Sequence-Aware Recommendation Benchmarking on MovieLens 1M

Akhil Arunachalam
(aarunachalam35@gatech.edu)

Akshay Daftardar
(adaftardar3@gatech.edu)

Harshada Kumbhare
(hkumbhare3@gatech.edu)

Rishi Sadhir
(rsadhir3@gatech.edu)

Abstract

This project benchmarks several generations of recommendation algorithms — classical matrix factorization (MF), deep matrix factorization (DeepMF), feature-aware deep models (FeatureAwareDeepMF), Improved Deep & Cross Network (DCN-V2), and sequence-aware transformers (SASRec) — on a sequence-aware next-item prediction task using the MovieLens 1M dataset [5]. The goal is to rigorously quantify the performance lift provided by architectural advancements and feature integration, specifically focusing on top-N ranking quality. We implement and study these models to refine recommendations through rating prediction, defining a next-item prediction task based on predicting the next movie a user rates highly (≥ 4 stars). Crucially, we establish a standardized evaluation protocol, including time-sensitive data splits [5] and a consistent negative sampling strategy, to ensure fair comparison between models [2]. This integrated approach not only enhances the accuracy of the recommendation, but also provides valuable insights into cross-feature interactions and sequential modeling. Our benchmark provides clear quantitative comparisons of architectural trade-offs and feature integration benefits, offering guidance for selecting appropriate models in modern recommendation systems.

1. Introduction/Background/Motivation

Recommender systems are essential for navigating catalogs of items and personalizing user experiences on platforms ranging from e-commerce to media streaming. A key challenge is predicting which item a user will interact with next, given their sequence of past interactions. This project tackles this sequence-aware next-item prediction problem. We created and evaluated five different models to generate insightful recommendations for movie watchers based on the MovieLens 1M dataset [5]. These models were as follows: Matrix Factorization (MF)[4], Deep Matrix Factorization (DeepMF)[4], Deep Matrix Factorization with Features (FeatureAwareDeepMF), Improved Deep & Cross Network (DCN-V2)[11], and a Transformer-Based Model called Self-Attentive Sequential Recommendation (SASRec)[7]. Our goal was to understand how powerful each model could be for recommending movies and to compare how different modeling choices, such as using user features, deeper network structures, cross layers, or sequential information, affected the quality of the recommendations. Our team tried to understand which model would be best for the given use case of movie recommendations.

Current state-of-the-art recommendation systems often rely on models like Deep and Cross Networks (e.g., DCN-V2 [11]), Two-Tower Embedding Models, or Large Language Models to make predictions. Historically, matrix factorization (MF) techniques have been a cornerstone, learning latent user and item factors but often ignoring interaction order. Deep learning models such as DeepFM [4] or DCN-V2 [11] excel at capturing feature interactions from tabular data, but may not fully leverage sequence dynamics. Sequence-aware models like SASRec [7], based on Transformers, directly model the order of the user history but can be computationally intensive. These advanced approaches are powerful but come with tradeoffs. Deep models can struggle with overfitting when there is not enough data or require significant feature engineering. Two-tower models are scalable but might lose important cross-feature interactions if not designed carefully. Large Language Models can personalize recommendations well but are often expensive to train and slow to run at scale. Fine-tuning can mitigate training time but scalability remains a concern. As a result, finding the right model often depends heavily on factors like training time, the specific problem (e.g., sequence awareness needed), the amount and type of available data (sparse interactions, rich features), and the system's speed requirements for delivering recommendations. A gap exists in rigorous benchmarking across these model generations under identical, time-aware data splits and evaluation protocols.

Recommendation Systems are a part of everyday life for many people, driving engagement on online marketplaces, media platforms, and more. Improving next-item prediction

directly translates to better user experience and business metrics. Better recommendations directly increase user satisfaction, ultimately improving business outcomes. Understanding the performance trade-offs between model complexity (MF vs DeepMF vs DCN-V2 vs SASRec), data requirements (impact of features), and prediction quality is crucial. By providing clear evidence on the performance gains from sequential modeling (SASRec) and feature integration (FA-DeepMF, DCN-V2) on a standard dataset using a fair evaluation protocol, this work directly informs architecture selection for practitioners building or improving movie recommendation systems, highlighting where complexity adds tangible value.

We utilize the widely-used **MovieLens 1M dataset** [5]. This dataset contains 1,000,209 explicit ratings (1-5 stars) given by 6,040 users to 3,900 movies.

Key Aspects:

- **Interaction Data**: Explicit star ratings provide a clear signal of user preference. We filtered these to ratings ≥ 4 to define positive interactions for our prediction task.
- **Temporal Information**: Crucially, the dataset includes timestamps for each rating, enabling the reconstruction of user interaction sequences, which is essential for benchmarking sequence-aware models like SASRec. EDA revealed trends in rating volume over time.
- **User/Item Features**: The dataset provides user demographic information (Gender, Age, Occupation) and movie attributes (Genres, Title containing year). This allows us to test the effectiveness of feature-aware models (like DCN-V2, FeatureAwareDeepMF) compared to ID-only models (MF, DeepMF, SASRec). EDA showed variations in rating behavior between different user groups (e.g., by age).
- **Sparsity**: While containing over 1 million ratings, the user-item interaction matrix is still relatively sparse compared to the number of possible interactions. EDA showed that all users had at least 20 ratings, mitigating extreme cold-start issues within this dataset. Movie popularity varied significantly.
- **Size**: With 6k users and 4k items, the dataset is large enough to demonstrate model differences but manageable for training multiple complex models for comparison.

Its size, richness in features, and essential temporal information make MovieLens 1M suitable for benchmarking different generations of recommendation models, assessing the value of incorporating side features, and evaluating sequence-aware approaches.

2. Approach

We implemented and evaluated five distinct model architectures [10] representing different recommendation paradigms on the sequence-aware next-item prediction task defined on MovieLens 1M. Our approach focuses on a rigorous comparison under identical data splits and evaluation protocols, rather than proposing a single novel architecture. Success is defined by demonstrating measurable improvements in ranking quality with increasing model sophistication and feature integration, and by comparing these against a sequence-aware model (SASRec).

2.1. Model Stack

We selected the following models, implemented using PyTorch [8]:

- **Matrix Factorization (MF)**[4]: Serves as our basline. A classical collaborative filtering baseline learning user and item embeddings. It models interaction score via dot product.
- **Deep Matrix Factorization (DeepMF)**[4]: Extends MF by replacing the dot product with a Multi-Layer Perceptron (MLP) to model user-item interactions, potentially capturing more complex non-linear relationships.
- **Feature-Aware DeepMF (FA-DeepMF)**[4]: Augments DeepMF by incorporating user features (Gender, Age, Occupation) and item features (Genres, Year) into the MLP interaction function. User Age and Movie Year were normalized to [0,1]. Categorical features were embedded.
- **Improved Deep & Cross Network (DCN-V2)**[11]: This model integrates an embedding layer for sparse features and directly uses dense features. It employs explicit cross layers to learn feature interactions, combined with a deep MLP path. We used 3 cross layers and a deep path with [128, 64, 32] hidden units. Features included UserID, Occupation, Age (bucketed), Zipcode, MovieID, Year (bucketed) as sparse features, and Gender (binary) and Genres (multi-hot binary) as dense features. The model was trained with dropout (0.4) for regularization.
- **Self-Attentive Sequential Recommendation (SASRec)**[7]: A Transformer-based model for sequential recommendation. It uses causal self-attention to weigh the importance of past items in a user's sequence to predict the next item. Includes user embeddings added to item+positional embeddings.

These models represent milestones in recommender system evolution. Our approach is a benchmarking study to clarify gains from architectural advancements.

2.2. Data Preparation Pipeline

The data preparation roughly follows the SASRec implementation:

1. **Loading & Cleaning**: Load 'ratings.dat', 'users.dat', 'movies.dat'. Filter ratings to valid users/movies and those ≥ 4 stars. Convert timestamps to datetime. Map UserIDs and MovieIDs to dense indices.

2. **Sequence Generation**: For each user, sort positive interactions chronologically ($s = [m_1, ..., m_T]$).

3. **Time-Aware Data Splitting**: Critically, split each user's sequence into training (first 80%), validation (next 10%), and test (final 10%) sets [7]. This was done to prevent data leakage. EDA showed that each user had rated at least 20 movies.

4. **Training Instance Construction**:

    - For sequence models (SASRec), generate (prefix, target) pairs: input $s[:t]$, target $m_t = s[t]$.
    - **Negative Sampling (Training)**: For each positive item during training, sample $K = 1$ negatives uniformly randomly from items the user hasn't interacted with.

5. **Sequence Padding/Masking (for SASRec)**: Right-pad input prefixes to $L = 50$ with index 0.

6. **Feature Engineering (for FA-DeepMF & DCN-V2)**[5]: Create feature tables as described in Sec 2.1 (user/movie features). Normalize continuous features (Age, Year for FA-DeepMF) or bucket them (Age, Year for DCN-V2). Create vocabularies and embeddings for categorical features (Gender, Occupation, Genres, IDs, Zipcode).

7. **DCN-V2 Data Adaptation**: Since DCN-V2 is inherently tabular, for each (user, target_item m_t) pair from the time-split data, we used the user's features and the target item's features as input. We treated the target item m_t as the positive example and sampled a negative item m_j for comparison during training/evaluation, effectively framing the task as predicting relevance for a given user-item pair based on features, rather than explicitly predicting the next item in sequence.

2.3. Relating to Deep Learning Concepts

- **Model Structure vs Problem**: SASRec's self-attention directly models sequence structure [7]. DeepMF/FA-DeepMF use MLPs for non-linear interactions but capture sequence implicitly in that the final learned user embedding becomes an aggregated representation or summary of that user's preferences, learned from the entire set of items they interacted with in the training portion of their history. By contrast, SASRec models the order [item1, item2, item3] $\rightarrow$ item4 using attention to weigh the importance of item1, item2, item3 based on their position and identity when predicting item4. Another area of comparison is that MF is linear while DCN-V2 uses cross layers and MLPs for feature interactions but doesn't model sequence order explicitly [11]. Our benchmark compares these different structural assumptions.

- **Learned Parameters**: All models learn user/item embeddings. Deep models (DeepMF, FA-DeepMF, DCN-V2, SASRec) learn weights/biases in MLPs, cross layers, or Transformer layers. FA-DeepMF DCN-V2 learn embeddings for side features.

- **Input/Output Processing**: As much as possible, inputs are kept the same across IDs, sequences, and features. Outputs are also the same set of ranking metrics such as NDCG. Pre-processing (filtering, ID mapping, sequencing, splitting, sampling, padding, feature extraction/normalization [5]) is consistent across algorithms [2].

- **Loss Function**: We employed the Bayesian Personalized Ranking (BPR) loss [9] as the optimization objective for all benchmarked models (MF, DeepMF, FA-DeepMF, DCN-V2, and SASRec). BPR is specifically tailored for implicit feedback scenarios where we only observe positive interactions (ratings ≥ 4) and the remaining interactions are unknown. Instead of predicting an absolute score or rating, BPR focuses directly on optimizing the *ranking* of items. It operates on triplets of $(user, positive, negative)$ and aims to maximize the score difference between the item the user interacted with and an item they did not. Our choice of BPR directly aligns with our evaluation goal: assessing how well models can rank the true next item highly within a list of candidates (measured by Hit@10, NDCG@10, etc.).

- **Regularization**: Techniques like Dropout (in MLP layers of FA-DeepMF, DCN-V2, SASRec) and Weight Decay (used in FA-DeepMF tuning) were employed to prevent overfitting, particularly important for complex models and feature-rich scenarios (see Sec 4.3).

- **Framework**: PyTorch was used for all implementations [8].

2.4. Anticipated and Encountered Problems

We anticipated challenges common in recommendation systems:

- Data sparsity and potential cold-start issues.
- Computational cost, especially for the Transformer-based SASRec model.
- Complexity of hyperparameter tuning for deep models (embedding dims, learning rates, layer sizes, regularization).
- Correctly implementing the time-aware data splitting and sequence handling logic [1].

Problems encountered during the project included:

- Initial hyperparameter choices often required iterative tuning on the validation set to achieve good performance and avoid overfitting. For example, MF tuning explored embedding dimensions and learning rates (see Sec 4.1).
- The data pipeline implementation, particularly time-aware splitting and padding/masking, was time consuming and needing careful verification.
- SASRec training was computationally intensive, necessitating adjustments to sequence length/model size and the use of GPU acceleration (A100 was significantly faster than MPS).
- DCN-V2 required careful feature engineering (deciding sparse vs dense, embedding sizes, bucketing) and tuning of cross/deep layer configurations and dropout to prevent overfitting.

The first attempt with default hyperparameters rarely yielded optimal results; systematic tuning was essential.

Code Usage: Implementations for MF, DeepMF, FA-DeepMF, and SASRec were based on standard architectures [4, 7]. DCN-V2 implementation adapted the CrossNetwork structure from deepctr_torch. Data preprocessing [1] and evaluation scripts [2] were developed following common practices. Our project code is available at [10].

3. Experiments and Results

Success was measured by the models' ability to rank the true next item (held-out positive item from the test set) highly against distractors, and by demonstrating performance differences aligned with model complexity and features.

3.1. Evaluation Protocol

We followed the protocol in 'evaluation.py'[2]:

1. **Candidate Set**: For each test user and their true next item i^+, create a set of 100 candidates: i^+ plus 99 random negative items never seen by the user.
2. **Scoring**: Score all 100 candidates using the trained model.
3. **Ranking**: Rank items by descending score.
4. **Metric Calculation**: Compute Hit@10, NDCG@10, MRR, MAP@10 based on the rank of i^+. Average metrics across users. Repeat 10 times with different negative samples and report mean ± std dev.

3.2. Evaluation Metrics

Standard top-N ranking metrics were used [2]:

- **Hit Rate @10 (Hit@10)**: Proportion of times i^+ is in the top 10.
- **Normalized Discounted Cumulative Gain @10 (NDCG@10)** [6]: Hit rate weighted by rank position (higher ranks better).
- **Mean Reciprocal Rank (MRR)**: Average of $1/\text{rank}(i^+)$.
- **Mean Average Precision @10 (MAP@10)**: Average precision up to rank 10.

3.3. Quantitative Results

Model performance on the test set (averaged over 10 trials) is in Table 1. Hyperparameters were selected via validation set performance (See Sec 4.1).

Table 1. Model Performance Comparison (Mean ± Std Dev over 10 trials)

Model	Hit@10	NDCG@10
MF	0.486 ± 0.003	0.252 ± 0.001
DeepMF	0.541 ± 0.003	0.301 ± 0.001
FA-DeepMF	0.629 ± 0.003	0.349 ± 0.001
DCN-V2	0.556 ± 0.003	0.299 ± 0.001
SASRec	**0.635 ± 0.002**	**0.369 ± 0.002**
Model	**MRR**	**MAP@10**
MF	0.205 ± 0.001	0.205 ± 0.001
DeepMF	0.251 ± 0.001	0.251 ± 0.001
FA-DeepMF	0.283 ± 0.001	0.283 ± 0.001
DCN-V2	0.242 ± 0.001	0.242 ± 0.001
SASRec	**0.306 ± 0.002**	**0.306 ± 0.002**

Analysis: The results demonstrate performance variations across architectures and the impact of features and sequence modeling.

- **MF vs DeepMF**: DeepMF significantly outperforms MF (e.g., NDCG@10: 0.301 vs 0.252), confirming the benefit of modeling non-linear user-item interactions via the MLP.

- **DeepMF vs FA-DeepMF**: Incorporating user/item features in FA-DeepMF yields another large improvement (NDCG@10: 0.349 vs 0.301), highlighting the value of context in this dataset.

- **DCN-V2**: DCN-V2 performed slightly worse than FA-DeepMF (NDCG@10: 0.299 vs 0.349), indicating that richer feature sets combined with deeper MLPs are more effective for this dataset than explicit feature crossing alone. DCN-V2 improves over MF but does not match the feature-enhanced models (FA-DeepMF, SASRec), highlighting that combining feature crossing with sequence modeling or richer features could further boost performance.

- **SASRec**: The sequence-aware SASRec model achieved the best performance across all metrics (e.g., NDCG@10: 0.369), even outperforming the feature-aware FA-DeepMF slightly. This strongly suggests that modeling the temporal order of interactions provides significant value for next-item prediction in this dataset, potentially capturing dynamic user interests more effectively than the feature-based models alone.

The small standard deviations indicate stable results. We succeeded in demonstrating quantitative lift from MF $\rightarrow$ DeepMF $\rightarrow$ FA-DeepMF, highlighting the benefits of deeper architectures and features. Furthermore, the strong performance of SASRec confirms the importance of sequence awareness. The relative performance of DCN-V2 suggests challenges in optimally applying it or its feature engineering in this specific ranking setup compared to the other approaches. These experiments successfully provide insights into the trade-offs between different modeling paradigms.

3.4. Qualitative Results

Examining specific recommendations provides qualitative insights. We analyzed item embedding similarity from the trained FA-DeepMF model by computing cosine similarity between movie embeddings. For example, when finding movies most similar to "Toy Story (1995)", FA-DeepMF ranked "Independence Day (ID4) (1996)" highly (Figure 1). While seemingly different (Animation vs Sci-Fi Action), this suggests the model learned connections based perhaps on shared broad genres (Adventure), release era proximity, or patterns in user co-viewing history (users enjoying family-friendly blockbusters might also watch contemporary action blockbusters). This illustrates the model capturing non-obvious relationships useful for a "Because you watched..." feature.

Interestingly, SASRec yielded different similarities. When performing a similar analysis (e.g., comparing the embedding of the last item in a sequence context), SASRec might rank dissimilar items higher if they frequently appear next in user sequences. For instance, a user watching "Toy Story" might next watch a classic like "Rebecca (1940)" if that pattern exists historically, a connection FA-DeepMF might miss. This highlights how sequence modeling can capture temporal dynamics beyond static feature similarity.

	MovieID	Title	similarity
0	0	Toy Story (1995)	1.000000
770	770	Independence Day (ID4) (1996)	0.603814
1085	1085	Top Gun (1986)	0.596194
1132	1132	Wrong Trousers, The (1993)	0.576420
1200	1200	Killer, The (Die xue shuang xiong) (1989)	0.561321
1246	1246	Unforgiven (1992)	0.545300
1372	1372	Jerry Maguire (1996)	0.522185
2072	2072	American Tail, An (1986)	0.515297
2260	2260	American History X (1998)	0.498058
2603	2603	Thirteenth Floor, The (1999)	0.489310
3867	3867	Runaway (1984)	0.476176

Figure 1. Example: FA-DeepMF Cosine Similarity Scores between Toy Story (1995) embedding and embeddings of other movies.

4. Hyperparameter Tuning & Ablation Studies

We performed hyperparameter tuning for all models and conducted ablation studies on FA-DeepMF.

4.1. Key Hyperparameters Tuned

Optimal hyperparameters found via validation set performance:

- **MF**: Emb Dim $k = 10$, LR=0.008, Batch=2048. (Tested $k \in [10, 20, 30]$, LR $\in [0.01, 0.005, 0.001]$ + finer search, Batch $\in [1024, 2048, 4096]$).

- **DeepMF**: $k = 25$, LR=0.0003, Batch=1024. (Tested $k \in [10, 25, 40]$, LR $\in [0.001, 0.0005, 0.0001]$, Batch $\in [512, 1024, 2048]$).

- **FA-DeepMF**: $k = 15$, LR=0.002, Batch=512, WeightDecay=1e-5, Dropout=0.3. (Tested $k \in [15, 32, 64]$, LR $\in [0.01, 0.005, 0.001]$, Batch $\in [256, 512, 1024]$).

- **DCN-V2**: LR=0.001, Batch=512, Hidden=[128, 64, 32], CrossLayers=3, Dropout=0.4. (Tested LR $\in [0.01, ..., 0.0005]$, Batch $\in [256, 512, 1024]$).

- **SASRec**: $k = 64$ (IDs), LR=0.001, Batch=1024, Heads=2, Layers=2. (Tested $k \in [64, 128]$, LR $\in [0.0001, 0.0005, 0.001]$, Heads $\in [2, 4]$, Layers $\in [2, 4]$).

Optimizer: Adam optimizer was used for all models.

4.2. Ratings Threshold Ablation Study

We retrained FA-DeepMF using only ratings ≥ 5 as positive interactions, compared to the baseline of ≥ 4.

Table 2. FA-DeepMF Performance with Ratings Threshold = 5 (Mean ± Std Dev over 10 trials)

Metric	Value	Std Dev
Hit@10	0.638	0.002
NDCG@10	0.370	0.001
MRR	0.303	0.001
MAP@10	0.303	0.001

Using a stricter definition of positive interaction (5 stars only) improved FA-DeepMF performance across all metrics compared to Table 1 (e.g., NDCG@10 increased from 0.349 to 0.370). This suggests the model benefits from a clearer positive signal, likely leading to more precise preference learning and better generalization.

4.3. Features Ablation Study & Regularization

Adding side features to DeepMF initially caused FA-DeepMF to overfit quickly, as shown by the diverging train/validation loss curves in Figure 2. The increased model complexity allowed memorization of the training data, potentially relying on noisy or irrelevant feature patterns.

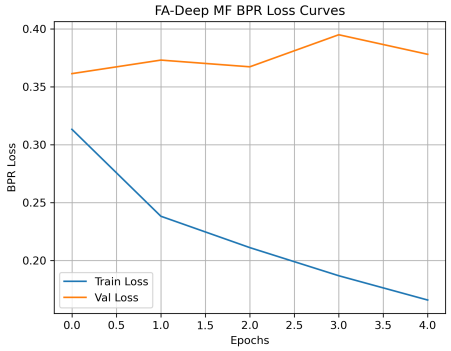

Figure 2. Initial FA-DeepMF Training Showing Overfitting (Validation loss increasing while Training loss decreases).

To mitigate this, we applied Dropout (rate 0.3 in MLP layers) and L2 regularization (Weight Decay = 1e-5). Dropout randomly zeroes activations, forcing the network to learn more robust representations not reliant on specific neurons/features. Weight decay penalizes large weights, promoting simpler models. These techniques successfully controlled overfitting, resulting in improved generalization and validation performance, as seen in the converged loss curves (Figure 3).

5. Conclusion

This project successfully benchmarked five recommendation models (MF, DeepMF, FA-DeepMF, DCN-V2, SASRec) on a sequence-aware next-item prediction task using

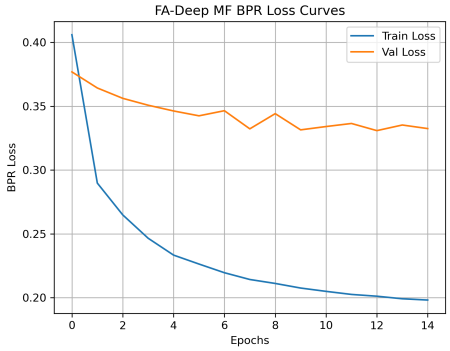

Figure 3. Tuned FA-DeepMF Training with Regularization (Validation loss closely tracks Training loss).

MovieLens 1M. Our results quantitatively demonstrate the benefits of non-linear modeling ($DeepMF > MF$) and incorporating side features ($FA-DeepMF > DeepMF$). Crucially, the Transformer-based SASRec achieved the highest performance, highlighting the significant value of explicitly modeling sequence dynamics for this task. The performance of DCN-V2 in our setup suggests potential challenges in feature engineering or adaptation to the ranking evaluation compared to alternatives. The rigorous time-aware evaluation protocol ensures fair comparison. Overall, this work provides empirical evidence guiding the selection between feature-rich tabular models and sequence-aware models based on performance trade-offs.

6. Potential Future Work

Beyond completing the SASRec evaluation and performing the planned ablation studies (impact of K, L, features, etc.), future work could involve:

- Exploring more sophisticated negative sampling strategies [3].
- Implementing and comparing against other sequence models (e.g., GRU4Rec, Caser) or different configurations of DCN-V2.
- Incorporating item similarity or content features more directly into models like SASRec.
- Analyzing performance stratified by user activity (cold-start vs power users).

References

[1] Akhil Arunachalam, Akshay Daftardar, Rishi Sadhir, and Harshada Kumbhare. Data Preprocessing Script. https://github.com/akshaydaf/recommender-system/blob/main/data_utils/preprocess.py, 2024. Part of ProjectGitRepo. 4

[2] Akhil Arunachalam, Akshay Daftardar, Rishi Sadhir, and Harshada Kumbhare. Evaluation Script. https://github.com/akshaydaf/recommender-system/blob/main/evaluation.py, 2024. Part of Project-GitRepo. 1, 3, 4

[3] Jiawei Chen, Xiang Wang, Jiancan Wu, Zhi-Hong Deng, and Xiangnan He. On the theories behind hard negative sampling for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 80–90, 2023. 6

[4] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 1725–1731, 2017. 1, 2, 4

[5] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19:1–19:19, 2015. 1, 2, 3

[6] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002. 4

[7] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206, 2018. 1, 2, 3, 4

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pages 8024–8035, 2019. 2, 3

[9] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2009*, pages 452–461, 2009. 3

[10] Rishi Sadhir, Akshay Daftardar, Akhil Arunachalam, and Harshada Kumbhare. MovieLens 1M Sequence-Aware Recommender Benchmark Project Code. https://github.com/akshaydaf/recommender-system, 2024. Accessed: 2025-04-27. 2, 4

[11] Ruoxi Wang, Rakesh Shivanna, Derek Z. Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed H. Chi. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining, WSDM 2021*, pages 936–944, 2021. 1, 2, 3