Why Are Deep Learning Models Not Consistently Winning Recommender Systems Competitions Yet?

A Position Paper

Dietmar Jannach
University of Klagenfurt, Austria
dietmar.jannach@aau.at

Gabriel De Souza P. Moreira
NVIDIA
gspmoreira@gmail.com

Even Oldridge
NVIDIA
eoldridge@gmail.com

ABSTRACT

For the past few years most published research on recommendation algorithms has been based on deep learning (DL) methods. Following common research practices in our field, these works usually demonstrate that a new DL method is outperforming other models not based on deep learning in offline experiments. This almost consistent success of DL based models is however not observed in recommendation-related machine learning competitions like the challenges that are held with the yearly ACM RecSys conference. Instead the winning solutions mostly consist of substantial feature engineering efforts and the use of gradient boosting or ensemble techniques. In this paper we investigate possible reasons for this surprising phenomenon. We consider multiple possible factors such as the characteristics and complexity of the problem settings, datasets, and DL methods; the background of the competition participants; or the particularities of the evaluation approach.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Evaluation, Machine Learning Competitions

ACM RecSys Challenge Workshop 2020, Online

1 INTRODUCTION

Deep Learning (DL) has become the method of choice in many areas of applied machine learning, and recommender systems are no exception. The main machine learning problem in the area of recommender systems is to predict the relevance of items for individual users, usually with the goal of creating ranked recommendation lists. Many new algorithms or algorithm variants are proposed every year, and in recent years the large majority of these algorithms rely on deep neural networks.

The common research practice in algorithms research is to compare algorithms with the help of offline experiments on historical data. The goal in such an experimental setup is to demonstrate empirically that a newly proposed method is favorable over existing methods mostly with respect to one or more prediction or ranking accuracy metrics. Given the high interest of DL methods in recent years, most published research therefore demonstrates through offline experiments that a given DL method outperforms previous techniques, in particular ones not based on neural networks. Supporting this narrative, DL-based recommender systems have been successfully deployed in production by large companies, such as Alibaba [4], Baidu, [34], Google [5], Pinterest [33] and Facebook [9].

However, when we look at the outcomes of machine learning competitions on recommendation tasks, we can *not* observe such a consistent win of DL methods over non-DL techniques in recent years. When we, for example, look at the *challenges* that are held yearly at the ACM Conference on Recommender Systems (ACM RecSys), we instead observe that other techniques dominate the leaderboard. Specifically, methods often seem to be favorable in these competitions that rely on comparably old gradient boosting and ensemble techniques. Moreover, in most cases, one key to success lies in the effectiveness of the feature engineering process, which usually requires a good understanding of the domain.

To provide evidence for this discrepancy, we scanned the proceedings of the workshops that are associated with the challenges held with ACM RecSys from 2017 to 2019. Specifically, we considered the top solutions each year, for which papers were published at the workshop [1, 3, 11, 14, 15, 17, 18, 25–31], which we analyze and discuss next.

1.1 Analysis of DL Usage in Recent ACM RecSys Challenges

Among the top five[1] solutions from the ACM RecSys Challenge from 2017 to 2019, only one relied solely on a combination of DL methods: the second place [31] in the 2018 challenge. Their approach for the problem of automatic playlist continuation was a combination of an autoencoder to process tracks and artists lists with a character-level CNN to process music track titles. All other solutions are typically based on extensive feature engineering, gradient boosting, or the combination of various techniques, including matrix factorization, SVMs, logistic regression, content-based techniques or nearest neighbors. In some of the combined solutions, DL models are included. In these cases, authors usually report the results for individual models, and we often observe that the DL models do not reach the performance of gradient boosting (e.g., [14, 28, 30]), even when using the same set of features.

In the winning two-stage model of 2018, for example, a CNN component was used in the first stage, but it had the weakest performance of the first-stage models according to the authors [25]. Gradient boosting is combined in an ensemble with a DL technique for example in [28] and [14]. In the latter work, however, the DL component was only trained for one epoch to avoid overfitting, and at the end it contributed little to the overall accuracy.

[1]No workshop paper was published for the fifth place in 2018; the code is however provided online at https://github.com/zakharovas/RecSys2018.

For the second-place solution of the 2019 competition [28], the authors report that the best single model used XGBoost. They experimented with LSTM-based and Transformer-based neural architectures to model the sequence of items displayed to the user after a query, using the same set of features. Additionally, they enriched the feature set with the predicted output leaf of pre-trained XGBoost models (a technique known as *leaf encoding*). However, according to the authors "*[even] with all of the above transformations, we found it difficult to match GBM performance with deep learning models.*"

The winning team of the ACM RecSys 2020 challenge created its final predictions as an ensemble of three Gradient-Boosted Decision Trees (i.e., XGBoost), trained on more than one hundred engineered features. They explored some neural approaches to leverage textual content (BERT tokens), but they were not included in their final solution.

1.2 Observations from Other Recommender Systems and Data Science Competitions with Tabular Data

A similar situation can be found in recommender systems competitions hosted on popular platforms like Kaggle[2]. In the *Outbrain Click Prediction Competition*[3] (2017), for example, *Field-aware Factorization Machines (FFM)* were the workhorse technique from the top-3 winning solutions [13]. The same technique was also successful in previous CTR (Click-Through Rate) prediction competitions hosted by Criteo[4] (2014) and Avazu[5] (2014). Differently from the latter two competitions, the task in the Outbrain competition was ranking (instead of CTR prediction). The FFM technique however continued to perform the best three years later, also for this alternative prediction goal. Neural networks played a secondary role in those competitions, after FFM, logistic regression models with FTRL (Follow-The-Regularized Leader) [21] optimization, and XGBoost.

When looking at other data science competitions with tabular data in general, i.e., not only on recommender systems, our observations are similar. Considering Kaggle competitions from the last three years on tabular data, most of them had GBM as the core models. A few exceptions were the *Porto Seguro's Safe Driver Prediction*[6] and *Predicting Molecular Properties* [7] competitions, where the winning solutions relied on DL models. In the former one, the provided features names were anonymized, making it impossible to use domain knowledge for feature engineering. We might speculate that this was one factor that favored the use of DL models. In the latter competition, the winning solution was a carefully designed graph neural network (GNN) with self-attention. Here, the underlying graph structure of the data might have been advantageous for graph-based architectures.

Overall, however, given our general observations the question that arises is why we see such discrepancy, i.e., why DL models almost consistently win in academic comparisons and papers and in industry but not (yet) in the competitions. In this paper, we analyze the potential reasons for this phenomenon and we identified three main categories of potential differences and explanations: (1) the dataset and problem characteristics, (2) the researchers goals and motivation, and (3) the evaluation methodology. While we cannot give definite answers on what causes the discrepancy, this investigation aims to highlight those gaps and create a discussion on different topics, e.g., regarding the difficulties of applying cutting-edge research in competition scenarios or regarding potential methodological issues in academic environments.

[2]https://www.kaggle.com/competitions
[3]https://www.kaggle.com/c/outbrain-click-prediction
[4]https://www.kaggle.com/c/criteo-display-ad-challenge
[5]https://www.kaggle.com/c/avazu-ctr-prediction/
[6]https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/
[7]https://www.kaggle.com/c/champs-scalar-coupling

2 ARE THE PROBLEMS DIFFERENT?

One potential reason for the observed discrepancy could lie in the characteristics of the problems that are addressed in academic research and in the competitions.

Dataset-Related Aspects: The datasets provided by companies for competitions such as the ACM RecSys Challenge often comprise several million interactions and are meant to be representative of industry data. The XING dataset from the 2017 challenge for example contained over 320 million recorded interactions. Since the Netflix Prize in 2006 with its 100 million ratings, datasets of such a size are not uncommon in recommender systems research. However, in academia, recent DL methods are often evaluated on smaller datasets, which contain only 100.000 interactions or even less [8].

One popular assumption of DL methods is that they work particularly well when large amounts of data are available for training. If this holds true, these methods would profit from the large datasets used in the competitions. On the other hand, however, training deep neural networks can be computationally expensive, which is one of the reasons why even recent academic papers use comparably small datasets in their evaluations. In cases where the contestants in a competition do not dispose of large computing capacities (e.g. GPUs with large memory to hold giant embedding tables for high-cardinality categorical features) they might therefore resort to alternative approaches like gradient boosting.

A decisive difference between competitions and real-world deployments may be the volume of data that is available in real-world systems. Data in competitions is sampled from a short period of time and a subset of users, whereas deployed systems leverage iterative retraining of existing models to maintain rich interaction histories of users and items. The advantage of DL methods in deployed systems might therefore only unfold once more data is available.

Another aspect related to dataset characteristics is that some of the datasets that are used in academia—such as the traditional MovieLens100k collection of movie ratings—are less sparse than datasets from competitions and than real-world datasets in general. High data sparsity is known to potentially lead to overfitting, and this might specifically apply to certain DL architectures. An important aspect to consider here is that competition datasets are in fact often large in terms of the number of recorded interactions. However, sometimes these interactions were collected within a narrow time window, e.g., of a few weeks. As a result, user and item embeddings that are learned by many DL models are often based only on a very small set of interactions per user and item.

In terms of *additional* data, in the last three ACM RecSys challenges various types of meta-data were made available, i.e., not only the interactions between users and items were provided, but the datasets contained information about jobs, artists, and hotels, respectively. DL methods are often said to have the advantage that they are well suited for heterogeneous or multimodal data [23] and that they are able to detect and leverage complex interactions in such data, e.g., by using a shared representation. Again, the availability of meta-data should in principle be advantageous for DL methods, but we only observe few examples like [31], where a CNN was used to leverage meta-data information.

In some situations, the availability of more meta-data features can even mislead DL methods in certain ways. In recommender systems competitions, the most common tasks are binary classification of user interactions or ranking items based on their predicted relevance. Those tasks require negative samples (non-existing user-item interactions) for evaluation, and most algorithms also use those negative samples for training. The perfect scenario would be to have real negative samples of items, which were actually seen by the user and ignored. However, the negative samples in competition datasets are often artificially generated from some probability distribution over the items (e.g., based on items recency, popularity, co-occurrence, or content similarity), trying to emulate items that the user could have seen (and ignored) in their browsing sessions. If the distribution of the negative items is not very close to the positive ones, complex models can learn patterns that separate positive and negative samples from the available *leaking* features, and leverage this shortcut for accurate predictions. This risk increases when a richer set of features is made available.

Neural networks easily overfit when such *leaking* features are present, even with regularization techniques such as L2-regularization or dropout. Ensembles of Trees (e.g. GBDT, Random Forests), in contrast, combat overfitting with techniques like bagging (instance sampling with replacement), feature bagging (column sampling), and boosting (optimizing for correctly predicting the errors from the previous training step). In the end, these techniques may result in more generalizable models (i.e., low-variance errors).

Prediction-Related Aspects: In academia, the most common problems that are addressed are *rating prediction*, *binary classification* and *top-n recommendation* given a matrix of historical user-item interactions. In many research works, one corresponding assumption is that for each user a number of past interactions is known for the training phase. The prediction problems in the last three ACM RecSys challenges were however different. In 2017, the goal in the offline part of the challenge was to predict which users would be interested in a newly posted job offer, which corresponds to an item cold-start problem. Differently from typical academic research, a very specific evaluation metric was used that considered various types of user reactions on the recommendations. In 2018, the task was to create music playlist continuation given the first few tracks. In 2019, finally, the goal was to predict which of the hotel search results was clicked by a user in a given session. In particular in the latter two cases, the problem was not a traditional matrix completion setup, but rather a session-based and context-aware recommendation problem.

When using neural networks in item cold-start and user cold-start recommendation scenarios, usually the item and user embeddings will be as random as their initialization, with no predictive power. Leveraging user and item metadata (e.g., demographics, content features) and contextual information (e.g., recent interactions, time, location) is therefore key for making sense on whether a given user might be interested in certain item in a given context.

In recent years, a number of DL-based methods were proposed for session-based and sequential recommendation tasks [24]. However, in particular for session-based approaches, it often turned out that DL methods based on RNNs or Attention do not necessarily outperform conceptually more simple techniques based, e.g., on nearest neighbors [20][8]. Note that many of these session-based Deep Learning algorithms rely solely on collaborative information (i.e., user-item interactions), but do not take into account side information, which might be one factor that limits their effectiveness. For example, the experiments from [22] for the news domain report that simple session-based algorithms (e.g., based on kNN and association rules) were able to provide higher accuracy than RNN-based and GNN (Graph Neural Networks)-based models when solely considering user-item interactions. Only when side information was added to an RNN-based architecture (*CHAMELEON* [6]) that was designed to deal with the cold-start problem, the accuracy was 20% better than any other session-based algorithm.

3 ARE THE RESEARCHERS AND THEIR GOALS DIFFERENT?

Another potential reason why we do not see DL methods consistently win might be tied to the people who participate in the competitions. One could, e.g., assume that some participants do not have access to GPU-powered hardware, as mentioned above, which is why they resort to other, computationally less demanding techniques. An alternative hypothesis could be that challenge participants regularly competing in data science competitions. Given the popularity and past success of gradient boosting in such competitions, one could speculate that these participants either have a preference for more traditional models or that they are not proficient in the latest deep learning techniques.

When looking at the affiliations of the researchers who participate in the ACM RecSys challenges, it is however difficult to maintain such hypotheses. The top-performing contributions come from teams with heterogeneous backgrounds. There are teams from AI-focused companies, there are members of academic research groups working on recommender systems, and there are individuals, assumedly independent researchers or machine learning enthusiasts, for which not much background is available. At least for the participants from companies, we may assume that they dispose of sufficient computing resources. Moreover, when looking at individual researcher profiles, both from industry and academia, it also becomes clear that these participants are well aware of DL methods and in some cases, as described above, report their experiences of adding DL components to their solutions.

[8] As a side observation, note that nearest-neighbor techniques were quite often part of well-performing solutions previously, e.g., in [1, 18, 19, 27, 29].

Nonetheless, researchers and challenge contestants may have different objectives, preferences and work processes. Regular participants of data science competitions usually rely on generic tools that have passed the test-of-time and were part of many winning competitions solutions, such as Gradient-Boosting Trees. Those models are very lightweight in terms of data pre-processing (e.g., do not require feature scaling), do automatic feature selection, are robust against overfitting, and are interpretable, thus providing insights on the most important features. Neural networks, on the other hand, often require in-depth expertise regarding features normalization, architecture design, regularization, or loss functions, and they also require specialized hardware (GPU) for high performance. Furthermore, neural models usually require effort and time for better results, due to sensitiveness to pre-processing, architecture design and hyper-parameter choices. The problem of finding an network architecture (including the structure, number of layers and nodes) alone can open a vast design space in which a well-performing solution might lie. As competitions usually have a relatively short time span, DL models may not be not the first option for contestants when given tabular data.[9]

In our experience, challenge contestants usually invest substantial amounts time on feature exploration and engineering when working with tabular data. On the other hand, academic researchers usually focus more on scientific aspects, such as exploring and proposing complex training algorithms and neural architecture designs for a given problem or domain. They are generally not focusing on feature engineering and leakage exploitation in their experiments. It is not common, for example, to have papers reporting experiments comparing novel neural architectures for hybrid recommendation against XGBoost models, where both models use the same rich set of features.

4 IS THE EVALUATION PROCESS DIFFERENT?

How the performance comparison experiments are set up and how the evaluation is actually done is quite different in academic research and in competitions. We illustrate the main differences in Table 1.

The difference between these two ways of benchmarking algorithms is striking. The way it is done in competitions appears to be more objective and not prone to potential biases by researchers who are the only ones who evaluate their own proposals before publication. In academic settings, researchers in fact have a lot of freedom when they decide on the specifics of the experimental setup. Having this freedom is absolutely important, because it allows researchers for example to explore novel configurations and research questions that were not investigated before. A potential downside of this freedom might be that researchers might consciously or unconsciously end up with experimental configurations that favor their hypothesis that their newly proposed method is better than previous ones. A typical issue in the context can lie in the selection and optimization of the baselines. The potential result is that the reported progress is only virtual because, e.g., the baselines were too weak or not as well optimized as a new method. Recent research provides several examples where DL methods were—in contrast to what was reported in the papers—not consistently outperforming existing and often quite simple methods [8, 20, 32]. Of course, we cannot know to what extent the consistent win of DL methods over previous approaches in the academic literature is due to methodological issues. This is particularly unclear because similar issues were observed before the DL era, e.g., in the field of information retrieval [2].

Overall, we can observe that the evaluation approach in academic research and in competitions is different in terms of who designs the experiment and who does the measurement. In the end, it however remains unclear if these differences are strongly tied to the observed discrepancies between the winning models in academic papers and in competitions. A similar difference is seen in industry where the success of recommenders in terms of organization-oriented, often longitudinal Key Performance Indicators [12]. For example, Google [5] reported from their online experiments a gain of 3.9% in *Online Acquisition Gain* with their Wide&Deep model, compared to a Linear model with the same set of features. On a more positive note, we can increasingly observe that academic researchers care more about reproducibility by making codes, hyper-parameter choice and datasets publicly available. This should lead to the effect that other researchers have the opportunity to objectively and independently validate the experiments and outcomes.

5 DISCUSSION AND CONCLUSIONS

Our work has highlighted a number of potential reasons why DL methods do not consistently win recommender systems competitions. Alternative reasons might exist as well, e.g., that it is just a coincidence that happened in the particular competition series. Actually, there was a second-place solution in one year that was purely based on DL. It could be that DL methods "just don't work well" for these types of problems. Such a generalizing explanation however seems unlikely, given the success of DL in various other application areas of machine learning and in industry.

Overall, we believe that the potential of DL methods for recommendation problems has not been fully exploited yet. Maybe we still need better methods that are more effective in combining different information sources in parallel. Previous works, for example in the area of session-based and sequential recommendations, indicate that combining information sources can be key to the success of DL methods [10]. This is particularly the case when there are certain specifics to be considered like in the news domain, where we have a permanent item cold-start problem [6]. Another potential way to achieve better results with DL in competitions may lie in the development and use of tools that lead to high performance "out-of-the-box", i.e., without the need of extensive feature engineering and neural architecture design. In the context of DL, such *AutoML* techniques for example include *Neural Architecture Search* [7].

Our discussions however also highlight the importance of not forgetting about non-DL methods in academic research. In recent years, we sometimes observe that newly proposed methods are only compared with other DL methods, and that previous approaches are not considered anymore. When it later turns out that these baseline DL methods were not necessarily better than what we had

[9] In recommender competitions tabular data is common. In other domains where DL is successful, we often see other types of data, e.g., images, text or graphs.

Table 1: Comparison of Evaluation Process between Academic Research and Competitions

Aspect	Academic Research	Machine Learning Competition
Selection of dataset	By researcher	By competition host
Selection and optimization of baselines	By researcher	Represented by other participants
Selection of evaluation metric(s)	By researcher	By competition host
Selection of protocol specifics[10]	By researcher	By competition host
Execution of measurement	By researcher	By competition host
Publication of measurement results	By researcher	By competition host
Test data	Available to researcher	Never available to participant
Source code sharing	Researcher may share	Must be shared sometimes
Dataset sharing	Researcher may share	Training data available to all participants

before—e.g., due to methodological issues mentioned above—than we end up again with "improvements that don't add up" [2, 8]. This pattern of DL focused baselines also occurs in the online evaluation provided on industry datasets and this advice may also be applicable there as well.

Finally, one might question the importance of machine learning competitions for scientific process in general. Such competitions in some ways reinforce a "leaderboard chasing" culture, where the main and often only goal is to outperform previous methods by a few percent on a set of accuracy measures. This may lead to the effect that the question whether or not these improvements matter in practice is never asked. Also, it becomes irrelevant *why* a certain solution led an improvement [16], because such a research approach is not based on underlying theories or research hypotheses. On the other hand, competitions can have a number of positive effects on scientific research. Through such competitions, organizations can for example share problems that matter to them with the academic community. Moreover, competitions are one of the most important sources for datasets for academic researchers, and they are also a helpful means to engage researchers to build continuously better recommender systems in the future.

REFERENCES

[1] Sebastiano Antenucci, Simone Boglio, Emanuele Chioso, Ervin Dervishaj, Shuwen Kang, Tommaso Scarlatti, and Maurizio Ferrari Dacrema. 2018. Artist-Driven Layering and User's Behaviour Impact on Recommendations in a Playlist Continuation Scenario. In *Proceedings of the ACM Recommender Systems Challenge 2018 (RecSys Challenge '18)*.

[2] 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)* (Hong Kong, China). 601–610.

[3] Mattia Bianchi, Federico Cesaro, Filippo Ciceri, Mattia Dagrada, Alberto Gasparin, Daniele Grattarola, Ilyas Inajjar, Alberto Maria Metelli, and Leonardo Cella. 2017. Content-Based Approaches for Cold-Start Job Recommendations. In *Proceedings of the Recommender Systems Challenge 2017 (RecSys Challenge '17)*.

[4] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior Sequence Transformer for e-commerce recommendation in Alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[6] Gabriel de Souza Pereira Moreira, Dietmar Jannach, and Adilson Marques da Cunha. 2019. Contextual Hybrid Session-based News Recommendation with Recurrent Neural Networks. *IEEE Access* 7 (2019), 169185–169203.

[7] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural Architecture Search: A Survey. *Journal of Machine Learning Research* 20, 55 (2019), 1–21.

[8] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 2019 ACM Conference on Recommender Systems (RecSys 2019)*.

[9] Udit Gupta, Carole-Jean Wu, Xiaodong Wang, Maxim Naumov, Brandon Reagen, David Brooks, Bradford Cottel, Kim Hazelwood, Mark Hempstead, Bill Jia, et al. 2020. The architectural implications of facebook's DNN-based personalized recommendation. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 488–501.

[10] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-Rich Session-Based Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. 241–248.

[11] Paweł Jankiewicz, Liudmyla Kyrashchuk, Paweł Sienkowski, and Magdalena Wójcik. 2019. Boosting Algorithms for a Session-Based, Context-Aware Recommender System in an Online Travel Domain. In *Proceedings of the Workshop on ACM Recommender Systems Challenge (RecSys Challenge '19)*.

[12] Dietmar Jannach and Michael Jugovac. 2019. Measuring the Business Value of Recommender Systems. *ACM Transactions on Management Information Systems* 10, 4 (2019).

[13] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 43–50.

[14] Huang Kung-Hsiang, Fu Yi-Fu, Lee Yi-Ting, Lee Tzong-Hann, Chan Yao-Chun, Lee Yi-Hui, and Lin Shou-De. 2019. A-HA: A Hybrid Approach for Hotel Recommendation. In *Proceedings of the Workshop on ACM Recommender Systems Challenge (RecSys Challenge '19)*.

[15] Jianxun Lian, Fuzheng Zhang, Min Hou, Hongwei Wang, Xing Xie, and Guangzhong Sun. 2017. Practical Lessons for Job Recommendations in the Cold-Start Scenario. In *Proceedings of the Recommender Systems Challenge 2017 (RecSys Challenge '17)*.

[16] Zachary C. Lipton and Jacob Steinhardt. 2018. Troubling Trends in Machine Learning Scholarship. arXiv:arXiv:1807.03341

[17] Malte Ludewig and Dietmar Jannach. 2019. Learning to Rank Hotels for Search and Recommendation from Session-based Interaction Logs and Meta Data. In *Proceedings of the ACM RecSys Challenge 2019 Workshop at ACM RecSys 2019*.

[18] Malte Ludewig, Michael Jugovac, and Dietmar Jannach. 2017. A Light-Weight Approach to Recipient Determination When Recommending New Items. In *Proceedings of the Recommender Systems Challenge 2017 (RecSys Challenge '17)*.

[19] Malte Ludewig, Iman Kamehkhosh, Nick Landia, and Dietmar Jannach. 2018. Effective Nearest-Neighbor Music Recommendations. In *Proceedings of the ACM RecSys Challenge 2018 Workshop at ACM RecSys 2018*.

[20] Malte Ludewig, Noemi Mauro, Sara Latifi, and Dietmar Jannach. 2019. Performance Comparison of Neural and Non-Neural Approaches to Session-based Recommendation. In *Proceedings of the 2019 ACM Conference on Recommender Systems (RecSys 2019)*.

[21] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1222–1230.

[22] G. D. S. P. Moreira, D. Jannach, and A. M. da Cunha. 2019. Contextual Hybrid Session-based News Recommendation with Recurrent Neural Networks. *IEEE Access* 7 (2019), 169185–169203.

[23] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. 2011. Multimodal Deep Learning.. In *ICML 2011*. 689–696.

[24] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *Comput. Surveys* 51, 4 (2018).

[25] Vasiliy Rubtsov, Mikhail Kamenshchikov, Ilya Valyaev, Vasiliy Leksin, and Dmitry I. Ignatov. 2018. A Hybrid Two-Stage Recommender System for Automatic Playlist Continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018 (RecSys Challenge '18)*.

[26] Masahiro Sato, Koki Nagatani, and Takuji Tahara. 2017. Exploring an Optimal Online Model for New Job Recommendation: Solution for RecSys Challenge 2017. In *Proceedings of the Recommender Systems Challenge 2017 (RecSys Challenge '17)*.

[27] Maksims Volkovs, Himanshu Rai, Zhaoyue Cheng, Ga Wu, Yichao Lu, and Scott Sanner. 2018. Two-Stage Model for Automatic Playlist Continuation at Scale. In *Proceedings of the ACM Recommender Systems Challenge 2018*.

[28] Maksims Volkovs, Anson Wong, Zhaoyue Cheng, Felipe Pérez, Ilya Stanevich, and Yichao Lu. 2019. Robust Contextual Models for In-Session Personalization. In *Proceedings of the Workshop on ACM Recommender Systems Challenge (RecSys Challenge '19)*.

[29] Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. 2017. Content-Based Neighbor Models for Cold Start in Recommender Systems. In *Proceedings of the Recommender Systems Challenge 2017 (RecSys Challenge '17)*.

[30] Zhe Wang, Yangbo Gao, Huan Chen, and Peng Yan. 2019. Session-Based Item Recommendation with Pairwise Features. In *Proceedings of the Workshop on ACM Recommender Systems Challenge (RecSys Challenge '19)*.

[31] Hojin Yang, Yoonki Jeong, Minjin Choi, and Jongwuk Lee. 2018. MMCF: Multimodal Collaborative Filtering for Automatic Playlist Continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018 (RecSys Challenge '18)*.

[32] Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. 2019. Critically Examining the Neural Hype: Weak Baselines and the Additivity of Effectiveness Gains from Neural Ranking Models. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. 1129–1132.

[33] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 974–983.

[34] Weijie Zhao, Deping Xie, Ronglai Jia, Yulei Qian, Ruiquan Ding, Mingming Sun, and Ping Li. 2020. Distributed Hierarchical GPU Parameter Server for Massive Scale Deep Learning Ads Systems. *arXiv preprint arXiv:2003.05622* (2020).