization: cross-validation, grid search, significance tests
- Metrics:
  - Accuracy: RMSE (rating), AUC, F1@K (retrieval), MRR@K, nDGCG@k (rank)
  - Beyond-Accuracy: coverage, novelty, diversity, fairness
  - Platform: robustness, privacy
  - Online metrics: CTR, CR
  - User perception: enjoyment, serendipity, trust
- Your considerations?

Good overview at: *https://warprec.readthedocs.io/*

Recommender System Libraries (Overview)

- Elliot [1]
- Cornac [14]
- RecBole [25, 24]
- Recommenders (formerly Microsoft Recommenders) [6, 2]
- NVIDIA Merlin [21]
- LensKit [5]
- WarpRec [3]
- Surprise [9]
- DaiysRec [17, 18] (archived)

Note: often focused on evaluation and not deployment.

Closing and Reflection

- Deep learning to improve predictions
- Two tower model as practical solution
- Beyond matrix completion

Closing and Reflection

- Deep learning to improve predictions
- Two tower model as practical solution
- Beyond matrix completion

Questions?

Tomorrow: ethical challenges

References for Present i

[1] Vito Walter Anelli et al. **“Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation”.** In: *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 2021, pp. 2405–2414.

[2] Andreas Argyriou, Miguel González-Fierro, and Le Zhang. **“Microsoft recommenders: Best practices for production-ready recommendation systems”.** In: *Companion Proceedings of the Web Conference 2020*. 2020, pp. 50–51.

[3] Marco Avolio et al. **“WarpRec: Unifying Academic Rigor and Industrial Scale for Responsible, Reproducible, and Efficient Recommendation”.** In: *arXiv preprint arXiv:2602.17442* (2026).

References for Present ii

[4] Heng-Tze Cheng et al. **“Wide & deep learning for recommender systems”.** In: *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016, pp. 7–10.

[5] Michael D Ekstrand. **“Lenskit for python: Next-generation software for recommender systems experiments”.** In: *Proceedings of the 29th ACM international conference on information & knowledge management*. 2020, pp. 2999–3006.

[6] Scott Graham, Jun-Ki Min, and Tao Wu. **“Microsoft recommenders: tools to accelerate developing recommender systems”.** In: *Proceedings of the 13th ACM Conference on Recommender Systems*. 2019, pp. 542–543.

References for Present iii

[7] Xiangnan He et al. **“Lightgcn: Simplifying and powering graph convolution network for recommendation”.** In: *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020, pp. 639–648.

[8] Xiangnan He et al. **“Neural collaborative filtering”.** In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 173–182.

[9] Nicolas Hug. **“Surprise: A Python library for recommender systems”.** In: *Journal of Open Source Software* 5.52 (2020), p. 2174.

[10] Wang-Cheng Kang and Julian McAuley. **“Self-attentive sequential recommendation”.** In: *2018 IEEE international conference on data mining (ICDM)*. IEEE. 2018, pp. 197–206.

References for Present iv

[11] Dawen Liang et al. **"Variational autoencoders for collaborative filtering".** In: *Proceedings of the 2018 world wide web conference*. 2018, pp. 689–698.

[12] Steffen Rendle et al. **"BPR: Bayesian personalized ranking from implicit feedback".** In: *arXiv preprint arXiv:1205.2618* (2012).

[13] Francesco Ricci, Lior Rokach, and Bracha Shapira. ***Introduction to recommender systems handbook*.** Springer, 2010.

[14] Aghiles Salah, Quoc-Tuan Truong, and Hady W Lauw. **"Cornac: A comparative framework for multimodal recommender systems".** In: *Journal of Machine Learning Research* 21.95 (2020), pp. 1–5.

References for Present v

[15] Yue Shi, Martha Larson, and Alan Hanjalic. **"List-wise learning to rank with matrix factorization for collaborative filtering".** In: *Proceedings of the fourth ACM conference on Recommender systems*. 2010, pp. 269–272.

[16] Gabriel de Souza Pereira Moreira et al. **"Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation".** In: *Proceedings of the 15th ACM conference on recommender systems*. 2021, pp. 143–153.

[17] Zhu Sun et al. **"Are we evaluating rigorously? benchmarking recommendation for reproducible evaluation and fair comparison".** In: *Proceedings of the 14th ACM Conference on Recommender Systems*. 2020, pp. 23–32.

References for Present vi

[18] Zhu Sun et al. **"Daisyrec 2.0: Benchmarking recommendation for rigorous evaluation".** In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.7 (2022), pp. 8206–8226.

[19] Yong Kiam Tan, Xinxing Xu, and Yong Liu. **"Improved recurrent neural networks for session-based recommendations".** In: *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016, pp. 17–22.

[20] Ashish Vaswani et al. **"Attention is all you need".** In: *Advances in neural information processing systems* 30 (2017).

[21] Zehuan Wang et al. **"Merlin hugectr: Gpu-accelerated recommender system training and inference".** In: *Proceedings of the 16th ACM Conference on Recommender Systems*. 2022, pp. 534–537.

References for Present vii

[22] Xinyang Yi et al. **“Sampling-bias-corrected neural modeling for large corpus item recommendations”.** In: *Proceedings of the 13th ACM conference on recommender systems*. 2019, pp. 269–277.

[23] Junliang Yu et al. **“Are graph augmentations necessary? simple graph contrastive learning for recommendation”.** In: *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 2022, pp. 1294–1303.

[24] Wayne Xin Zhao et al. **“Recbole 2.0: Towards a more up-to-date recommendation library”.** In: *Proceedings of the 31st ACM international conference on information & knowledge management*. 2022, pp. 4722–4726.

References for Present viii

[25] Wayne Xin Zhao et al. **“Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms”.** In: *proceedings of the 30th acm international conference on information & knowledge management*. 2021, pp. 4653–4664.