Everyone's a Winner! On Hyperparameter Tuning of Recommendation Models

FAISAL SHEHZAD and DIETMAR JANNACH, University of Klagenfurt, Austria

The performance of a recommender system algorithm in terms of common offline accuracy measures often strongly depends on the chosen hyperparameters. Therefore, when comparing algorithms in offline experiments, we can obtain reliable insights regarding the effectiveness of a newly proposed algorithm only if we compare it to a number of state-of-the-art baselines that are carefully tuned for each of the considered datasets. While this fundamental principle of any area of applied machine learning is undisputed, we find that the tuning process for the baselines in the current literature is barely documented in much of today's published research. Ultimately, in case the baselines are actually not carefully tuned, progress may remain unclear. In this paper, we exemplify through a computational experiment involving seven recent deep learning models how every method in such an unsound comparison can be reported to be outperforming the state-of-the-art. Finally, we iterate appropriate research practices to avoid unreliable algorithm comparisons in the future.

CCS Concepts: • **Information systems → Recommender systems**.

Additional Key Words and Phrases: Recommender systems, Evaluation, Methodology

ACM Reference Format:
Faisal Shehzad and Dietmar Jannach. 2023. Everyone's a Winner! On Hyperparameter Tuning of Recommendation Models. In *Seventeenth ACM Conference on Recommender Systems (RecSys '23), September 18–22, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3604915.3609488

1 INTRODUCTION

Recommender systems are a highly visible success story of applied machine learning. Early reports of the value of such systems date back almost 25 years [27]. Today, most major online platforms use such systems to provide personalized item suggestions to their users, nowadays often based on deep learning, see, e.g., [7, 30]. Academic research in this area is flourishing as well, and many new machine learning models or network architectures for the recommendation task are published every year. However, there are indications that the progress that is reported to be achieved in academic papers is not as strong as one could expect, Today's publication culture more or less mandates that every new published model must significantly improve upon the "state-of-the-art" on various metrics and datasets. It was however observed in the related field of information retrieval many years ago that the reported improvements often do not add up [4]. Similar observations were made in the area of recommender systems [9, 23], as well as in other fields of applied machine learning, e.g., in time series forecasting [21]. In these and in several other works it turned out that the latest published models are in fact often *not* outperforming existing models and sometimes conceptually simple or longer-known methods can reach at least similar performance levels, at least in offline evaluations [19, 24].[1]

There are different factors that contribute to this issue, as discussed in [9, 18, 21]. Besides a certain researcher freedom when it comes to the selection of the baselines, the evaluation protocol and the metrics, one major problem seems to be that the hyperparameters of the selected baselines are often not properly tuned [20, 25], whereas significant effort

[1] An interesting performance comparison can be found in [16], where complex models were selected over simpler ones for production in the end.

Manuscript submitted to ACM

may go into tuning the newly proposed model. It is clear, however, that it is impossible to derive any insights from experiments in which not all compared models—both the proposed ones and the chosen baselines—are carefully tuned *for each dataset* considered in the evaluation. With the considerable computational complexity of some modern deep learning models, this systematic tuning process can lead to significant demands in terms of times and resources. It is therefore surprising that the documentation of this process—which may easily take weeks to complete—is often only very briefly covered in many papers or not mentioned at all. In some cases, only one set of optimal hyperparameters is reported, even though more than one dataset is used. Sometimes, the hyperparameters for the baselines are taken from the original paper, and no discussion is provided if the same datasets (after pre-processing) were actually used in the original paper. In yet other situations, probably the default parameters of a given public implementation of certain baselines have been used. Another observation here is that the code of the used baselines and the code that was used to automatically tune the hyperparameters for the experiments is not shared by the authors either.

With this short essay, we would like to showcase the dangers of what might be common practice in our field. Next, in Section 2, we report the results of the inspection of a set of recent conference papers published at highly relevant outlets for recommender systems in terms of *what is reported in the papers* regarding hyperparameter tuning of the baselines. We emphasize that we do not suggest that no proper hyperparameter tuning was actually done in these papers. Our observations only refer to the documentation of the process in the papers. In Section 3, we then report the outcome of an illustrative experiment, in which we tuned a set of recent models and then compared the results to those obtained when using the *random* parameters for the same models in a recent evaluation framework. Not too surprisingly, the results show that *every model can be a winner* when we compare its optimized version against non-optimized baselines. The paper ends with a discussion of implications and ways forward in Section 4.

2 A QUALITATIVE ANALYSIS OF THE STATE-OF-THE-PRACTICE

To obtain a picture of the state-of-the-practice in terms of what is reported today in research papers, we scanned the 2022 proceedings of five relevant ACM conference series, namely KDD, RecSys, SIGIR, TheWebConf, WSDM, for papers that report algorithmic improvements. Since our experiment in Section 3 focuses on the top-n recommendation task based mainly on user-item interaction matrices, we only considered such papers in our analysis. We identified 21 relevant papers, which we list in the online material. To avoid highlighting individual research works here we refrain from providing exact citations here for individual observations. We iterate that our goal is to provide evidence regarding *what is documented* in the context of hyperparameter tuning, and we are not challenging the reported results in any paper. We also acknowledge that our analysis is based on a certain selection of conferences series, and things may be different for other publication outlets. Still, we believe that our selection of papers is representative for today's research practices. Finally, we declare that there are similar patterns of shallow reporting of the hyperparameter tuning process in our own previous works as well. Our observations of what we find in the papers can be summarized as follows.

Tuning of the proposed model: On the more positive end, one of the examined works reports the searched ranges for "common" hyperparameters for such as learning rate, dropout ratio or the coefficient for L2 regularization. Other parameters are however taken from the original papers or using defaults from the provided codes. Furthermore, the optimal values are not reported in the end, and the pointer to the code leads to an empty GitHub repository. Another paper reports some of the hyperparameter ranges and some chosen values (also for the baselines), but does not report on how the parameters were found, e.g., by grid search or some other method. In this case, only one set of hyperparameter values is reported, even though evaluations were done on three datasets. Yet another paper only reports the fixed

learning rate that was used for all models and datasets. Furthermore, in this case, the embedding size was kept constant across all compared models "for fair comparison". In reality, however, embeddings sizes are hyperparameters to tune, and fixing them to one specific value (without much justification) may actually lead to an unfair comparison. In the end, we only identified two papers among the 21 ones we considered, which reported all optimal hyperparameter sets (proposed model and baseline) for all examined datasets and which documented the hyperparameter ranges and search procedure in more detail. The code for one of the papers was however not shared publicly.

Tuning of baseline models: The documentation of how the baselines were tuned and which parameters were finally chosen is even more sparse than for the proposed models. Four authors refer to default parameters of public code or the values that were used in the original papers (even though we notice from the previous paragraph that the quality of documentation can be improved even for the proposed models). In six papers, no information about the hyperparameters and the tuning process is provided at all. In seven cases, a subset of baseline hyperparameter values is reported, but without listing the ranges or how the values were determined. Again, in four cases only one fixed set of hyperparameter values is reported even though various datasets are used in the experiments. Finally, one paper reports that they used hyperparameters from the original papers if available, and applied grid search for the others. As mentioned, only two papers report detailed hyperparameter ranges and final values for all datasets.

Shared code: In 12 of the 21 papers, a link to a repository is provided. In two cases, these URLs were pointing to empty or non-existent places, leaving us with a ratio of 50% in terms of code sharing[2]. For the repositories actually containing code, we found that *none* of them contains the code of the used baselines. Also none of them contains code for the hyperparameter search, which would allow other researchers to reconstruct, e.g., the considered hyperparameter ranges. Note however that some papers mentioned that hyperparameter tuning was done in a manual process.

Discussion: Our analysis of a set of papers published at renowned conferences in 2022 clearly shows that the documentation of the hyperparameter tuning process—both for the proposed models and the baselines—is in most cases quite incomplete or even missing. Typically, authors spend about one paragraph in a section typically named "Implementation details", where this information is packed. In some works, the methodology also seems unclear, e.g., when authors report one single set of hyperparameter values for different datasets or when values from the original works are reused for very different datasets. Overall, this lack of documentation makes it unclear if it is mainly a issue of reporting (e.g., because of space limitations) or if we are frequently facing methodological issues of competing against non-tuned baselines, as indicated in the literature.

3 EXPERIMENTS

In this section, we showcase that adopting a research practice of not carefully tuning the hyperparameters of the baselines can indeed lead to arbitrary "winners" during the hunt for models that outperform the state-of-the-art.

3.1 Comparing tuned and non-tuned models

3.1.1 Methodology. We use the recent *Elliot* framework [2] as a basis for our evaluation. This Python-based framework implements a rich variety of recommendation models and it integrates components for automated hyperparameter

[2]This roughly corresponds to the rate reported in [9], who tried to reproduce recommender systems papers published at a similar set of conferences.

search and evaluation. Experiments, including hyperparameter ranges, can be defined through text-based configuration files, which allows for convenient reproducibility.[3].

Algorithms and Datasets. We focus our experiments on deep learning algorithms, which are the method of choice today. We initially considered all such methods implemented in *Elliot* at the time of the experiment. As some of the model implementations—including those proposed in [6],[12], and[28]—led to much lower performance levels as reported in the original papers (at an order of magnitude), we omitted them from this experiment and reported the issue to the framework authors. The following models were included in our experiments, all of which were proposed during the last few years, and which might be considered to represent the *state-of-the-art* in a research paper.

- Multinomial Likelihood Denoising Autoencoder (Mult-DAE) [17]
- Multinomial Likelihood Variational Autoencoder (Mult-VAE) [17]
- Convolutional Matrix Factorization (ConvMF) [15]
- Generalized Matrix Factorization (GMF) [14]
- Neural network based Collaborative Filtering (NeuMF) [14]
- Outer product-based Neural Collaborative Filtering (ONCF) [13], (named *ConvNeuMF* in *Elliot*)
- Neural Graph Collaborative Filtering (NGCF) [31]

In addition to these models, we include the non-personalized *MostPop* method in our experiment, which simply recommends the most popular items in the dataset (in terms of the interaction counts) to every user. We also used the implementation from the *Elliot* framework.

Researchers have a lot of freedom to select datasets and metrics for their experiments, which represents another reason why it is difficult to impossible to determine what actually represents the state-of-the-art. To avoid any bias in our setup, we strictly followed the experimental setup described in [3], which was also based on the *Elliot* framework. Regarding the dataset, we therefore used the exact same datasets and *p-core* pre-processing procedures. Specifically, the datasets include (i) *MovieLens-1M (ML-1m)*, a dataset with movie ratings, (ii) *Amazon Digital Music (AMZm)*, a dataset containing ratings for musical tracks, and (iii) *Epinions*, a dataset containing binary trust relationships between users of a social network. For the *ML-1M* and *AMZm* datasets, the rating data were transformed in a way that ratings with a value greater than 3 were encoded as positive signals.[4]

Hyperparameter Tuning Process. We used the *Tree Parzen Estimators* for hyperparameter tuning, a method that is embedded in *Elliot*. In terms of finding suitable *ranges* for the different hyperparameters, we adopted different strategies to inform our choices. First, we looked up the original papers to see if the authors reported ranges that they explored. If this was not the case, we selected ranges that are frequently observed in the literature, e.g., for the learning rate, a common range could be from 0.0001 to 0.01, and typical embedding dimensions in the literature are 64, 128 and 256. We set the number of iterations depending on the computational complexity of the tested algorithms. The number of iterations ranged from 10 to 50. The exact ranges and finally chosen values are documented in the online repository. Importantly, we note here that we selected ranges in a way that they are *good enough* for the purpose of our experiment. This means that we need to find a set of hyperparameter values for each model that is outperforming any other model when using random values. We therefore do not rule out that better hyperparameters can be found for each model and dataset; and that the ranking of the tuned algorithms may change with other values.

[3]We share all code and data for reproducibility online at https://github.com/Faisalse/RecSys2023_hyperparameter_tuning
[4]We note that the datasets are not too large, with the *AMZm* dataset being the largest one with about 1,5 million ratings before pre-processing. As observed in [3], systematically tuning the hyperparameters can be challenging already for these modest-sized datasets.

For the sake of our experiment, we first executed all models with the non-tuned (random) values for the hyperparameters. The random procedure consisted of arbitrarily picking values manually from the given ranges for each hyperparameter. We chose this procedure because we argue that there is a certain randomness in terms of how hyperparameters for the baselines are chosen in many papers. It can, for example, depend on what worked well for the datasets used in the original paper; or it can be simply based on the hard-coded default values that were left in the published source code by the authors. After having obtained the results when using non-tuned hyperparameters, we systematically tuned the hyperparameters for all models on all datasets to obtain the best accuracy values. We used the Normalized Discounted Cumulative Gain (NDCG) as the target metric during hyperparameter optimization.

3.1.2 Results. Table 1 shows the results for all models on the three datasets in terms of the NDCG@10. The ranking of the algorithms when using other metrics such as NDCG or MAP are as usual roughly aligned and can be found in the online repository. The upper part of the table shows the results after systematic hyperparameter tuning, and the lower parts shows the outcomes when running all models with the non-tuned hyperparameters. The numbers in the table correspond to the averages obtained through a five-fold cross-validation procedure.[5]

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

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

The most important observation is that for each dataset, even the worst-performing tuned model is better than the best model with non-tuned hyperparameters. Now this might not sound particularly surprising, given that it is well known that the performance of machine learning models can highly depend on the chosen hyperparameters. Looking at our results, we in fact find that the difference between tuned and non-tuned models can be of an order of magnitude and more.

[5]In terms of a sanity check, we compared the absolute numbers obtained in our experiment with those reported in [3], which are based on an identical experimental setting. We find that the best-performing methods are in the range of the performance of the Mult-VAE and NeuMF methods, which were considered in [3] as well.

However, the important point is that even the worst-performing models for each dataset, which have NDCG values that are actually *substantially lower* than the best performing tuned models, can be reported as being a *winner* in a comparison in which the hyperparameters of the competitors are not tuned. In other words, it can be sufficient to outperform an existing method that is actually quite weak to report an improvement over the state-of-the-art that includes as many as seven recent neural methods on three datasets. Considering however how little is documented about hyperparameter tuning of baselines in the current literature (see Section 2), there can be major concerns regarding the reliability of the reported rankings and improvements over the state-of-the-art. In fact, previous reproducibility analyses as reported, e.g., in [9], confirm that methodological issues of various kinds can hamper progress in recommender systems research.

Another striking observation here is that the popularity-based *MostPop* method is performing better than some of the tuned models. A similar phenomenon was also reported in [9]. The poor performance of some of the models—also compared to the *MostPop* method—may to some extent lie in the particular types of modest-sized datasets in our experiment. Or it may be the result of the somewhat restricted hyperparameter tuning process, which was only executed to the extent that was necessary for experiment. We iterate here that the ranking of the tuned models in Table 4 is not important, because we limited our search for hyperparameter ranges to typical values, and we limited the number of tuning iterations for the computationally complex models. Exploring alternative or more unusual ranges, as done recently in [24], may help to further improve the performance of the individual models.

3.2 Comparison with Other Baselines

Besides comparisons with baselines that are not well tuned, another potential methodological issue observed in [9] can lie in the *choice of the baselines* and the *propagation of weak baselines*. Remember that we limited our comparison to recent deep learning models, and reviewers might probably not complain, given the large set of recent baselines. However, there are several indications, also reported in [9] and other works, that simple methods can be competitive as well. The propagation of weak baseline may happen, if we only consider models from one family, e.g., neural models. In the analysis in [3], the linear and "shallow" EASE^R model [29] was the strongest performing method on the datasets used in our experiments. Also the traditional user-based nearest-neighbor method (userKNN), which was proposed in the context of the GroupLens system [26] from 1994 can lead to competitive results when tuned properly [3].

In fact, had we included EASE^R in our comparison, it would have been on top of the ranking list of the tuned models for all datasets, with NDCG@10 = 0.343 for *ML-1M*, NDCG@10 = 0.086 for *AMZm*, and NDCG@10 = 0.164 for the Epinions dataset. In that context it is also worth noting that EASE^R has only one relevant hyperparameter to tune (L2-norm). The performance of EASE^R was also quite good in case we selected this hyperparameter randomly, and a randomly tuned EASE^R model would be ranked about in the middle of the tuned models for all three datasets. It remains therefore important to consider simpler, e.g., linear, models, even though they might not be able to learn non-linear, higher-order dependencies in the data that neural models are often claimed to do in the literature.

4 IMPLICATIONS AND CONCLUSIONS

Our study showcases that almost arbitrary performance rankings for a given set of algorithms can be obtained depending on the level of fine-tuning of the hyperparameters. While this is certainly not a huge surprise, the fact that in many published papers insufficient information about the hyperparameter tuning process is provided may raise major concerns regarding the true progress that is achieved. A number of previous research efforts both in the area of recommender systems and related fields like information retrieval confirm this issue, e.g., [9, 18, 23]. The additional degrees of freedom

that researchers usually have, e.g., in terms of the selection of baselines, metrics, datasets, and pre-processing steps, can further aggravate the problem.

Various measures may help to better address these fundamental methodological problems. First, despite increased awareness in the community, *reproducibility* is still a major obstacle to achieving progress. We may observe that researchers share the code for their newly proposed models more often nowadays, and more attention is also paid to reproducibility in the peer-reviewing process, e.g., through explicit items on review forms. However, the reproducibility packages are often incomplete—in particular in terms of missing code for the baselines or the tuning process—which makes true reproducibility challenging. Various earlier works discuss this issue and propose guidelines and checklists for reproducible research both for general AI, for machine learning, and for recommender systems [8, 11, 22]. However, it seems that the research community is only making very slow progress towards a systematic and strict adoption of such checklists and guidelines. In the context of our present work, an important next step would be to make the provision of more detailed information about the hyperparameter tuning process and the sharing of the related software artifacts a mandatory requirement for paper submissions.

The increased use of *validated evaluation frameworks* such as *Elliot* is another measure that may help to address the problem. Various alternative frameworks exist, which often not only feature a rich number of baseline algorithms, but also include pre-implemented and validated procedures for systematic hyperparameter tuning and model evaluation. While custom evaluation procedures may be required for specific application scenarios, many of today's libraries can be used almost off-the-shelf for the most predominant evaluation scenarios, e.g., for traditional matrix-completion problems and top-n recommendation, as well as for alternative scenarios such as session-based recommendation. However, in the majority of works that we reviewed in the context of this work (see Section 2), researchers did not rely on existing frameworks, which also leads to the risk of unsound evaluation procedures. These problems may both stem from a lack of knowledge about proper evaluation methodology, or they may simply be the result of programming errors. The use of existing frameworks should therefore be strongly encouraged, both to avoid such mistakes and to increase reproducibility.

Generally, it seems that *increased awareness* of this fundamental issue is needed in the community, both on the side of researchers who develop and evaluate new models and on the side of reviewers and journal editors. One main instrument in this area may lie in *improved education* of scholars who are entering the field [5]. For example, today's textbooks on recommender systems, e.g., [1, 10], provide in-depth coverage of how to perform offline evaluation and which metrics may be used. Methodological questions of proper hyperparameter tuning for reliable experimental research results are however not discussed in much detail. Thus, it is important that in the future these topics are communicated more frequently through various educational channels, including books, lecture materials, or tutorials. Moreover, explicitly asking researchers to include more information about the hyperparameter tuning in their submitted works and considering this aspect in review forms may certainly increase the awareness of the problem.

Ultimately, one may speculate to what extent our currently acceptable practice of not reporting in depth about the tuning of the baselines and publication pressure—tuning many models on various datasets can be computationally highly demanding—contribute to the apparent problems in our field. Confirmation bias, where researchers have a predisposition to expect that their own model works better than previous ones, may also play a role. In general, we do not expect to see a radical change of the research practices in our field in the near future, also because many issues, e.g., related to reproducibility, are known for many years now. Through constant educational measures and increased awareness—also in the form of the showcase study in this present work–we however believe that we will observe more reliable results in recommender systems research in the future.

REFERENCES

[1] Charu C. Aggarwal. 2016. *Recommender Systems - The Textbook*. Springer.

[2] Vito Walter Anelli, Alejandro Bellogín, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco Maria Donini, and Tommaso Di Noia. 2021. Elliot: A Comprehensive and Rigorous Framework for Reproducible Recommender Systems Evaluation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2405–2414.

[3] Vito Walter Anelli, Alejandro Bellogin, Tommaso Di Noia, Dietmar Jannach, and Claudio Pomo. 2022. Top-N Recommendation Algorithms: A Quest for the State-of-the-Art. In *30th ACM Conference on User Modeling, Adaptation and Personalization (UMAP 2022)*.

[4] Timothy G. Armstrong, Alistair Moffat, William Webber, and Justin Zobel. 2009. Improvements That Don't Add Up: Ad-hoc Retrieval Results Since 1998. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*. 601–610.

[5] Christine Bauer, Maik Fröbe, Dietmar Jannach, Udo Kruschwitz, Paolo Rosso, Damiano Spina, and Nava Tintarev. 2023. Overcoming Methodological Challenges in Information Retrieval and Recommender Systems through Awareness and Education. In *Dagstuhl Seminar 23031: Frontiers of Information Access Experimentation for Research and Education*, Christine Bauer, Ben Carterette, Nicola Ferro, and Norbert Fuhr (Eds.). https://doi.org/10.48550/arXiv.2305.01509

[6] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jung-Tae Lee. 2018. CFGAN: A Generic Collaborative Filtering Framework based on Generative Adversarial Networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 137–146.

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the* 10^{th} *ACM Conference on Recommender Systems (RecSys 2016)*. 191–198.

[8] Paolo Cremonesi and Dietmar Jannach. 2021. Progress in Recommender Systems Research: Crisis? What crisis? *AI Magazine* 42, 3 (2021), 43–54.

[9] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. 2021. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. *ACM Transactions on Information Systems (TOIS)* 39 (2021). Issue 2.

[10] Bracha Shapira Francesco Ricci, Lior Rokach (Ed.). 2023. *Recommender Systems Handbook* (3 ed.). Springer.

[11] Odd Erik Gundersen, Yolanda Gil, and David W. Aha. 2018. On Reproducible AI: Towards Reproducible Research, Open Science, and Digital Scholarship in AI Publications. *AI Mag.* 39, 3 (2018), 56–68.

[12] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. 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*. 639–648.

[13] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-Based Neural Collaborative Filtering. 18 (2018), 2227–2233.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[15] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional Matrix Factorization for Document Context-aware Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 233–240.

[16] Pigi Kouki, Ilias Fountalis, Nikolaos Vasiloglou, Xiquan Cui, Edo Liberty, and Khalifeh Al Jadda. 2020. From the Lab to Production: A Case study of Session-based Recommendations in the Home-Improvement Domain. In *Proceedings of the* 14^{th} *ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys 2020)*. 140–149.

[17] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference*. 689–698.

[18] Jimmy Lin. 2019. The Neural Hype and Comparisons Against Weak Baselines. *ACM SIGIR Forum* 52, 2 (2019), 40–51.

[19] Malte Ludewig, Sara Latifi, Noemi Mauro, and Dietmar Jannach. 2021. Empirical Analysis of Session-Based Recommendation Algorithms. *User Modeling and User-Adapted Interaction* 31 (2021), 149–181.

[20] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are We Really Making Much Progress? Revisiting, Benchmarking and Refining Heterogeneous Graph Neural Networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Virtual Event, Singapore) *(KDD '21)*. 1150–1160.

[21] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2018. Statistical and Machine Learning Forecasting Methods: Concerns and Ways Forward. *PloS one* 13, 3 (2018).

[22] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. 2021. Improving Reproducibility in Machine Learning Research (a Report from the NeurIPS 2019 Reproducibility Program). *J. Mach. Learn. Res.* 22, 1 (2021).

[23] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys '20)*.

[24] Steffen Rendle, Walid Krichene, Li Zhang, and Yehuda Koren. 2022. Revisiting the Performance of IALS on Item Recommendation Benchmarks. In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys '22)*. 427–435.

[25] Steffen Rendle, Li Zhang, and Yehuda Koren. 2019. On the Difficulty of Evaluating Baselines: A Study on Recommender Systems. *CoRR* abs/1905.01395 (2019). http://arxiv.org/abs/1905.01395

[26] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*. 175–186.

[27] J. Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender Systems in E-Commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce (EC '99)*. 158–166.

[28] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders Meet Collaborative Filtering. In *Proceedings of the 24th International Conference on World Wide Web*. 111–112.

[29] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *The World Wide Web Conference, WWW 2019*. 3251–3257.

[30] Harald Steck, Linas Baltrunas, Ehtsham Elahi, Dawen Liang, Yves Raimond, and Justin Basilico. 2021. Deep Learning for Recommender Systems: A Netflix Case Study. *AI Magazine* 42, 3 (2021), 7–18.

[31] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 165–174.