SMU SINGAPORE MANAGEMENT UNIVERSITY | School of Computing and Information Systems

REPRODUCIBILITY IN RECOMMENDER SYSTEMS

Presented by: Do Dinh Hieu

PREFERRED.AI

WHO HERE DOESN'T KNOW BPR?

…please don't raise your hand!

AND YOU ARE CONFIDENT YOU COULD IMPLEMENT BPR CORRECTLY?

"Claude, help me build a BPR pipeline."


You sure about that?

BPR implementations are NOT the same

SMU Singapore Management University | School of Computing and Information Systems

Table 2: BPR implementations. Source of the implementation: from the ACM Recommender Systems List of Frameworks (★), from the GitHub search (◇). Availability of features: available (✓), missing (✗). Original - MyMediaLite implementation.

	GitHub Stars[6]	Commits[7]	Features									
				Regularization				Optimizer		Negative Sampling	
			Item Biases	User λ_u	Positive λ_i	Negative λ_j	Shared λ	SGD	Others[8]	Uniform	Adaptive
Cython → Cornac★	820	116	✓	✗	✗	✗	✓	✓	✗	✓	✗
DaisyRec★	55	31	✗	✗	✗	✗	✓	✓	✓	✓	✗
Elliot★	265	1	✓	✓	✓	✓	✗	✓	✗	✓	✗
Recbole★	3,200	174	✗	✗	✗	✗	✓	✓	✓	✓	✗
ReChorus★	492	5	✗	✗	✗	✗	✓	✓	✓	✓	✗
RecPack★	6	70	✗	✓	✓	✗	✗	✗	✓	✓	✗
Implicit◇	3,400	27	✓	✗	✗	✗	✓	✓	✗	✗[9]	✗
LightFM◇	4,600	15	✓	✓	✓	✗	✗	✗	✓	✓	✗
C# → **Original**	498	0	✓	✓	✓	✓	✗	✓	✗	✓	✗
Ours			✓	✓	✓	✓	✓	✓	✓	✓	✓

[6]As of May 15, 2024. [7]From January 1, 2023, to May 15, 2024. [8]Adam, Adagrad, Momentum SGD, RMSProp. [9]Implicit utilizes popularity-based negative sampling.

PREFERRED.AI

Milogradskii, Aleksandr, et al. "Revisiting BPR: A Replicability Study of a Common Recommender System Baseline." Proceedings of the 18th ACM Conference on Recommender Systems. 2024.

WHICH IMPLEMENTATIONS WILL PERFORM THE BEST?

BPR implementations are NOT the same

SMU Singapore Management University | School of Computing and Information Systems

Table 2: BPR implementations. Source of the implementation: from the ACM Recommender Systems List of Frameworks (★), from the GitHub search (◇). Availability of features: available (✓), missing (✗). Original - MyMediaLite implementation.

	GitHub Stars[6]	Commits[7]	Features									
			Item Biases	Regularization				Optimizer		Negative Sampling	
				User λ_u	Positive λ_i	Negative λ_j	Shared λ	SGD	Others[8]	Uniform	Adaptive
Cornac★	820	116	✓	✗	✗	✗	✓	✓	✗	✓	✗
DaisyRec★	55	31	✗	✗	✗	✗	✓	✓	✓	✓	✗
Elliot★	265	1	✓	✓	✓	✓	✗	✓	✗	✓	✗
Recbole★	3,200	174	✗	✗	✗	✗	✓	✓	✓	✓	✗
ReChorus★	492	5	✗	✗	✗	✗	✓	✓	✓	✓	✗
RecPack★	6	70	✗	✓	✓	✗	✗	✗	✓	✓	✗
Implicit◇	3,400	27	✓	✗	✗	✗	✓	✓	✗	✗[9]	✗
LightFM◇	4,600	15	✓	✓	✓	✗	✗	✗	✓	✓	✗
Original	498	0	✓	✓	✓	✓	✗	✓	✗	✓	✗
Ours			✓	✓	✓	✓	✓	✓	✓	✓	✓

[6]As of May 15, 2024. [7]From January 1, 2023, to May 15, 2024. [8]Adam, Adagrad, Momentum SGD, RMSProp. [9]Implicit utilizes popularity-based negative sampling.

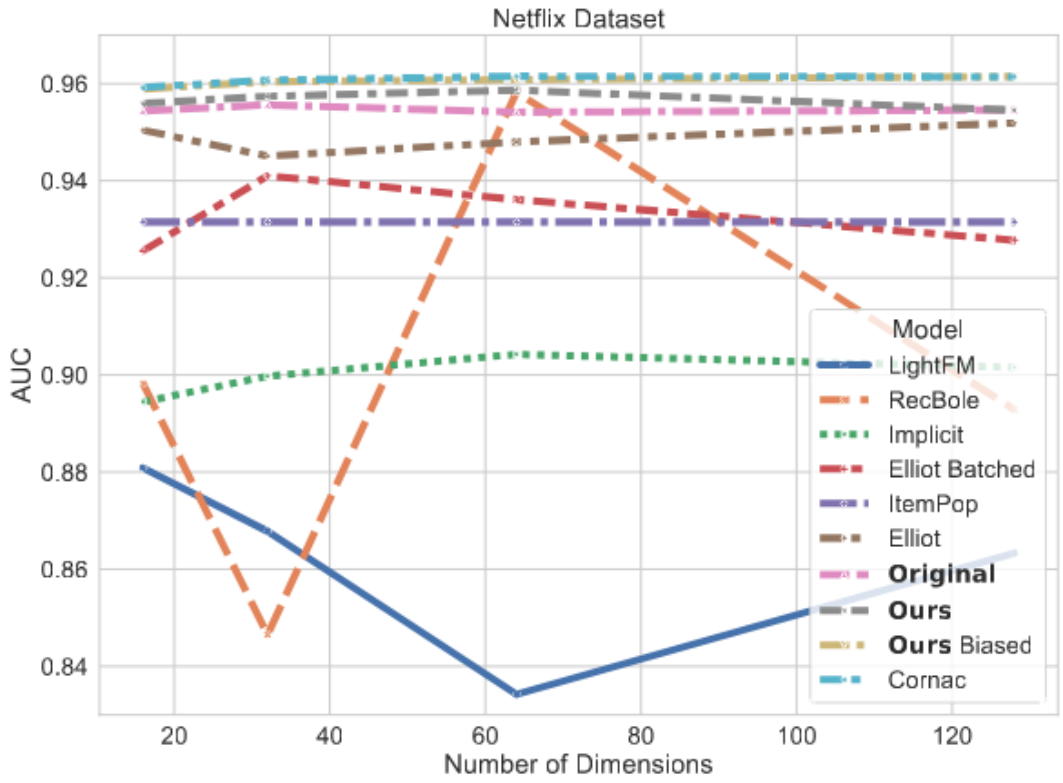


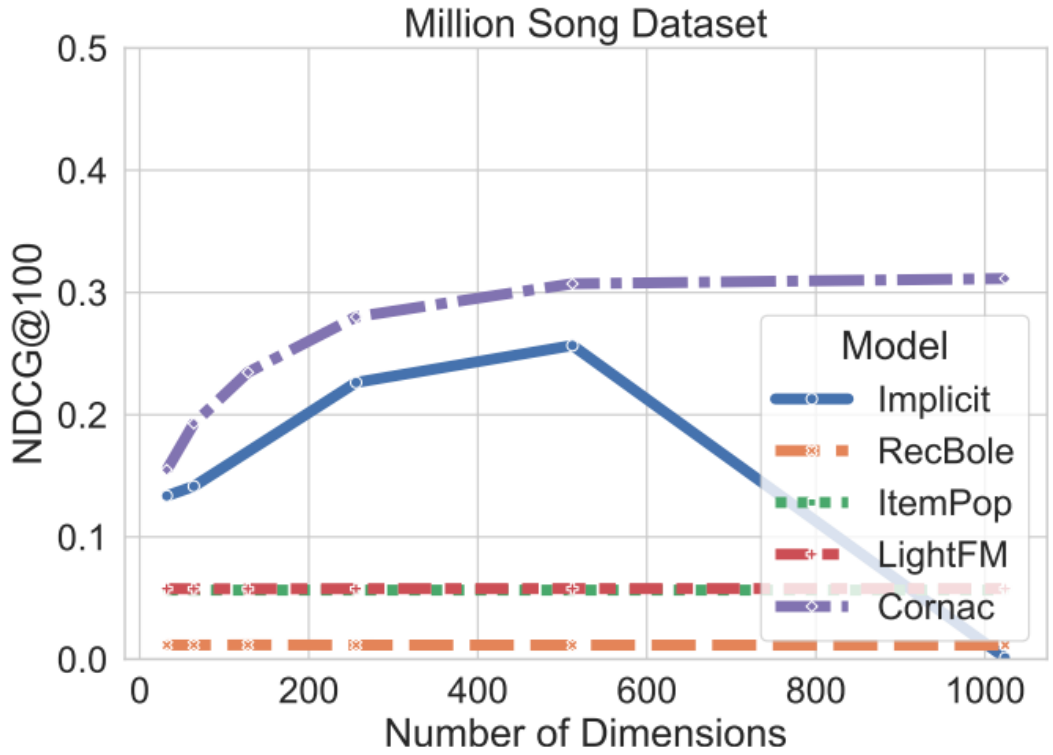
Figure 2: Area Under the ROC Curve (AUC) prediction quality for the Netflix dataset using various open-source BPR implementations. The ItemPop model is included to compare performance against a non-personalized baseline.

PREFERRED.AI

Milogradskii, Aleksandr, et al. "Revisiting BPR: A Replicability Study of a Common Recommender System Baseline." Proceedings of the 18th ACM Conference on Recommender Systems. 2024.

BPR implementations are NOT the same

(a) MovieLens-20M — NDCG@100



(b) MSD — NDCG@100

Figure 3: Performance in NDCG@100 relative to the number of embedding dimensions on the two datasets.

PREFERRED.AI

Milogradskii, Aleksandr, et al. "Revisiting BPR: A Replicability Study of a Common Recommender System Baseline." Proceedings of the 18th ACM Conference on Recommender Systems. 2024.

Why even BPR isn't safe?

SMU Singapore Management University | School of Computing and Information Systems

- Research is a **trial-and-error** loop:
  - Paper reports **one winning lap**
  - Publish **successes**, not the failed experiments to learn from
- Benchmark discrepancy:
  - Different split, preprocessing
- New work copies flawed evaluation setups from prior work
  - Propagating the same flaws

This is not uncommon

7 / 18

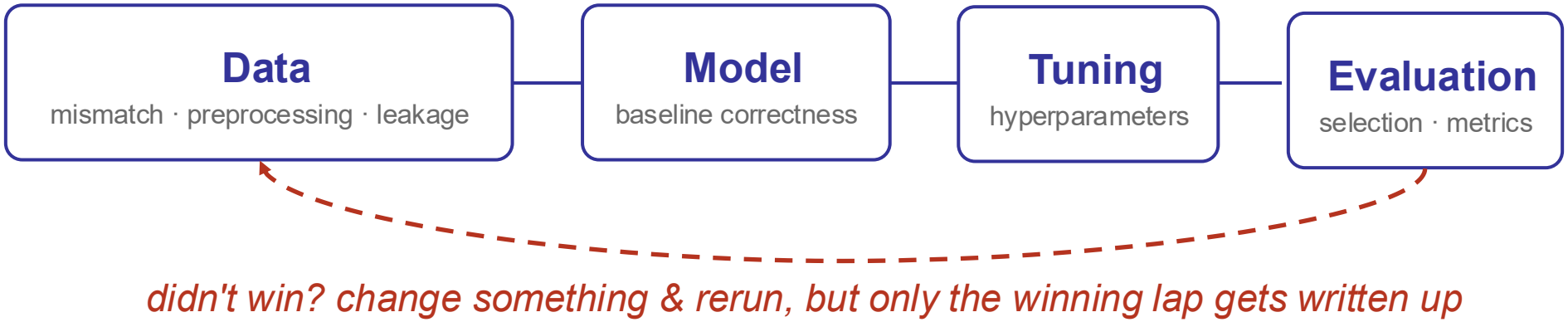
neural top-N methods from top venues were **reproducible with reasonable effort**.

6 / 7

of those were **beaten by simple, well-tuned baselines** (e.g., nearest neighbor).

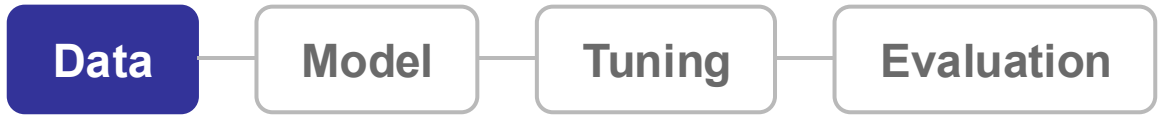
Ferrari Dacrema et al., "Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches." RecSys '19: Proceedings of the 13th ACM Conference on Recommender Systems 2019.
Ferrari Dacrema et al., " A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research." ACM Transactions on Information Systems (TOIS) 39.2 (2021)

Flaws can happen at every stage



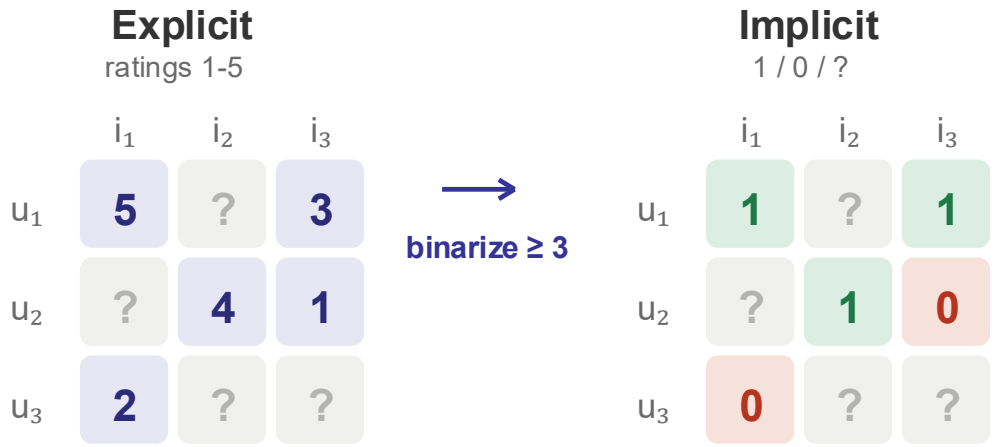
DATA

Dataset-task mismatch · Inconsistent preprocessing · Leakage



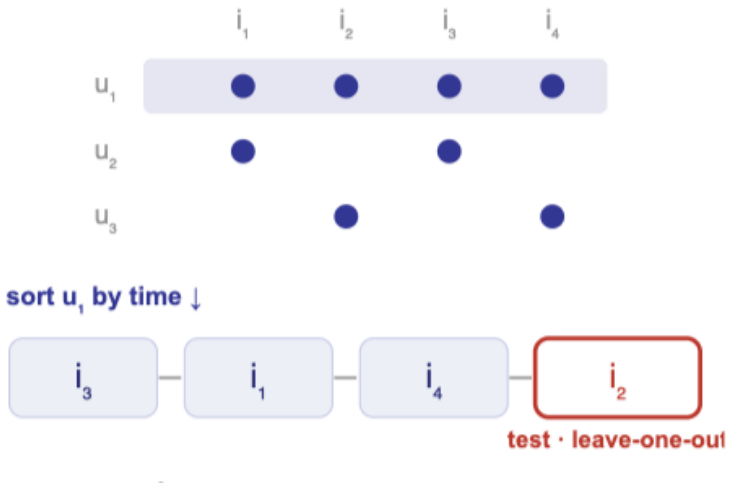
Data isn't built for the task

- Explicit data: user *explicitly* tells how much they rate/like the item (1-5)
- Implicit data: we only see the action happened (click, play, buy), binary (0/1)
- Models need *positive* and *true negative* items
  - Usually not available
  - Derived from rating dataset: e.g., Amazon rating (scale 1-5)
    - Positive: rating >= 3
    - Negative: rating < 3
  - How about unobserved items?

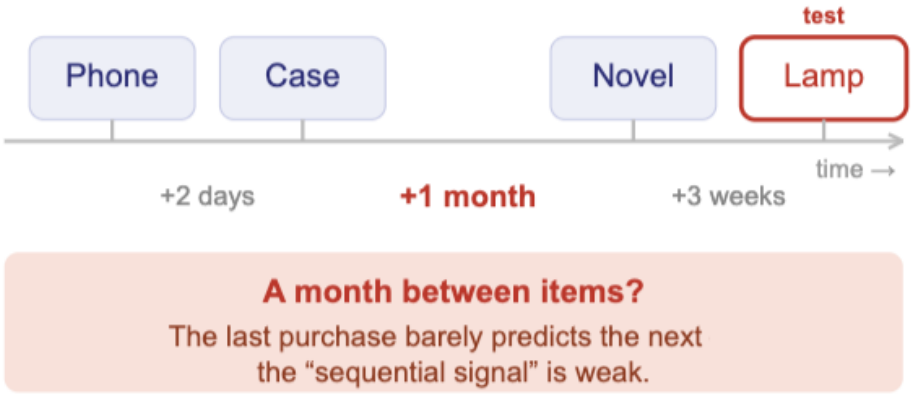


Sequential data needs meaningful order

- [1] shows that “Amazon dataset has weak sequential signals”
- E.g., RecSys’24 papers still uses rating datasets (Amazon, MovieLens)
  - CALRec: Contrastive Alignment of Generative LLMs for Sequential Recommendation
  - A Pre-trained Zero-shot Sequential Recommendation Framework via Popularity Dynamics
  - Scaling Law of Large Sequential Recommendation Models



Each user → one time-ordered sequence;
the last item is held out (leave-one-out).



An e-commerce *session,* clicks seconds apart,
would instead give a strong signal.

[1] Tang, Jiaxi, and Ke Wang. "Personalized top-n sequential recommendation via convolutional sequence embedding." Proceedings of the eleventh ACM international conference on web search and data mining. 2018.

Inconsistent Preprocessing

SMU Singapore Management University | School of Computing and Information Systems

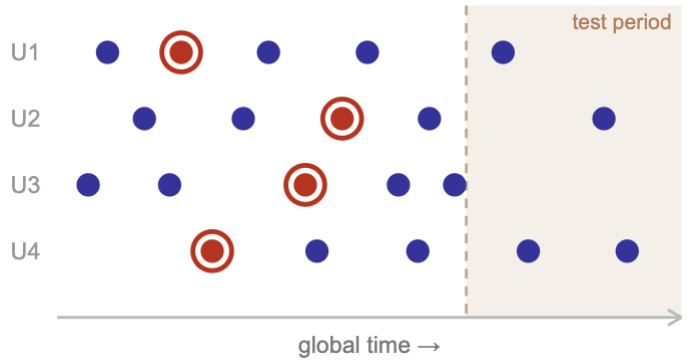
- Keep user/item with at least **5** ratings vs. minimum **10** ratings
- Keep duplicates vs. remove duplicates (e.g., same [a,b,c] sequence, different timestamp)
- Retain repetitions [a,b,a,c,a] in session data vs. remove [a,b,c]
- Data leakage (non-temporal split)

→ The same dataset (name) becomes many different datasets:

Data Leakage

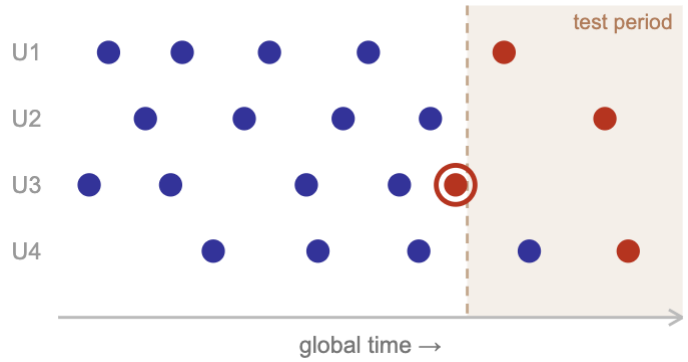
SMU Singapore Management University | School of Computing and Information Systems

Random split



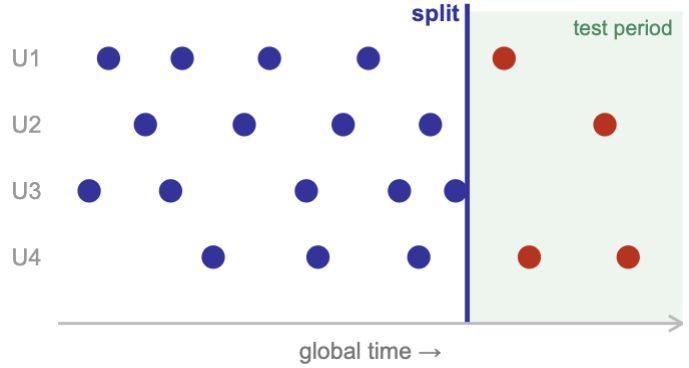
Test items land all over the past.

Leave-one-out



U3's *last* item is still in everyone's past.

Global temporal split



One cut: train on the past, test on the future.

training · held-out test · leak, tested before later training data · test period (future)

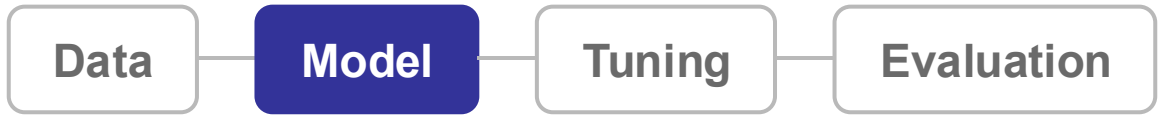
Use future training data can increase HR@20 by **+89.5%**

PREFERRED.AI

Ji, Yitong, et al. "A critical study on data leakage in recommender system offline evaluation." *ACM Transactions on Information Systems* 41.3 (2023): 1-27.

MODEL

Baseline correctness: when the comparison itself is broken



Case Study: GRU4Rec

- GRU4Rec: popular baseline for sequential recommendation (~5k citations)
  - Original implementation: Theano
  - There are the needs for other frameworks implementations (Pytorch, Tensorflow)

Most GitHub stars

Table 1: Availability of GRU4Rec's main features in third-party versions: available (✓), missing (✗), or partially present / flawed (*).

GRU4Rec feature		GRU4REC-pytorch	Torch-GRU4Rec	GRU4Rec_Tensorflow	KerasGRU4Rec	Recpack
Session parallel mini-batches		✓	✓	✓	✓	*
Negative sampling	Mini-batch	✓	✓	✓	✗	*
	Shared extra	✗	✓	✗	✗	*
Loss	Cross-entropy	*	✓	✓	✓	✓
	BPR-max	*	✓	✗	✗	✓
Embedding	No embedding	✓	✓	✗	✓	✗
	Separate	✓	✓	✓	✗	✓
	Shared	✗	✗	✗	✗	✗

Hidasi, Balázs, and Ádám Tibor Czapp. "The effect of third party implementations on reproducibility." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

Most GitHub stars implementation: the worst

SMU Singapore Management University | School of Computing and Information Systems

Table 3: Quantitative comparison of GRU4REC-pytorch to the official implementation using recall@20 and MRR@20. Relative difference compared to the official version is shown in parentheses.

Dataset	Official GRU4Rec				GRU4REC-pytorch					
	All features		Matching features		Out-of-the-box		Inference fix		Major fix	
	Cross-entropy loss									
	Recall	MRR	Recall	MRR	Recall	MRR	Recall	MRR	Recall	MRR
Yoochoose	0.6804	0.3002	0.4583 (-33%)	0.1523 (-49%)	0.1169 (-83%)	0.1006 (-66%)	0.1212 (-82%)	0.1037 (-65%)	0.4271 (-37%)	0.1227 (-59%)
Rees46	0.5291	0.2003	0.3059 (-42%)	0.0828 (-59%)	0.1028 (-81%)	0.0713 (-64%)	0.1093 (-79%)	0.0755 (-62%)	0.1994 (-62%)	0.0371 (-81%)
Coveo	0.2947	0.0960	0.2061 (-30%)	0.0591 (-38%)	0.0628 (-79%)	0.0386 (-60%)	0.0663 (-78%)	0.0397 (-59%)	0.1806 (-39%)	0.0497 (-48%)
RetailRocket	0.4953	0.2012	0.3378 (-32%)	0.1121 (-44%)	0.0616 (-88%)	0.0517 (-74%)	0.0642 (-87%)	0.0545 (-73%)	0.2812 (-43%)	0.0986 (-51%)
Diginetica	0.4874	0.1440	0.2973 (-39%)	0.0749 (-48%)	0.0457 (-91%)	0.0329 (-77%)	0.0503 (-90%)	0.0366 (-75%)	0.2862 (-41%)	0.0736 (-49%)
	BPR-max loss									
	Recall	MRR	Recall	MRR	Recall	MRR	Recall	MRR	Recall	MRR
Yoochoose	0.6799	0.2931	0.4655 (-32%)	0.1760 (-40%)	0.0087 (-99%)	0.0011 (-99%)	0.1066 (-84%)	0.0118 (-96%)	0.1923 (-72%)	0.0391 (-87%)
Rees46	0.5206	0.1913	0.2615 (-50%)	0.0691 (-64%)	0.1316 (-75%)	0.0225 (-88%)	0.1418 (-73%)	0.0232 (-88%)	0.1764 (-66%)	0.0355 (-81%)
Coveo	0.3123	0.0994	0.2216 (-29%)	0.0643 (-35%)	0.1141 (-63%)	0.0279 (-72%)	0.1510 (-52%)	0.0368 (-63%)	0.1582 (-49%)	0.0410 (-59%)
RetailRocket	0.5187	0.2131	0.3142 (-39%)	0.1067 (-50%)	0.0794 (-85%)	0.0227 (-89%)	0.2528 (-51%)	0.0848 (-60%)	0.2554 (-51%)	0.0916 (-57%)
Diginetica	0.4995	0.1500	0.3141 (-37%)	0.0851 (-43%)	0.0070 (-99%)	0.0015 (-99%)	0.2367 (-53%)	0.0608 (-59%)	0.2616 (-48%)	0.0692 (-54%)

PREFERRED.AI

Hidasi, Balázs, and Ádám Tibor Czapp. "The effect of third party implementations on reproducibility." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

It can be done (almost) right

SMU Singapore Management University | School of Computing and Information Systems

Table 4: Quantitative comparison of Torch-GRU4Rec to the official implementation using recall@20 and MRR@20. Relative difference compared to the official version is shown in parentheses.

Dataset	Official GRU4Rec				Torch-GRU4Rec	
	All features		Matching features		Out-of-the-box	
Cross-entropy loss						
	Recall	MRR	Recall	MRR	Recall	MRR
Yoochoose	0.6804	0.3002	0.6690 (-2%)	0.2882 (-4%)	0.6671 (-2%)	0.2847 (-5%)
Rees46	0.5291	0.2003	0.4716 (-11%)	0.1624 (-19%)	0.4688 (-11%)	0.1606 (-20%)
Coveo	0.2947	0.0960	0.2848 (-3%)	0.0928 (-3%)	0.2835 (-4%)	0.0897 (-7%)
RetailRocket	0.4953	0.2012	0.4249 (-14%)	0.1635 (-19%)	0.3834 (-23%)	0.1464 (-27%)
Diginetica	0.4874	0.1440	0.4255 (-13%)	0.1246 (-13%)	0.4245 (-13%)	0.1229 (-15%)
BPR-max loss						
	Recall	MRR	Recall	MRR	Recall	MRR
Yoochoose	0.6799	0.2931	0.6769 (-0%)	0.2919 (-0%)	0.6711 (-1%)	0.2907 (-1%)
Rees46	0.5206	0.1913	0.5112 (-2%)	0.1839 (-4%)	0.5081 (-2%)	0.1820 (-5%)
Coveo	0.3123	0.0994	0.2995 (-4%)	0.0962 (-3%)	0.2960 (-5%)	0.0944 (-5%)
RetailRocket	0.5187	0.2131	0.4639 (-11%)	0.1773 (-17%)	0.4201 (-19%)	0.1633 (-23%)
Diginetica	0.4995	0.1500	0.4597 (-8%)	0.1383 (-8%)	0.4550 (-9%)	0.1372 (-9%)

15 stars
As of today

PREFERRED.AI

Hidasi, Balázs, and Ádám Tibor Czapp. "The effect of third party implementations on reproducibility." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

TUNING

Hyperparameters: SOTA models could just be training artifacts

Everyone's A Winner!

- Ideally, baselines and proposed models get **equal tuning effort**
- In practice, tuning baseline to its best **works against** the paper publication
- Freedom to choose datasets and metrics

→ Can be easily manipulated

Shehzad, Faisal, and Dietmar Jannach. "Everyone's a Winner! On Hyperparameter Tuning of Recommendation Models" Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

MostPop can be a winner too!

SMU Singapore Management University | School of Computing and Information Systems

Table 1: Accuracy results (NDCG@10) for tuned and non-tuned models, sorted by NDCG in descending order.

Tuned models					
ML-1M		**AMZm**		**Epinions**	
Model	*nDCG@10*	*Model*	*nDCG@10*	*Model*	*nDCG@10*
Mult-DAE	0,300	NeuMF	0,056	Mult-VAE	0,149
Mult-VAE	0,294	Mult-VAE	0,054	Mult-DAE	0.146
GMF	0,280	GMF	0,051	GMF	0,128
NeuMF	0,277	Mult-DAE	0,048	NeuMF	0,118
ONCF	0,225	*MostPop*	*0,013*	ONCF	0,077
MostPop	*0,162*	ConvMF	0,011	*MostPop*	*0,045*
ConvMF	0,160	NGCF	0,008	ConvMF	0,043
NGCF	0,100	ONCF	0,009	NGCF	0,031
Non-tuned models	>		>		>
Mult-DAE	0,071	Mult-DAE	0,003	Mult-DAE	0,015
ONCF	0,037	Mult-VAE	0,002	ONCF	0,005
ConvMF	0,022	ConvMF	0,002	NGCF	0,003
NeuMF	0,021	GMF	0,0007	GMF	0,002
GMF	0,016	NGCF	0,0006	Mult-VAE	0,002
NGCF	0,013	ONCF	0,0004	NeuMF	0,0008
Mult-VAE	0,006	NeuMF	0,0004	ConvMF	0,0008

PREFERRED.AI

Shehzad, Faisal, and Dietmar Jannach. "Everyone's a winner! on hyperparameter tuning of recommendation models." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

Are we making real progress

0/ 21

shared *baseline code*, or the code that tuned their own model.

6/ 21

gave *no information* on baseline tuning; four more used defaults.

2/ 21

reported the *full optimal hyperparameters* for model and baselines.

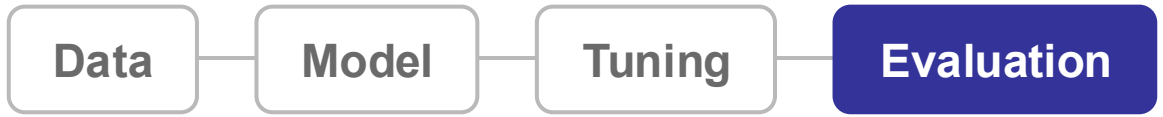
- Paper's scope:
  - Not to name or blame specific works
- Paper scanning:
  - 2022 proceedings of KDD, RecSys, SIGIR, TheWebConf, WSDM
  - Papers that report algorithmic improvements (top-N recommendation): 21 papers

Shehzad, Faisal, and Dietmar Jannach. "Everyone's a winner! on hyperparameter tuning of recommendation models." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

Shehzad, Faisal, et al. "'"We Share Our Code Online": Why This Is Not Enough to Ensure Reproducibility and Progress in Recommender Systems Research." *Proceedings of the Nineteenth ACM Conference on Recommender Systems*. 2025.

EVALUATION

Model selection · Sampled metrics · NCF vs. MF



Model Selection

The intended protocol:

Train set: fit the parameters.

Validation set: choose the model, early-stop.

Test set: report once, untouched until the end.

Potential flaws:

The test set isn't actually unseen.

Different papers use different test sets.

The "best" model is chosen on the test set.

Negative Sampling During Testing

SMU Singapore Management University | School of Computing and Information Systems

- Many only rank positive item against small **sampled negative candidate**
  - Reason cited: save test-time compute
  - For offline evaluation: training time >> inference time
  - Should only applied to multi-stage recommendation
- Negative sampling **during testing**:
  - High bias: instability, depend heavily on negative set [1]
  - More chances for proposed models to perform well
- Change the performance-based ordering of models [2,3]

[1] Krichene, Walid, and Steffen Rendle. "On sampled metrics for item recommendation." Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020.
[2] Cañamares, Rocío, and Pablo Castells. "On target item sampling in offline recommender system evaluation." Proceedings of the 14th ACM Conference on Recommender Systems. 2020.
[3] Dallmann, Alexander, Daniel Zoller, and Andreas Hotho. "A case study on sampling strategies for evaluating neural sequential item recommendation models." Proceedings of the 15th ACM Conference on Recommender Systems. 2021.

Case Study: Neural Collaborative Filtering

$$\phi^{\text{dot}}(\mathbf{p}, \mathbf{q}) = \langle \mathbf{p}, \mathbf{q} \rangle$$

$$\phi^{\text{MLP}}(\mathbf{p}, \mathbf{q}) = \mathbf{f}_{W_l, \mathbf{b}_l}(\ldots \mathbf{f}_{W_1, \mathbf{b}_1}([\mathbf{p}, \mathbf{q}]) \ldots)$$

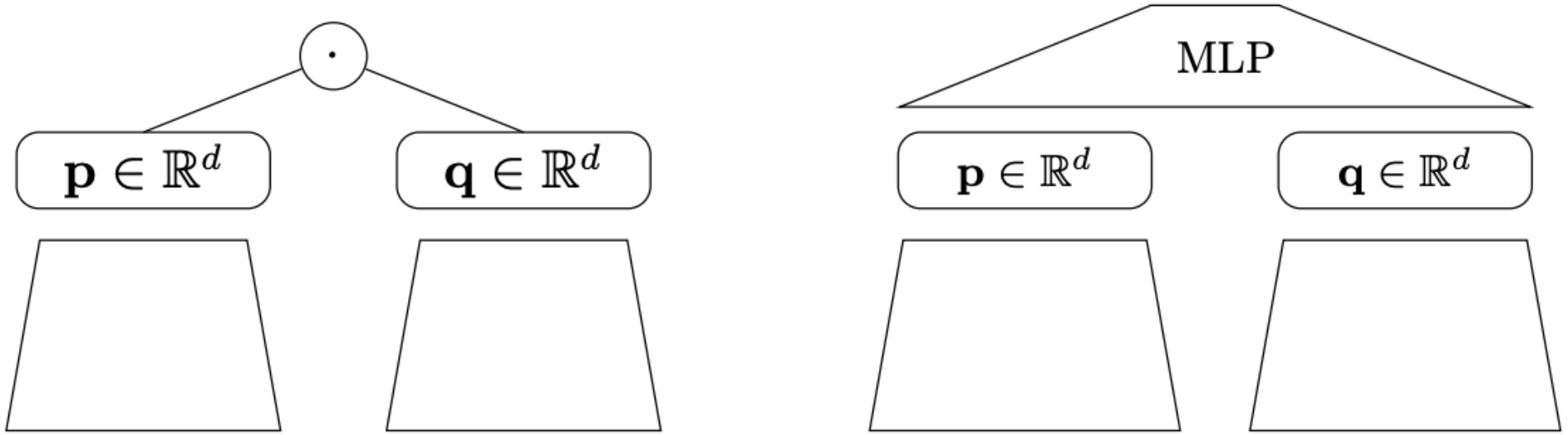


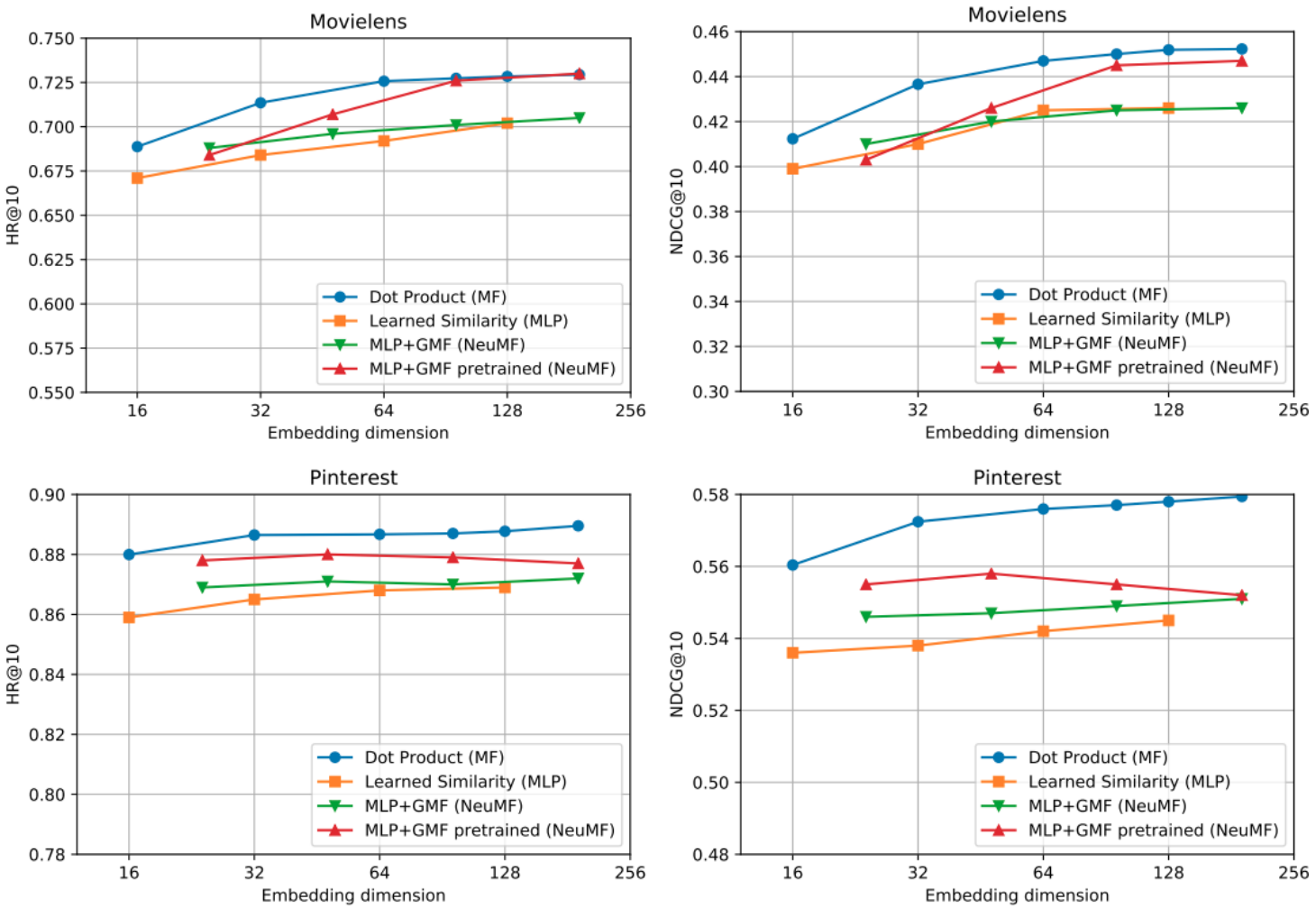
Figure 1: A model with dot product similarity (left) and MLP-based learned similarity (right).

Case Study: NCF vs. MF

SMU Singapore Management University | School of Computing and Information Systems

"The number of training epochs… was **erroneously optimized on the test data** *." [1]*

"The results for NeuMF and MLP were **cherry-picked,** *reported for the best iteration on the test set." [2]*



[1] Ferrari Dacrema, Maurizio, et al. "A troubling analysis of reproducibility and progress in recommender systems research." ACM Transactions on Information Systems (TOIS) 39.2 (2021): 1-49.
[2] Rendle, Steffen, et al. "Neural collaborative filtering vs. matrix factorization revisited." Proceedings of the 14th ACM Conference on Recommender Systems. 2020.

Same finding, new architecture, every year

SMU Singapore Management University | School of Computing and Information Systems

2019	Ferrari Dacrema: 6 of 7 reproduced **neural methods** beaten by tuned baselines (nearest neighbor). RecSys'19 / TOIS'21
2020	Rendle: a tuned dot product beats **Neural CF**. RecSys'20
2022	Petrov & Macdonald: BERT4Rec ports reach ~50% of reported numbers. RecSys'22
2023	Hidasi and Czapp: GRU4Rec ports lose up to 99%; Anelli**:** the **graph-CF** lead is a myth. RecSys'23
2024	Milogradskii: BPR shows up to 34× spread in performance across implementations. RecSys'24
2025	Shehzad & Jannach: well tuned GRU4Rec beats newer **graph-based** models. RecSys'25
2026	Benigni & Jannach: **diffusion** recommenders only 25% reproducible. TORS'26

Takeaways

- **Trust**, but **Verify**:
  - Published numbers in papers represent a “winning lap”, optimized for specific datasets.
  - Treat them as hypothesis, not guarantee for your use case.
- Performance = Algorithm + Implementation + Data:
  - **Always audit, debug, tune** library or code you clone.
  - Small bug at any stage can hurt performance significantly.
  - Choose models that fit your data.