User Modeling and User-Adapted Interaction
https://doi.org/10.1007/s11257-020-09277-1

Empirical analysis of session-based recommendation algorithms

A comparison of neural and non-neural approaches

Malte Ludewig[1] · **Noemi Mauro**[2] · **Sara Latifi**[3] · **Dietmar Jannach**[3]

Received: 1 November 2019 / Accepted in revised form: 12 September 2020

Abstract
Recommender systems are tools that support online users by pointing them to potential items of interest in situations of information overload. In recent years, the class of session-based recommendation algorithms received more attention in the research literature. These algorithms base their recommendations solely on the observed interactions with the user in an ongoing session and do not require the existence of long-term preference profiles. Most recently, a number of deep learning-based ("neural") approaches to session-based recommendations have been proposed. However, previous research indicates that today's complex neural recommendation methods are not always better than comparably simple algorithms in terms of prediction accuracy. With this work, our goal is to shed light on the state of the art in the area of session-based recommendation and on the progress that is made with neural approaches. For this purpose, we compare twelve algorithmic approaches, among them six recent neural methods, under identical conditions on various datasets. We find that the progress in terms of prediction accuracy that is achieved with neural methods is still limited. In most cases, our experiments show that simple heuristic methods based on nearest-neighbors schemes are preferable over conceptually and computationally more complex methods. Observations from a user study furthermore indicate that recommendations based on heuristic methods were also well accepted by the study participants. To support future progress and reproducibility in this area, we publicly share the SESSION-REC evaluation framework that was used in our research.

Keywords Session-based recommendation · Performance evaluation · Reproducibility

This work combines and significantly extends our own previous work published in Ludewig and Jannach (2019) and Ludewig et al. (2019). This paper or a similar version is not currently under review by a journal or conference. This paper is void of plagiarism or self-plagiarism as defined by the Committee on Publication Ethics and Springer Guidelines. A preprint version of this work is available at https://arxiv.org/abs/1910.12781.

Extended author information available on the last page of the article

Published online: 20 October 2020

1 Introduction

Recommender systems (RS) are software applications that help users in situations of information overload, and they have become a common feature on many modern online services. Collaborative filtering (CF) techniques, which are based on behavioral data collected from larger user communities, are among the most successful technical approaches in practice. Historically, these approaches mostly rely on the assumption that information about longer-term preferences of the individual users is available, e.g., in the form of a user–item rating matrix (Resnick et al. 1994). In many real-world applications, however, such longer-term information is often not available, because users are not logged in or because they are first-time users. In such cases, techniques that leverage behavioral patterns in a community can still be applied (Jannach and Zanker 2019). The difference is that instead of the long-term preference profiles only the observed interactions with the user in the ongoing session can be used to adapt the recommendations to the assumed needs, preferences, or intents of the user. Such a setting is usually termed a *session-based recommendation* problem (Quadrana et al. 2018).

Interestingly, research on session-based recommendation was very scarce for many years despite the high practical relevance of the problem setting. Only in recent years, we can observe an increased interest in the topic in academia (Wang et al. 2019), which is at least partially caused by the recent availability of public datasets in particular from the e-commerce domain. This increased interest in session-based recommendations coincides with the recent boom of deep learning (neural) methods in various application areas. Accordingly, it is not surprising that several neural session-based recommendation approaches have been proposed in recent years, with GRU4REC being one of the pioneering and most cited works in this context (Hidasi et al. 2016a).

From the perspective of the evaluation of session-based algorithms, the research community—at the time when the first neural techniques were proposed—had not yet established a level of maturity as is the case for problem setups that are based on the traditional user–item rating matrix. This led to challenges that concerned both the question what represents the state of the art in terms of algorithms and the question of the evaluation protocol when time-ordered user interaction logs are the input instead of a rating matrix. Partly due to this unclear situation, it soon turned out that in some cases comparably simple non-neural techniques, in particular ones based on nearest-neighbors approaches, can lead to very competitive or even better results than neural techniques (Jannach and Ludewig 2017; Ludewig and Jannach 2018). Besides being competitive in terms of accuracy, such more simple approaches often have the advantage that their recommendations are more transparent and can more easily be explained to the users. Furthermore, these simpler methods can often be updated online when new data become available, without requiring expensive model retraining.

However, during the last few years after the publication of GRU4REC, we have mostly observed new proposals in the area of complex models. With this work, our aim is to assess the progress that was made in the last few years in a

reproducible way. To that purpose, we have conducted an extensive set of experiments in which we compared twelve session-based recommendation techniques under identical conditions on a number of datasets. Among the examined techniques, there are six recent neural approaches, which were published at highly ranked publication outlets such as KDD, AAAI, or SIGIR after the publication of the first version of GRU4REC in 2015.[1]

The main outcome of our offline experiments is that the progress that is achieved with neural approaches to session-based recommendation is still limited. In most experiment configurations, one of the simple techniques outperforms all the neural approaches. In some cases, we could also not confirm that a more recently proposed neural method consistently outperforms the much earlier GRU4REC method. Generally, our analyses point to certain underlying methodological issues, which were also observed in other application areas of applied machine learning. Similar observations regarding the competitiveness of established and often more simple approaches were made before, e.g., for the domains of information retrieval, time-series forecasting, and recommender systems (Yang et al. 2019; Ferrari Dacrema et al. 2019b; Makridakis et al. 2018; Armstrong et al. 2009), and it is important to note that these phenomena are not tied to deep learning approaches.

To help overcome some of these problems for the domain of session-based recommendation, we share our evaluation framework SESSION-REC online[2]. The framework not only includes the algorithms that are compared in this paper; it also supports different evaluation procedures, implements a number of metrics, and provides pointers to the public datasets that were used in our experiments.

Since offline experiments cannot inform us about the quality of the recommendation as *perceived* by users, we have furthermore conducted a user study. In this study, we compared heuristic methods with a neural approach and the recommendations produced by a commercial system (SPOTIFY) in the context of an online radio station. The main outcomes of this study are that heuristic methods also lead to recommendations—playlists in this case—that are well accepted by users. The study furthermore sheds some light on the importance of other quality factors in the particular domain, i.e., the capability of an algorithm to help users discover new items.

The paper is organized as follows. Next, in Sect. 2, we provide an overview of the algorithms that were used in our experiments. Section 3 describes our offline evaluation methodology in more detail, and Sect. 4 presents the outcomes of the experiments. In Sect. 5, we report the results of our user study. Finally, we summarize our findings and their implications in Sect. 7.

[1] Compared to our previous work presented in Ludewig and Jannach (2018) and Ludewig et al. (2019), our present analysis includes considerably more recent deep learning techniques and baseline approaches. We also provide the outcomes of additional measurements regarding the scalability and stability of different algorithms. Finally, we also contrast the outcomes of the offline experiments with the findings obtained in a user study (Ludewig and Jannach 2019).

[2] https://github.com/rn5l/session-rec.

2 Algorithms

Algorithms of various types were proposed over the years for session-based recommendation problems. A detailed overview of the more general family of *sequence-aware recommender systems*, where session-based ones are a part of, can be found in Quadrana et al. (2018). In the context of this work, we limit ourselves to a brief summary of parts of the historical development and how we selected algorithms for inclusion in our evaluations.

2.1 Historical development and algorithm selection

Nowadays, different forms of session-based recommendations can be found in practical applications. The recommendation of *related items* for a given reference object can, for example, be seen as a basic and very typical form of session-based recommendations in practice. In such settings, the selection of the recommendations is usually based solely on the very last item viewed by the user. Common examples are the recommendation of additional articles on news Web sites or recommendations of the form "Customers who bought ...also bought" on e-commerce sites. Another common application scenario is the creation of automated playlists, e.g., on YouTube, Spotify, or Last.fm. Here, the system creates a virtually endless list of next-item recommendations based on some seed item and additional observations, e.g., skips or likes, while the media is played. These application domains—Web page and news recommendation, e-commerce, music playlists—also represent the main driving scenarios in academic research.

For the recommendation of *Web pages* to visit, Mobasher et al. proposed one of the earliest session-based approaches based on frequent pattern mining in 2002 (Mobasher et al. 2002). In 2005, Shani et al. (2005) investigated the use of an MDP-based (Markov Decision Process) approach for session-based recommendations in *e-commerce* and also demonstrated its value from a business perspective. Alternative technical approaches based on Markov processes were later on proposed in 2012 and 2013 for the *news* domain in Garcin et al. (2012, 2013).

An early approach to *music playlist generation* was proposed in 2005 (Ragno et al. 2005), where the selection of items was based on the similarity with a seed song. The music domain was, however, also very important for collaborative approaches. In 2012, the authors of Hariri et al. (2012) used a session-based nearest-neighbors technique as part of their approach for playlist generation. This nearest-neighbors method and improved versions thereof later on turned out to be highly competitive with today's neural methods (Ludewig and Jannach 2018). More complex methods were also proposed for the music domain, e.g., an approach based on Latent Markov Embeddings (Chen et al. 2012) from 2012.

Some novel technical proposals in the years 2014 and 2015 were based on a non-public *e-commerce* dataset from a European fashion retailer and either used Markov processes and side information (Tavakol and Brefeld 2014) or a simple re-ranking scheme based on short-term intents (Jannach et al. 2015). More importantly, however, in the year 2015, the ACM RecSys conference hosted a challenge, where the

problem was to predict if a consumer will make a purchase in a given session, and if so, to predict which item will be purchased. A corresponding dataset (YOOCHOOSE) was released by an industrial partner, which is very frequently used today for benchmarking session-based algorithms. Technically, the winning team used a two-stage classification approach and invested a lot of efforts into feature engineering to make accurate predictions (Romov and Sokolov 2015).

In late 2015, Hidasi et al. (2016a) then published the probably first deep learning-based method for session-based recommendation called GRU4REC, a method which was continuously improved later on, e.g., in Hidasi and Karatzoglou (2018) or Tan et al. (2016). In their work, they also used the mentioned YOOCHOOSE dataset for evaluation, although with the slightly different optimization goal, i.e., to predict the immediate next item click event. As one of their baselines, they used an item-based nearest-neighbors technique. They found that their neural method is significantly better than this technique in terms of prediction accuracy. The proposal of their method and the booming interest in neural approaches subsequently led to a still ongoing wave of new proposals that apply deep learning approaches to session-based recommendation problems.

In the present work, we consider a selection of algorithms that reflects these historical developments. We consider basic algorithms based on item co-occurrences, sequential patterns and Markov processes as well as methods that implement session-based nearest-neighbors techniques. Looking at neural approaches, we benchmark the latest versions of GRU4REC as well as five other methods that were published later and which state that they outperform at least the initial version of GRU4REC to a significant extent.

Regarding the selected neural approaches, we limit ourselves to methods that do not use side information about the items in order to make our work easily reproducible and not dependent on such metadata. Another constraint for the inclusion in our comparison is that the work was published in major conferences, i.e., one that is rated A or A* according to the Australian CORE scheme. Finally, while in theory algorithms should be reproducible based on the technical descriptions in the paper, there are usually many small implementation details that can influence the outcome of the measurement. Therefore, like in Ferrari Dacrema et al. (2019b), we only considered approaches where the source code was available and could be integrated in our evaluation framework with reasonable effort.

2.2 Considered algorithms

In total, we considered 12 algorithms in our comparison. Table 1 provides an overview of the *non-neural* methods. Table 2 correspondingly shows the neural methods considered in our analysis, ordered by their publication date.

Except for the CT method, the non-neural methods from Table 1 are conceptually very simple or almost trivial. As mentioned above, this can lead to a number of potential practical advantages compared to more complex models, e.g., regarding online updates and explainability. From the perspective of the computational costs, the time needed to "train" the simple methods is often low, as this phase often

Table 1 Overview of the *non-neural* methods compared in our analysis

AR	This simple "Association Rules" method counts pairwise item co-occurrences in the training sessions. Recommendations for an ongoing session are generated by this method by returning those items that most frequently co-occurred with the last item of the current session in the past. For a formal definition, see Ludewig and Jannach (2018).
SR	The method called "Sequential Rules" was proposed in Ludewig and Jannach (2018). It is similar to AR in that it counts pairwise item co-occurrences in the training sessions. In addition to AR, however, it considers the order of the items in a session and the distance between them using a decay function. The method often led to competitive results in particular in terms of the Mean Reciprocal Rank in the analysis in Ludewig and Jannach (2018).
SKNN/V-SKNN	The analysis in Jannach and Ludewig (2017) showed that a simple session-based nearest-neighbors method similar to the one from Hariri et al. (2015) was competitive with the first version for GRU4REC. Conceptually, the idea is to find past sessions that contain the same elements as the ongoing session. The recommendations are then based on selecting items that appeared in the most similar past session. Since the sequence in which items are consumed in the ongoing user session might be of importance in the recommendation process, a number of "sequential extensions" to the SKNN method were proposed in Ludewig and Jannach (2018). Here, the order of the items in a session is proved to be helpful, both when calculating the similarities and in the item scoring process. Furthermore, according to Ludewig et al. (2018) it can be beneficial to put more emphasis on less popular items by applying an Inverse Document Frequency (IDF) weighting scheme. In this paper, all those extensions are implemented in the V-SKNN method.
STAN	The method called "Sequence and Time Aware Neighborhood" was presented at SIGIR '19 (Garg et al. 2019). STAN is based on SKNN (Jannach and Ludewig 2017), but it additionally takes into account the following factors for making recommendations: i) the position of an item in the current session, ii) the recency of a past session w.r.t. to the current session, and iii) the position of a recommendable item in a neighboring session. Their results show that STAN significantly improves over SKNN and is even comparable to recently proposed state-of-the-art deep learning approaches.
VSTAN	This method, which we propose in this present paper, combines the ideas from STAN and V-SKNN in a single approach. It incorporates all three previously mentioned particularities of STAN, which already share some similarities with the V-SKNN method. Furthermore, we add a sequence-aware item scoring procedure as well as the IDF weighting scheme from V-SKNN.
CT	This technique is based on Context Trees, which were originally proposed for lossless data compression. It is a non-parametric method and based on variable-order Markov models. The method was proposed in Mi and Faltings (2018), where it showed promising results.

reduces to counting item co-occurrences in the training data or to preparing some in-memory data structures. To make the nearest-neighbors technique scalable, we implemented the internal data structures and data sampling strategies proposed in Jannach and Ludewig (2017). Specifically, we pre-process the training data to build fast in-memory look-up tables. These tables can then be used to almost immediately retrieve a set of potentially relevant neighbor sessions in the training data given an item in the test session. Furthermore, to speed up processing times, we sample only a fraction (e.g., 1000) of the most recent training sessions when we look for neighbors, as this proved effective in several application domains. In the end, the CT

Table 2 Overview of the *neural* methods compared in our analysis

GRU4REC	GRU4REC (Hidasi et al. 2016a) was the first neural approach that employed RNNs for session-based recommendation. This technique uses Gated Recurrent Units (GRUs) (Cho et al. 2014) to deal with the vanishing gradient problem. The technique was later on improved using more effective loss functions (Hidasi and Karatzoglou 2018).
NARM	This model (Li et al. 2017) extends GRU4REC and improves its session modeling with the introduction of a hybrid encoder with an attention mechanism. The attention mechanism is in particular used to consider items that appeared earlier in the session and which are similar to the last clicked one. The recommendation scores for each candidate item are computed with a bilinear matching scheme based on the unified session representation.
STAMP	In contrast to NARM, this model (Liu et al. 2018) does not rely on an RNN. A short-term attention/memory priority model is proposed, which (a) is capable of capturing the users' general interests from the long-term memory of a session context and (b) also takes the users' most recent interests from the short-term memory into account. The users' general interests are captured by an external memory built from all the historical clicks in a session prefix (including the last click). The attention mechanism is built on top of the embedding of the last click that represents the user's current interests.
NEXTITNET	This recent model (Yuan et al. 2019) also discards RNNs to model user sessions. In contrast to STAMP, convolutional neural networks are adopted with a few domain-specific enhancements. The generative model is designed to explicitly encode item inter-dependencies, which allows to directly estimate the distribution of the output sequence (rather than the desired item) over the raw item sequence. Moreover, to ease the optimization of the deep generative architecture, the authors propose to use residual networks to wrap convolutional layer(s) by residual block.
SR-GNN	This method (Wu et al. 2019) models session sequences as graph structured data (i.e., directed graphs). Based on the session graph, SR-GNN is capable of capturing transitions of items and generating item embedding vectors correspondingly, which are difficult to be revealed by conventional sequential methods like MC-based and RNN-based methods. With the help of item embedding vectors, SR-GNN furthermore aims to construct reliable session representations from which the next-click item can be inferred.
CSRM	This method (Wang et al. 2019) is a hybrid framework that uses collaborative neighborhood information in session-based recommendations. CSRM consists of two parallel modules: an Inner Memory Encoder (IME) and an Outer Memory Encoder (OME). The IME models a user's own information in the current session with the help of Recurrent Neural Networks (RNNs) and an attention mechanism. The OME exploits collaborative information to better predict the intent of current sessions by investigating neighborhood sessions. Then, a fusion gating mechanism is used to selectively combine information from the IME and OME to obtain the final representation of the current session. Finally, CSRM obtains a recommendation score for each candidate item by computing a bi-linear match with the final representation of the current session.

method was the only one from the set of non-neural methods for which we encountered scalability issues in the form of memory consumption and prediction time when the set of recommendable items is huge.

Regarding alternative non-neural approaches, note that in our previous evaluation in Ludewig and Jannach (2018) only one neural method, but several other machine learning approaches were benchmarked. We do not include these alternative machine

Table 3 Overview of the baseline techniques that each neural session-based approach was originally compared to

Method	Publication	IKNN	SKNN	BPR-MF	FPMC	GRU4REC	NARM	STAMP
GRU4REC	ICLR (05/16)	✗		✗				
GRU4REC+	RecSys (09/16)	✗				✗		
NARM	CIKM (11/17)	✗		✗	✗	✗		
STAMP	KDD (08/18)	✗			✗	✗	✗	
GRU4REC2	CIKM (10/18)	✗				✗		
NEXTITNET	WSDM (02/19)					✗		
SR-GNN	AAAI (02/19)	✗		✗	✗	✗	✗	✗
CSRM	SIGIR (07/19)	✗	✗	✗	✗	✗	✗	

The methods are ordered chronologically by the date of publication. The marks (✗) indicate which baselines were used in the comparison

learning methods (IKNN, FPMC, MC, SMF, BPR-MF, FISM, FOSSIL)[3]. In our present analysis, because the findings in Ludewig and Jannach (2018) showed that they either are generally not competitive or only lead to competitive results in few special cases.

The development over time regarding the *neural* approaches is summarized in Table 3. The table also indicates which baselines were used in the original papers. The analysis shows that GRU4REC was considered as a baseline in all papers. Most papers refer to the original GRU4REC publication from 2016 or an early improved version that was proposed shortly afterward (which we term GRU4REC+ here, see Tan et al. 2016). Most papers, however, do not refer to the improved version (GRU4REC2) discussed in Hidasi and Karatzoglou (2018). Since the public code for GRU4REC was constantly updated, we, however, assume that the authors ran benchmarks against the updated versions. NARM, as one of the earlier neural techniques, is the only neural method other than GRU4REC that is considered quite frequently by more recent works.

The analysis of the used baselines furthermore showed that only one of the more recent papers proposing a neural method (CSRM) considers, i.e., Wang et al. (2019), session-based nearest-neighbors techniques as a baseline, even though their competitiveness was documented in a publication at the ACM Recommender Systems conference in 2017 (Jannach and Ludewig 2017). Wang et al. (2019) (CSRM), however, only consider the original proposal and not the improved versions from 2018 (Ludewig and Jannach 2018). The only other papers in our analysis, which consider session-based nearest-neighbors techniques as baselines, are about non-neural techniques (CT and STAN). The paper proposing STAN furthermore is an exception in that since it considers quite a number of neural approaches (GRU4REC2, STAMP, NARM, SR-GNN) in its comparison.

[3] IKNN: Item-based kNN (Hidasi et al. 2016a), FPMC: Factorized Personalized Markov Chains (Rendle et al. 2010), MC: Markov Chains (Norris 1997), SMF: Session-based Matrix Factorization (Ludewig and Jannach 2018), BPR-MF: Bayesian Personalized Ranking (Rendle et al. 2009), FISM: Factored Item Similarity Models (Kabbur et al. 2013), FOSSIL: FactOrized Sequential Prediction with Item SImilarity ModeLs (He and McAuley 2016).

Table 4 Datasets used in the experiments

RSC15	An e-commerce dataset used in the 2015 ACM RecSys Challenge
RETAIL	An e-commerce dataset from the company Retail Rocket
DIGI	An e-commerce dataset shared by the company Diginetica
ZALANDO	A non-public dataset consisting of interaction logs from the European fashion retailer Zalando
30MU	Music listening logs obtained from Last.fm
NOWP	Music listening logs obtained from Twitter
AOTM	A public music dataset containing music playlists
8TRACKS	A private music dataset with handcrafted playlists

3 Evaluation methodology

We benchmarked all methods under the same conditions, using the evaluation framework that we share online to ensure reproducibility of our results.

3.1 Datasets

We considered eight datasets from two domains for our evaluation, e-commerce and music. Six of them are public and several of them were previously used to benchmark session-based recommendation algorithms. Table 4 briefly describes the datasets.

We pre-processed the original datasets in a way that all sessions with only one interaction were removed. As done in previous works, we also removed items that appeared less than five times in the dataset. Multiple interactions with the same item in one session were kept in the data. While the repeated recommendation of an item does not lead to item discovery, such recommendations can still be helpful from a user's perspective, e.g., as reminders (Ren et al. 2019; Lerche et al. 2016; Jannach et al. 2017). Furthermore, we use an evaluation procedure where we run repeated measurements on several subsets (splits) of the original data; see Sect. 3.2. The average characteristics of the subsets for each dataset are shown in Table 5. We share all datasets except ZALANDO and 8TRACKS online.

3.2 Evaluation procedure and metrics

3.2.1 Data and splitting approach

We apply the following procedure to create train–test splits. Since most datasets consist of time-ordered events, usual cross-validation procedures with the randomized allocation of events across data splits cannot be applied. Several authors only use one single time-ordered training–testing split for their measurements. This, however, can lead to undesired random effects. We therefore rely on a protocol where we create five non-overlapping and contiguous subsets (splits) of the datasets. As

Table 5 Characteristics of the datasets

Dataset	RSC15	RETAIL	DIGI	ZALANDO	30MU	NOWP	AOTM	8TRACKS
Actions	5.4M	210k	264k	4.5M	640k	271k	307k	1.5M
Sessions	1.4M	60k	55k	365k	37k	27k	22k	132k
Items	29k	32k	32k	189k	91k	75k	91k	376k
Days cov.	31	27	31	90	90	90	90	90
Actions/Sess.	3.95	3.54	4.78	12.43	17.11	10.04	14.02	11.32
Items/Sess.	3.17	2.56	4.01	8.39	14.47	9.38	14.01	11.31
Actions/Day	175k	8k	8.5k	50k	7k	3.0k	3.4k	16.6k
Sessions/Day	44k	2.2k	1.7k	4k	300	243	243	1.4k

The values are averaged over all five splits

done in previous works, we use the last n days of each split for evaluation (testing) and the other days for training the models.[4] The reported measurements correspond to the averaged results obtained for each split. The playlist datasets (AOTM and 8TRACKS) are exceptions here as they do not have timestamps. For these datasets, we therefore randomly generated timestamps, which allows us to use the same procedure as for the other datasets. Note that during the evaluation, we only considered items in the test split that appeared at least once in the training data.

3.2.2 Hyperparameter optimization

Proper hyperparameter tuning is essential when comparing machine learning approaches. We therefore tuned all hyperparameters for all methods and datasets in a systematic approach, using MRR@20 as an optimization target as done in previous works. Technically, we created subsets from the training data for validation. The size of the validation set was chosen in a way that it covered the same number of days that was used in the final test set. We applied a random hyperparameter optimization approach with 100 iterations as done in Hidasi and Karatzoglou (2018), Liu et al. (2018) and Li et al. (2017). Since NARM and CSRM only have a smaller set of hyperparameters, we only had to do 50 iterations for these methods. Since the tuning process was particularly time-consuming for SR-GNN and NEXTITNET, we had to limit the number of iterations to 50 both for SR-GNN on the ZALANDO dataset and for NEXTITNET on the RSC15 dataset. The final hyperparameter values for each method and dataset can be found online, along with a description of the investigated ranges.

[4] The number of days used for testing (n) was determined based on the characteristics of the dataset. We, for example, used the last day for the RSC15 dataset, two for RETAIL, five for the music datasets, and seven for DIGI to ensure that train–test splits are comparable.

3.2.3 Accuracy measures

For each session in the test set, we incrementally reveal one event of a session after the other, as was proposed in Hidasi et al. (2016a)[5]. The task of the recommendation algorithm is to generate a prediction for the next event(s) in the session in the form of a ranked list of items. The resulting list can then be used to apply standard accuracy measures from information retrieval. The measurement can be done in two different ways.

- As in Hidasi et al. (2016a) and other works, we can measure if the immediate next item is part of the resulting list and at which position it is ranked. The corresponding measures are the Hit Rate and the Mean Reciprocal Rank.
- In typical information retrieval scenarios, however, one is usually not interested in having one item right (e.g., the first search result), but in having as many predictions as possible right in a longer list that is displayed to the user. For session-based recommendation scenarios, this applies as well, as usually, e.g., on music and e-commerce sites, more than one recommendation is displayed. Therefore, we measure Precision and Recall in the usual way, by comparing the objects of the returned list with the entire remaining session, assuming that not only the immediate next item is relevant for the user. In addition to Precision and Recall, we also report the Mean Average Precision metric.

The most common cutoff threshold in the literature is 20, probably because this was the chosen threshold by the authors of GRU4REC (Hidasi et al. 2016a). We have made measurements for alternative list lengths as well, but will only report the results when using 20 as a list length in this paper. We report additional results for cutoff thresholds of 5 and 10 in an online appendix.[6]

3.2.4 Coverage and popularity

Depending on the application domain, factors other than prediction accuracy might be relevant as well, including coverage, novelty, diversity, or serendipity (Shani and Gunawardana 2011). Since we do not have information about item characteristics, we focus on questions of coverage and novelty in this work.

With *coverage*, we here refer to what is sometimes called "aggregate diversity" (Adomavicius and Kwon 2012). Specifically, we measure the fraction of items of the catalog that ever appears in any top-n list presented to the users in the test set. This coverage measure in a way also evaluates the level of context adaptation, i.e., if an algorithm tends to recommend the same set of items to everyone or specifically varies the recommendations for a given session.

[5] Note that the revealed items from a session can be used by an algorithm for the subsequent predictions, but the revealed interactions are not added to the training data.

[6] https://rn5l.github.io/session-rec/umuai.

We approximate the *novelty* level of an algorithm by measuring how popular the recommended items are on average. The underlying assumption is that recommending more unpopular items leads to higher novelty and discovery effects. Algorithms that mostly focus on the recommendation of popular items might be undesirable from a business perspective, e.g., when the goal is to leverage the potential of the long tail in e-commerce settings. Technically, we measure the *popularity* level of an algorithm as follows. First, we compute min-max normalized popularity values of each item in the training set. Then, during evaluation, we compute the popularity level of an algorithm by determining the average popularity value of each item that appears in its top-n recommendation list. Higher values correspondingly mean that an algorithm has a tendency to recommend rather popular items.

3.2.5 Running times

Complex neural models can need substantial computational resources to be trained. Training a "model", i.e., calculating the statistics, for co-occurrence-based approaches like SR or AR can, in contrast, be done very efficiently. For nearest-neighbors-based approaches, actually no model is learned at all. Instead, some of our nearest-neighbors implementations need some time to create internal data structures that allow for efficient recommendation at prediction time. In the context of this paper, we will report running times for some selected datasets from both domains.

We executed all experiments on the same physical machine. The running times for the neural methods were determined using a GPU; the non-neural methods used a CPU. In theory, running times should be compared on the same hardware. Therefore, since the running times of the neural methods are much longer even when a GPU can be used, we can assume that the true difference in computational complexity is in fact even higher than we can see in our measurements.

3.2.6 Stability with respect to new data

In some application domains, e.g., news recommendation or e-commerce, new user–item interaction data can come in at a high rate. Since retraining the models to accommodate the new data can be costly, a desirable characteristic of an algorithm can be that the performance of the model does not degenerate too quickly before the retraining happens. To put it differently, it is desirable that the models do not overfit too much to the training data.

To investigate this particular form of model stability, we proceeded as follows. First, we trained a model on the training data T_0 of a given train-test split[7]. Then, we made measurements using two different protocols, which we term *retraining* and *no-retraining*, respectively.

[7] We also optimized the hyperparameters on a subset of T_0 that was used as a validation set. The hyperparameters were kept constant for the remaining measurements.

- In the *retraining* configuration, we first evaluated the model that was trained on T_0 using the data of the first day of the test set. Then, we added this first day of the test set to T_0 and retrained the model on this extended dataset, which we name T_1. Then, we continued with the evaluation with the data from the second day of the test data, using the model trained on T_1. This process of adding more data to the training set, retraining the full model, and evaluating on the next day of the test set was done for all days of the test set except the last one.
- In the *no-retraining* configuration, we also evaluated the performance day by day on the test data, but did not retrain the models, i.e., we used the model trained on T_0 for all days in the test data.

To enable a fair comparison in both configurations, we only considered items in the evaluation phase that appeared at least once in the original training data T_0.

Note that the absolute accuracy values for a given test day depend on the characteristics of the recorded data on that day. In some cases, the accuracy for the second test day can therefore even be higher than for the first test day, even if there was no retraining. An exact comparison of absolute values is therefore not too meaningful. However, we consider the *relative* accuracy drop when using the initial model T_0 for a number of consecutive days as an indicator of the generalizability or stability of the learned models, provided that the investigated algorithms start from a comparable accuracy level.

4 Results

In this section, we report the results of our offline evaluation. We will first focus on accuracy, then look at alternative quality measures, and finally discuss aspects of scalability and the stability of different models over time.

4.1 Accuracy results

4.1.1 E-commerce datasets

Table 6 shows the results for the e-commerce datasets, ordered by the values obtained for the MAP@20 metric. The non-neural models are marked with full circles, while the neural ones can be identified by empty ones. The highest value across all techniques is printed in bold; the highest value obtained by the other family of algorithms—neural or non-neural—is underlined. Stars indicate significant differences ($p < 0.05$) according to a Kruskal–Wallis test between all the models and a Wilcoxon signed-rank test between the best-performing techniques from each category. The results for the individual datasets can be summarized as follows.

Table 6 Results for the e-commerce datasets, ordered by MAP@20

Metrics	MAP@20	P@20	R@20	HR@20	MRR@20	COV@20	POP@20
RETAIL							
• STAN	**0.0285**	**0.0543**	**0.4748**	**0.5938**	0.3638*	0.5929	0.0518
• VSTAN	0.0284	0.0542	0.4741	0.5932	0.3636	0.5982	0.0488
• SKNN	0.0283	0.0532	0.4707	0.5788	0.3370	0.5709	0.0540
• V-SKNN	0.0278	0.0531	0.4632	0.5745	0.3395	0.5562	0.0598
○ GRU4REC	0.0272	0.0502	0.4559	0.5669	0.3237	0.7973*	**0.0347**
○ NARM	0.0270	0.0501	0.4526	0.5549	0.3196	0.6472	0.0569
○ CSRM	0.0252	0.0467	0.4246	0.5169	0.2955	0.6049	0.0496
○ SR-GNN	0.0241	0.0441	0.4125	0.4998	0.3252	0.5521	0.0743
○ STAMP	0.0223	0.0420	0.3806	0.4620	0.2527	0.4865	0.0677
• AR	0.0205	0.0387	0.3533	0.4367	0.2407	0.5444	0.0527
• SR	0.0194	0.0362	0.3359	0.4174	0.2453	0.5185	0.0424
○ NEXTITNET	0.0173	0.0320	0.3051	0.3779	0.2038	0.5737	0.0703
• CT	0.0162	0.0308	0.2902	0.3632	0.2305	0.4026	0.3740
DIGI							
• SKNN	**0.0255**	**0.0596**	0.3715	0.4748	0.1714	0.8701	0.1026
• VSTAN	0.0252	0.0588	**0.3723**	0.4803*	0.1837*	0.9384	0.0858
• STAN	0.0252	0.0589	0.3720	0.4800	0.1828	0.9161	0.0964
• V-SKNN	0.0249	0.0584	0.3668	0.4729	0.1784	0.9419	0.0840
○ GRU4REC	0.0247	0.0577	0.3617	0.4639	0.1644	**0.9498**	**0.0567**
○ CSRM	0.0227	0.0544	0.3335	0.4258	0.1421	0.7337	0.0833
○ NARM	0.0218	0.0528	0.3254	0.4188	0.1392	0.8696	0.0832
○ STAMP	0.0201	0.0489	0.3040	0.3917	0.1314	0.9188	0.0799
• AR	0.0189	0.0463	0.2872	0.3720	0.1280	0.8892	0.0863
○ SR-GNN	0.0186	0.0451	0.2840	0.3638	0.1564	0.8593	0.1092
• SR	0.0161	0.0401	0.2489	0.3277	0.1216	0.8736	0.0707
○ NEXTITNET	0.0149	0.0380	0.2416	0.2922	0.1424	0.7935	0.0947
• CT	0.0115	0.0294	0.1860	0.2494	0.1075	0.7554	0.4262
ZALANDO							
• VSTAN	**0.0168**	0.0777*	*0.2073	0.5362*	0.2488	0.5497	0.0664
• STAN	0.0167	0.0774	0.2062	0.5328	0.2468	0.4918	0.0734
• V-SKNN	0.0158	0.0740	0.1956	0.5162	0.2487	0.6246	0.0680
• SKNN	0.0157	0.0738	0.1891	0.4352	0.1724	0.3316	0.0843
○ SR-GNN	0.0146	0.0700	0.1823	0.4755	0.2804	0.3845	0.0865
○ NARM	0.0144	0.0692	0.1795	0.4598	0.2248	0.3695	0.0837
○ CSRM	0.0143	0.0695	0.1764	0.4500	0.2347	0.2767	0.0789
○ GRU4REC	0.0143	0.0666	0.1797	0.4925	**0.3069**	**0.6365**	0.0403*
• SR	0.0136	0.0638	0.1739	0.4824	0.3043	0.5849	0.0696
• AR	0.0133	0.0631	0.1690	0.4665	0.2579	0.4672	0.0886
• CT	0.0118	0.0564	0.1573	0.4561	0.2993	0.4653	0.2564
○ STAMP	0.0104	0.0515	0.1359	0.3687	0.2065	0.2234	0.0868

Table 6 (continued)

Metrics	MAP@20	P@20	R@20	HR@20	MRR@20	COV@20	POP@20
RSC15							
○ NARM	**0.0357**	**0.0735**	**0.5109**	0.6751	0.3047	0.6399	0.0638
○ SR-GNN	0.0351	0.0725	0.5060	0.6713	**0.3142**	0.5105	0.0720
○ NEXTITNET	0.0350	0.0722	0.5033	0.6691	0.3132	0.5295	0.0677
• VSTAN	0.0350	0.0718	0.5080	**0.6761**	0.2943	0.6762	0.0634
○ CSRM	0.0346	0.0714	0.4952	0.6566	0.2961	0.5929	0.0626
○ STAMP	0.0344	0.0713	0.4979	0.6654	0.3033	0.5803	0.0655
• STAN	0.0342	0.0701	0.4986	0.6656	0.2933	0.6828	0.0773
• V-SKNN	0.0341	0.0707	0.4937	0.6512	0.2872	0.6333	0.0777
○ GRU4REC	0.0334	0.0682	0.4837	0.6480	0.2826	**0.7482**	**0.0294**
• SR	0.0332	0.0684	0.4853	0.6506	0.3010	0.6674	0.0716
• AR	0.0325	0.0673	0.4760	0.6361	0.2894	0.6297	0.0926
• SKNN	0.0318	0.0657	0.4658	0.5996	0.2620	0.6099	0.0796
• CT	0.0316	0.0654	0.4710	0.6359	0.3072	0.6270	0.1446

The best results for each metric are highlighted in bold font. The next best results for algorithms from the other category (either neural or non-neural) are underlined. Non-neural methods are marked with full circles, and neural ones with empty ones

- On the RETAIL dataset, the nearest-neighbors methods consistently lead to the highest accuracy results on all the accuracy measures. Among the complex models, the best results were obtained by GRU4REC on all the measures except for MRR, where SR-GNN led to the best value. The results for NARM and GRU4REC are almost identical on most measures.
- The results for the DIGI dataset are comparable, with the neighborhood methods leading to the best accuracy results. GRU4REC is again the best method across the complex models on all the measures.
- For the ZALANDO dataset, the neighborhood methods dominate all accuracy measures, except for the MRR. Here, GRU4REC is minimally better than the simple SR method. Among the complex models, GRU4REC achieves the best HR value, and the recent SR-GNN method is the best one on the other accuracy measures.
- Only for the RSC15 dataset, we can observe that a neural method (NARM) is able to slightly outperform our best simple baseline VSTAN in terms of MAP, Precision, and Recall. Interestingly, however, NARM is one of the earlier neural methods in this comparison. The best Hit Rate is achieved by VSTAN, and the best MRR by SR-GNN. The differences between the best neural and non-neural methods are often tiny, in most cases around or less than 1 %.

Looking at the results across the different datasets, we can make the following additional observations.

- Across all e-commerce datasets, the VSTAN method proposed in this paper is, for most measures, the best neighborhood-based method. This suggests that it is reasonable to include it as a baseline in future performance comparisons.
- The ranking of the *neural* methods varies largely across the datasets and does not follow the order in which the methods were proposed. Like for the non-neural methods, the specific ranking therefore seems to be strongly depending on the dataset characteristics. This makes it particularly difficult to judge the progress that is made when only one or two datasets are used for the evaluation.
- The results for the RSC15 dataset are generally different from the other results. Specifically, we found that some neural methods (NARM, SR-GNN, NEXTITNET) are competitive and sometimes even slightly outperform our baselines. Moreover, STAMP and NEXTITNET are usually not among the top performers, but work well for this dataset. Unlike for other e-commerce datasets, CT works particularly well for this dataset in terms of the MRR. Given these observations, it seems that the RSC15 dataset has some unique characteristics that are different from the other e-commerce datasets. Therefore, it seems advisable to consider multiple datasets with different characteristics in future evaluations.
- We did not include measurements for NEXTITNET, one of the most recent methods, for some of the datasets (ZALANDO, 30MU, 8TRACKS, NOWP), because our machines ran out of memory (> 32 GB). These datasets were either comparably large or had longer sessions on average.

4.1.2 Music domain

In Table 7, we present the results for the music datasets. In general, the observations are in line with what we observed for the e-commerce domain regarding the competitiveness of the simple methods.

- Across all datasets excluding the 8TRACKS dataset, the nearest-neighbors methods are consistently favorable in terms of Precision, Recall, MAP, and the Hit Rate, and the CT method leads to the best MRR. Moreover, the simple SR technique often leads to very good MRR values.
- For the 8TRACKS dataset, the best Recall, MAP, and the Hit Rate values are again achieved by neighborhood methods. The best Precision and the MRR values are, however, achieved by a neural method (NARM).
- Again, no consistent ranking of the algorithms can be found across the datasets. In particular, the neural approaches take largely varying positions in the rankings across the datasets. Generally, NARM seems to be a technique which performs consistently well on most datasets and measures.

Table 7 Results for the music domain datasets

Metrics	MAP@20	P@20	R@20	HR@20	MRR@20	COV@20	POP@20
NOWP							
• V-SKNN	0.0193*	**0.0664**	0.1828*	0.2534	0.0810	0.4661	0.0582
• SKNN	0.0186	0.0655	0.1809	0.2450	0.0687	0.3150	0.0619
• STAN	0.0175	0.0585	0.1696	0.2414	0.0871	0.5128	0.0473
• VSTAN	0.0174	0.0609	0.1795	0.2597*	0.0853	0.4299	0.0505
• AR	0.0166	0.0564	0.1544	0.2076	0.0710	0.4531	0.0511
• SR	0.0133	0.0466	0.1366	0.2002	0.1052	0.4661	0.0383
○ SR-GNN	0.0125	0.0490	0.1400	0.2113	0.0935	0.3265	0.0576
○ NARM	0.0118	0.0463	0.1274	0.1849	0.0894	0.4715	0.0488
○ GRU4REC	0.0116	0.0449	0.1361	0.2261	0.1076	0.5795*	**0.0286**
○ STAMP	0.0111	0.0456	0.1244	0.1954	0.0921	0.2148	0.0714
○ CSRM	0.0095	0.0388	0.1065	0.1508	0.0594	0.2445	0.0494
• CT	0.0065	0.0287	0.0893	0.1679	**0.1094**	0.2714	0.2984
30MU							
• V-SKNN	0.0309*	0.1090*	0.2347*	0.3830	0.1162	0.3667	0.0485
• VSTAN	0.0296	0.1003	0.2306	0.3904*	0.1564	0.4333	0.0293
• SKNN	0.0290	0.1073	0.2217	0.3443	0.0898	0.1913	0.0574
• STAN	0.0278	0.0949	0.2227	0.3830	0.1533	0.4315	0.0347
• AR	0.0254	0.0886	0.1930	0.3088	0.0960	0.3524	0.0393
• SR	0.0240	0.0816	0.1937	0.3327	0.2410	0.4131	0.0317
○ NARM	0.0155	0.0675	0.1486	0.2956	0.1945	0.3858	0.0425
○ GRU4REC	0.0150	0.0617	0.1529	0.3273	0.2369	**0.4881**	**0.0255**
○ CSRM	0.0118	0.0536	0.1236	0.2652	0.1503	0.2290	0.0390
○ SR-GNN	0.0108	0.0482	0.1151	0.2883	0.1894	0.3965	0.0412
○ STAMP	0.0093	0.0411	0.0875	0.1539	0.0819	0.0852	0.0491
• CT	0.0058	0.0308	0.0885	0.2882	0.2502*	0.1932	0.4255
AOTM							
• SKNN	0.0037*	0.0139*	0.0390*	0.0417*	0.0054	0.2937	0.1467
• V-SKNN	0.0032	0.0116	0.0312	0.0352	0.0057	0.5886	0.1199
• STAN	0.0031	0.0126	0.0357	0.0402	0.0054	0.2979	0.1667
• VSTAN	0.0024	0.0083	0.0231	0.0271	0.0060	0.6907*	**0.0566**
• AR	0.0018	0.0076	0.0200	0.0233	0.0059	0.5532	0.1049
• SR	0.0010	0.0047	0.0134	0.0186	0.0074	0.5669	0.0711
○ NARM	0.0009	0.0050	0.0146	0.0202	0.0088	0.4816	0.1119
• CT	0.0006	0.0043	0.0126	0.0191	0.0111*	0.3357	0.4680
○ SR-GNN	0.0006	0.0032	0.0096	0.0148	0.0082	0.4283	0.0812
○ CSRM	0.0005	0.0040	0.0109	0.0100	0.0021	0.0056	0.6478
○ NEXTITNET	0.0004	0.0024	0.0071	0.0139	0.0065	0.4851	0.0960
○ STAMP	0.0003	0.0020	0.0063	0.0128	0.0088	0.5168	0.0872
○ GRU4REC	0.0003	0.0020	0.0063	0.0130	0.0074	0.5898	0.0594
8TRACKS							
• SKNN	0.0024*	0.0129	0.0343	0.0377*	0.0054	0.2352	0.1622

Table 7 (continued)

Metrics	MAP@20	P@20	R@20	HR@20	MRR@20	COV@20	POP@20
• STAN	0.0022	0.0119	0.0313	0.0357	0.0052	0.2971	0.1382
• V-SKNN	0.0021	0.0110	0.0276	0.0312	0.0056	0.4572	0.1064
• VSTAN	0.0018	0.0086	0.0227	0.0265	0.0056	0.5192*	0.0757
○ NARM	0.0018	**0.0131**	0.0311	0.0345	0.0083*	0.0788	0.1589
○ SR-GNN	0.0017	0.0123	0.0301	0.0330	0.0077	0.0211	0.1833
• AR	0.0016	0.0088	0.0219	0.0255	0.0071	0.4529	0.0912
○ STAMP	0.0015	0.0114	0.0256	0.0272	0.0061	0.0405	0.1374
• SR	0.0012	0.0067	0.0166	0.0201	0.0071	0.4897	0.0657*
○ CSRM	0.0011	0.0087	0.0189	0.0204	0.0048	0.0417	0.1587
○ GRU4REC	0.0007	0.0060	0.0132	0.0161	0.0051	0.2839	0.0825
• CT	0.0007	0.0054	0.0127	0.0170	0.0071	0.2732	0.2685

The best results for each metric are highlighted in bold font. The next best results for algorithms from the other category (either neural or non-neural) are underlined. Again, non-neural methods are marked with a full circle, and neural ones with an empty one

4.2 Coverage and popularity

Tables 6 and 7 also contain information about the popularity bias of the individual algorithms and coverage information. Remember that we described in Sect. 3.2 how the numbers were calculated. From the results, we can identify the following trends regarding individual algorithms and the different algorithm families.

4.2.1 Popularity bias

- The CT method is very different from all other methods in terms of its *popularity bias*, which is much higher than for any other method.
- The GRU4REC method, on the other hand, is the method that almost consistently recommends the most unpopular (or: novel) items to the users.
- The neighborhood-based methods are often in the middle. There are, however, also neural methods, in particular SR-GNN, which seem to have a similar or sometimes even stronger popularity bias than the nearest-neighbors approaches. The assumption that nearest-neighbors methods are in general more focusing on popular items than neural methods can therefore not be confirmed through our experiments.

4.2.2 Coverage

- In terms of *coverage*, we found that GRU4REC often leads to the highest values.

Table 8 Running times for selected algorithms on two datasets

Algorithm	Training (min)			Predicting (ms)		
	RSC15	ZALANDO	8TRACKS	RSC15	ZALANDO	8TRACKS
○ GRU4REC2	43.14	39.65	12.54	7.72	25.97	278.23
○ STAMP	32.51	133.17	112.84	14.94	55.45	423.94
○ NARM	225.82	797.72	623.76	7.83	25.00	211.35
○ SR-GNN	827.37	1527.17	482.46	27.67	120.15	797.97
○ CSRM	156.89	203.15	96.83	24.98	66.93	250.23
○ NEXTITNET	1577.40	–	–	8.98	–	–
• AR	0.40	1.00	0.34	4.66	12.00	105.43
• SR	0.41	0.53	0.25	4.66	11.77	101.98
• SKNN	0.18	0.13	0.05	37.82	27.77	291.26
• V-SKNN	0.19	0.13	0.05	18.75	30.56	278.51
• STAN	0.18	0.20	0.05	36.78	33.26	317.23
• VSTAN	0.18	0.13	0.06	21.33	55.58	288.40
• CT	11.00	15.60	4.35	73.34	484.87	1452.71

- The coverage of the neighborhood-based methods varies quite a lot, depending on the specific algorithm variant. In some configurations, their coverage is almost as high as for GRU4REC, while in others the coverage can be low.
- The coverage values of the other neural methods also do not show a clear ranking, and they are often in the range of the neighborhood-based methods and sometimes even very low.

4.3 Scalability

We present selected results regarding the running times of the algorithms for two e-commerce datasets and one music dataset in Table 8. The reported times were measured for training and predicting for one data split. The numbers reported for predicting correspond to the average time needed to generate a recommendation for a session beginning in the test set. For this measurement, we used a workstation computer with an Intel Core i7-4790k processor and an Nvidia GeForce GTX 1080 Ti graphics card (Cuda 10.1/CuDNN 7.5).

The results generally show that the computational complexity of neural methods is, as expected, much higher than for the non-neural approaches. In some cases, researchers therefore only use a smaller fraction of the original datasets, e.g., or of the RSC15 dataset. Several algorithms—both neural ones and the CT method—exhibit major scalability issues when the number of recommendable items increases. Furthermore, for the NEXTITNET method, we found that it is consuming a lot of memory for some datasets, as mentioned above, leading to out-of-memory errors.

In some cases, like for CT or SR-GNN, not only the training time increases, but also the prediction times. In particular, the prediction times can, however, be subject to strict time constraints in production settings. The prediction times for the

Table 9 Relative accuracy decrease (in percent) for the evaluated algorithms on two datasets, ordered by HR@20 for the DIGI dataset

Metrics	DIGI		NOWP	
	HR@20 (%)	MRR@20 (%)	HR@20 (%)	MRR@20 (%)
• SKNN	– 1.90	– 0.17	– 23.42	– **14.29**
• V-SKNN	– 2.28	– 0.64	– 27.20	– 14.36
• VSTAN	– 2.53	– 0.64	– 28.53	– 28.22
• STAN	– 2.97	– 0.29	– 27.21	– 27.92
• AR	– 4.83	– 5.33	– 29.76	– 33.94
• SR	– 6.22	– 6.14	– 32.38	– 70.05
• CT	– 7.98	– 6.94	– 50.49	– 85.97
○ NARM	– **1.84**	**0.30**	– 35.10	– 70.28
○ GRU4REC	– 2.79	– 1.84	– 46.03	– 74.11
○ NEXTITNET	– 3.75	– 4.69	–	–
○ SR-GNN	– 3.76	– 2.14	– 46.05	– 75.74
○ CSRM	– 4.20	– 4.68	– **17.84**	– 41.27
○ STAMP	– 7.80	– 7.28	– 46.48	– 45.78

The best results for each metric are highlighted in bold font. The next best results from the other category (neural or non-neural) are underlined

nearest-neighbors methods are often slightly higher than those measured for methods like GRU4REC, but usually lie within the time constraints of real-time recommendation (e.g., requiring about 30ms for one prediction for the ZALANDO dataset).

Since datasets in real-world environments can be even larger, this leaves us with questions regarding the practicability of some of the approaches. In general, even in case where a complex neural method would slightly outperform one of the more simple ones in an offline evaluation, it remains open if it is worth the effort to put such complex methods into production. For the ZALANDO dataset, for example, the best neural method (SR-GNN) needs several orders of magnitude[8] more time to train than the best non-neural method VSTAN, which also only needs half the time for recommending.

A final interesting observation is that there can be a large spread, i.e., in the range of an order of magnitude and more, between the running times of the neural methods. For example, the methods that use convolution (NEXTITNET) or graph structures (SR-GNN) often need much more time than other techniques like GRU4REC or NARM. A detailed theoretical analysis of the computational complexity of the different algorithms and their underlying architectures is, however, beyond the scope of our present work, which compares the effectiveness and efficiency in an empirical way.

[8] The training time for SR-GNN is 10,000 times higher than for VSTAN.

4.4 Stability with respect to new data

We report the stability results for the examined neural and non-neural algorithms in Table 9. Given the computational requirements for this simulation-based analysis, which requires multiple full retraining phases, we selected one of the smaller datasets for each domain in this analysis, DIGI and NOWP.

We used two months of training data and 10 days of test data for both datasets, DIGI and NOWP. The reported values show how much the accuracy results of each algorithm degrade (in percent), averaged across the test days when there is no daily re-training.

We can see from the results that the drop in accuracy without retraining can vary a lot across datasets (domains). For the DIGI dataset, the decrease in performance ranges between 0 and 10 percent across the different algorithms and performance measures. The NOWP dataset from the music domain seems to be more short-lived, with more recent trends that have to be considered. Here, the decrease in performance ranges from about 15 to 50 percent in terms of HR and from about 15 to 85 percent in terms of MRR.[9]

Looking at the detailed results, we see that in both families of algorithms, i.e., neural and non-neural ones, some algorithms are much more stable than others when new data are added to a given dataset. For the non-neural approaches, we see that nearest-neighbor approaches are generally better than the other baselines techniques based on association rules or context trees.

Among the neural methods, NARM is the most stable one on the DIGI dataset, but often falls behind the other deep learning methods on the NOWP dataset.[10] On this latter dataset, the CSRM method leads to the most stable results. In general, however, no clear pattern across the datasets can be found regarding the performance of the neural methods when new data come in and no retraining is done.

Overall, given that the computational costs of training complex models can be high, it can be advisable to look at the stability of algorithms with respect to new data when choosing a method for production. According to our analysis, there can be strong differences across the algorithms. Furthermore, the nearest-neighbors methods appear to be quite stable in this comparison.

5 Observations from a user study

Offline evaluations, while predominant in the literature, can have certain limitations, in particular when it comes to the question of how the quality of the provided recommendations is *perceived* by users. We therefore conducted a controlled experiment, in which we compared different algorithmic approaches for session-based

[9] Generally, comparing the numbers across the datasets is not meaningful due to their different characteristics.

[10] The experiments for NEXTITNET could not be completed on this dataset because the method's resource requirements exceeded our computing capacities.

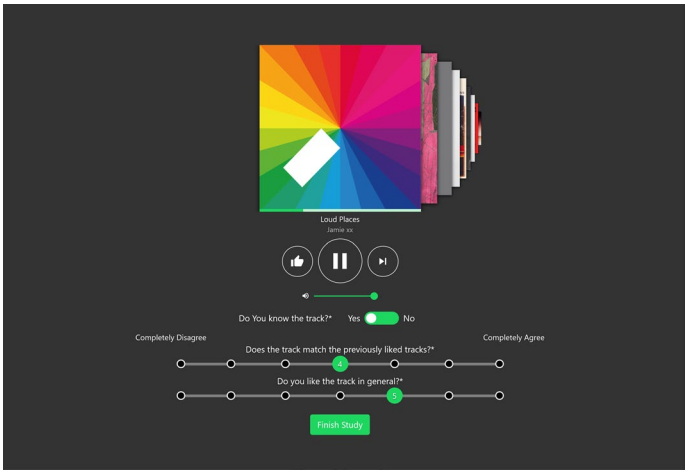

Fig. 1 Track rating interface of the application

recommendation in the context of an online radio station. In the following sections, we report the main insights of this experiment. While the study did not include all algorithms from our offline analysis, we consider it helpful to obtain a more comprehensive picture regarding performance of session-based recommenders. More details about the study can be found in Ludewig and Jannach (2019).

5.1 Research questions and study setup

5.1.1 Research questions

Our offline analysis indicated that simple methods are often more competitive than the more complex ones. Our main research question therefore was how the recommendations generated by such simple methods are perceived by its users in different dimensions, in particular compared to recommendations by a complex method. Furthermore, we were interested how users perceive the recommendations of a commercial music streaming service, in our case Spotify, in the same situation.

5.1.2 Study setup

An online music listening application in the form of an "automated radio station" was developed for the purpose of the study. Similar to existing commercial services, users of the application could select a track they like (called a "seed track"), based on which the application created a playlist of subsequent tracks.

The users could then listen to an excerpt of the next track and were asked to provide feedback about it as shown in Fig. 1. Specifically, they were asked if (1) if they already knew the track, (2) to what extent the track matched the previously played track, and (3) to what extent they liked the track (independent of the playlist). In addition, the participants could press a "like" button before skipping

Table 10 Questions about users' quality perceptions

	Question
Suitability of Tracks and Perceived Personalization	
Q1	I liked the automatically generated radio station
Q2	The radio suited my general taste in music
Q3	The tracks on the radio musically matched the track I selected in the beginning
Q4	The radio was tailored to my preferences the more positive feedback I gave
Perceived Diversity, Serendipity, and Familiarity	
Q5	The radio was diversified in a good way
Q6	The tracks on the radio surprised me
Q7	I discovered some unknown tracks that I liked in the process
Attention Check	
Q8	I am participating in this study with care so I change this slider to two
Intention to Reuse and to Recommend to Others	
Q9	I would listen to the same radio station based on that track again
Q10	I would use this system again, e.g., with a different first song
Q11	I would recommend this radio station to a friend
Q12	I would recommend this system to a friend

to the next track. In case of such a *like* action, the list of upcoming tracks was updated. Users were visually hinted that such an update takes place. This update of the playlist was performed for all methods including Spotify, i.e., in that case we re-fetched a new playlist through Spotify API after each *like* statement.

Once the participants had listened to and rated at least 15 tracks, they were forwarded to a post-task questionnaire. In this questionnaire, we asked the participants 11 questions about how they perceived the service; see also Pu et al. (2011). Specifically, the participants were asked to provide answers to the questions using seven-point Likert scale items, ranging from "completely disagree" to "completely agree". The questions, which include a twelfth question as an attention check, are listed in Table 10. In the table, we group the questions according to the different quality dimensions they refer to, inspired by Pu et al. (2011). This grouping was not visible for participants in the online study.

The study itself was based on a between-subjects design, where the treatments for each user group correspond to different algorithmic approaches to generate the recommendations. We included algorithms from different families in our study.

- AR: Association rules of length two, as described in Sect. 2. We included this method as a simple baseline.
- CAGH: Another relatively simple baseline, which recommends the greatest hits of artists similar to those liked in the current session. This music-specific method is often competitive in offline evaluations as well; see Bonnin and Jannach (2014).

- SKNN: The basic nearest-neighbors method described above. We took the simple variant as a representative for the family of such approaches, as it performed particularly well in the ACM RecSys 2018 challenge (Ludewig et al. 2018).
- GRU4REC: The RNN-based approach discussed above, used as a representative for neural methods. NARM would have been a stable alternative, but did not scale well for the used dataset.
- SPOTIFY: Recommendations in this treatment group were retrieved in real time from Spotify API.

We optimized and trained all models on the Million Playlist Dataset Million Playlist Dataset (MPD)[11] provided by Spotify. We then recruited study participants using Amazon's Mechanical Turk crowdsourcing platform. After excluding participants who did not pass the attention checks, we ended up with *N=250* participants, i.e., 50 for each treatment group, for which we were confident that they provided reliable feedback.

Most of the recruited participants (almost 80%) were US-based. The most typical age range was between 25 and 34, with more than 50% of the participants falling into this category. On average, the participants considered themselves to be music enthusiasts, with an average response of 5.75 (on the seven-point scale) to a corresponding survey question. As usual, the participants received a compensation for their efforts through the crowdsourcing platform.

5.2 User study outcomes

The main observations can be summarized as follows.

5.2.1 Feedback the listening experience

Looking at the feedback that was observed during the listening session, we observed the following.

- *Number of Likes* There were significant differences regarding the number of *likes* we observed across the treatment groups. Recommendations by the simple AR method received the highest number of likes (6.48), followed by SKNN (5.63), CAGH (5.38), GRU4REC (5.36), and SPOTIFY (4.48).
- *Popularity of Tracks* We found a clear correlation (r=0.89) between the general popularity of a track in the MPD dataset and the number of likes in the study. The AR and CAGH methods recommended, on average, the most popular tracks. The recommendations by SPOTIFY and GRU4REC were more oriented toward tracks with lower popularity.
- *Track Familiarity* There were also clear differences in terms of how many of the recommended tracks were already known by the users. The CAGH (10.83 %) and

[11] https://recsys-challenge.spotify.com/.

Table 11 Descriptive statistics and outcomes of the statistical significance tests for the post-task questionnaire

	S-KNN			CAGH			GRU4REC		
	Mean ± Std	Md	Mo	Mean ± Std	Md	Mo	Mean ± Std	Md	Mo
Q1	5.980 ±1.145^{gas}	6	7	5.796 ±1.369^{gas}	6	6	5.224 ±1.504	5	5
Q2	5.673 ±1.231^{a}	6	6	5.735 ±1.483^{a}	6	6	5.490 ±1.502^{a}	6	7
Q3	5.673 ±1.281^{gas}	6	7	5.286 ±1.646^{a}	6	6	4.673 ±2.125^{a}	6	6
Q4	5.633 ±1.202^{a}	6	6	5.510 ±1.697^{a}	6	6	5.224 ±1.531	5	6
Q5	5.204 ±1.399	5	5	5.224 ±1.545	5	5	4.653 ±1.786	5	4
Q6	3.878 ±1.589	4	3	3.755 ±1.774	4	5	4.000 ±1.720	4	3
Q7	4.061 ±2.155	4	7	3.939 ±2.193	4	1	4.041 ±1.848	5	5
Q9	5.653 ±1.422^{as}	6	6	5.347 ±1.809^{a}	6	7	5.082 ±1.730	5	7
Q10	6.204 ±1.000^{ga}	6	7	6.000 ±1.258^{ga}	6	7	5.388 ±1.681	6	7
Q11	5.449 ±1.487^{a}	6	7	5.408 ±1.790^{a}	6	7	4.959 ±1.744	5	6
Q12	5.816 ±1.269^{ga}	6	6	5.735 ±1.455^{ga}	6	7	5.122 ±1.654	5	5
	AR			SPOTIFY					
Q1	4.776 ±1.598	5	3	5.367 ±1.453^{a}	6	6			
Q2	4.735 ±1.765	5	3	5.306 ±1.475	5	5			
Q3	4.245 ±1.843	4	2	4.980 ±1.548	5	5			
Q4	5.082 ±1.205	5	4	5.592 ±1.273^{a}	6	7			
Q5	4.633 ±1.603	5	3	4.959 ±1.707	5	5			
Q6	4.204 ±1.384	5	5	4.286 ±1.620	4	3			
Q7	4.286 ±2.189	6	6	5.224 ±1.476^{kcga}	5	5			
Q9	4.755 ±1.362	4	4	5.224 ±1.476^{a}	5	6			
Q10	5.245 ±1.465	5	4	6.041 ±1.274^{ga}	6	7			
Q11	4.490 ±1.647	4	3	5.265 ±1.524^{a}	5	7			
Q12	4.796 ±1.720	5	3	5.551 ±1.473^{a}	6	7			

We report mean, standard deviation (Mean ± Std), median (Md), and mode (Mo) for the post-task questionnaire. We furthermore applied a Kruskal–Wallis test and subsequently a Mann–Whitney U test when appropriate. Significant pairwise differences between the algorithms according to the Mann–Whitney U test ($p < 0.05$) are noted with *k* for SKNN, *c* for CAGH, *g* for GRU4REC, *a* for AR, and *s* for Spotify

SKNN (10.13 %) methods recommended the largest number of known tracks. The AR method, even though it recommended very popular tracks, led to much more unfamiliar recommendations (8.61 %). GRU4REC was somewhere in the middle (9.30 %), and SPOTIFY recommended the most novel tracks to users (7.00 %).

- *Suitability of Track Continuations* The continuations created by SKNN and CAGH were perceived to be the most suitable ones. The differences between SKNN and AR, GRU4REC, and SPOTIFY were significant. The recommendations made by the AR method were considered to match the playlist the least. This is not too surprising because the AR method only considers the very last played track for the recommendation of subsequent tracks.
- *Individual Track Ratings* The differences regarding the individual ratings for each track rating are generally small and not significant. Interestingly, the playlist-

independent ratings for tracks recommended by the AR method were the lowest ones, even though these recommendations received the highest number of likes. An analysis of the rating distribution shows that the AR method often produces very bad recommendations, with a *mode* value of 1 on the 1–7 rating scale.

5.2.2 Post-task questionnaire

The detailed statistics of the answers to the post-task questionnaire are shown in Table 11. The analysis of the data revealed the following aspects:

- Q1: The radio station based on SKNN was significantly more liked than the stations that used GRU4REC, AR, and SPOTIFY.
- Q2: All radio stations matched the users general taste quite well, with median values between 5 and 6 on a seven-point scale. Only the station based on the AR method received a significantly lower rating than the others.
- Q3: The SKNN method was found to perform significantly better than AR and GRU4REC with respect to identifying tracks that musically match the seed track.
- Q4: The adaptation of the playlist based on the like statements was considered good for all radio stations. Again, the feedback for the AR method was significantly lower than for the other methods.
- Q5 and Q6: No significant differences were found regarding the surprise level of the different recommendation strategies.
- Q7: Regarding the capability of recommending unknown tracks that the users liked, the recommendations by SPOTIFY were perceived to be much better than for the other methods, with significant differences compared to all other methods.
- Q9 to Q12: The best performing methods in terms of the intention to reuse and the intention to recommend the radio station to others were SKNN, CAGH, and SPOTIFY. GRU4REC and AR were slightly worse, sometimes with differences that were statistically significant.

Overall, the study confirmed that methods like SKNN do not only perform well in an offline evaluation, but are also able, according to our study, to generate recommendations that are well perceived in different dimensions by the users. The study also revealed a number of additional insights.

First, we found that optimizing for *like* statements can be misleading. The AR method received the highest number of likes, but was consistently worse than other techniques in almost all other dimensions. Apparently, this was caused by the fact that the AR method made a number of bad recommendations; see also Patrick et al. (2013) for an analysis of the effects on bad recommendations in the music domain.

Second, it turned out that *discovery support* seems to be an important factor in this particular application domain. While the recommendations of SPOTIFY were slightly less appreciated than those by SKNN, we found no difference in terms of the user's intention to reuse the system or to recommend it to friends. We hypothesize that the better discovery support of SPOTIFY's recommendations was an important

factor for this phenomenon. This observation points to the importance of considering multiple potential quality factors when comparing systems.

6 Research limitations

Our work does not come without limitations, both regarding the offline evaluations and the user study.

6.1 Potential data biases

One general problem of offline evaluations based on historical data is that we often know very little about the circumstances and environment in which the data were collected. For the e-commerce datasets, for example, what we see as interactions in the log can be at least partially the result of the recommender system that was in place during the time of data collection, or it can simply be the result of how certain items or categories were promoted in the online shop. For the music datasets, and in particular for data obtained from Last.fm (30MU), it might be that the logs to some extent reflect what the Last.fm radio station functionality was playing automatically given a seed track. Well-performing algorithms, i.e., those that predict the next items in the log accurately, might therefore be the ones that are able to "reconstruct" the logic of an existing recommender in some ways. The results of such a biased offline evaluation might therefore not fully reflect the effectiveness of a system.

Over the years, a number of approaches were proposed to deal with such problems, e.g., by using evaluation measures that take biases into account or by trying to "de-bias" the datasets (Steck 2010; Carraro and Bridge 2019). In particular, in the context of reinforcement learning and bandit-based approaches, a number of research proposals were made for unbiased offline evaluation protocols to obtain more realistic performance estimates from log data; see Li et al. (2011) for an early work. The analysis or consideration of such biases was, however, not in the scope of the work, which aimed at the comparison of different existing algorithms using standard evaluation protocols. While the outcomes of these analyses (and of the original works) maybe therefore suffer from potential biases, the conducted user study provided us with strong indications that the generated recommendations were also liked by users.

6.2 Empirical nature of the work

Generally, our work—like the papers that proposed the analyzed neural models—is mainly an empirical one in terms of the research approach. Algorithmic papers that propose new models in many cases do not start with a theoretical model, but probably more often with an intuition of what kind of signals there could be in the data. In case performance increases are found when using model that is designed to capture these signals, a common approach in that context is to use ablation studies to determine, again empirically, to what extent certain parts of the network architecture

contribute to the overall performance. In the context of our comparative work, in contrast, it would be interesting to understand why even computationally very complex models are *not* consistently performing better than the more simple models. Possible explanations could be that some underlying assumptions do not hold for the majority of the datasets. In some domains, the sequential ordering of the events, as captured by RNNs, might for example not be very important. Another problem could lie in a certain tendency of overfitting of the complex models, even when the hyperparameters are optimized on a held-out validation set. A detailed analytical investigation of the potential reasons why each of the six complex models in our comparison does *not* perform consistently better than the more simple ones, however, lies beyond the scope of this present work and is left for future work.

6.3 User study limitations

Finally, the user study discussed in Sect. 5—like most studies of that type—has certain limitations as well. Typical issues that apply also to our study are questions related the representativeness of the user population. Furthermore, while we developed a realistic and fully functional online radio station, the setting remains artificial and users were paid for their participation. The attention checks and the statistics of how users interacted with the system, however, make us confident that the majority of the participants completed the task with care and that the results are reliable. Another potential limitation of our study design is that we used one single item for each of the investigated quality dimensions in the post-task questionnaire. Since we mainly used established questions from the literature, e.g., from Pu et al. (2011), the associated risks are low.

7 Conclusions and ways forward

Our work reveals that despite a continuous stream of papers that propose new neural approaches for session-based recommendation, the progress in the field seems still limited. According to our evaluations, today's deep learning techniques are in many cases not outperforming much simpler heuristic methods. Overall, this indicates that there still is a huge potential for more effective neural recommendation methods in the future in this area.

In a related analysis of deep learning techniques for recommender systems (Ferrari Dacrema et al. 2019a, b), the authors found that different factors contribute to what they call *phantom progress*. One first problem is related to the reproducibility of the reported results. They found that in less than a third of the investigated papers, the code was made available to other researchers. The problem also exists to some extent for session-based recommendation approaches. To further increase the level of reproducibility, we share our evaluation framework publicly, so that other researchers can easily benchmark their own methods with a comprehensive set of neural and non-neural approaches on different datasets.

Through sharing our evaluation framework, we hope to also address other methodological and procedural issues mentioned in Ferrari Dacrema et al. 2019b) that can make the comparison of algorithms unreliable or inconclusive. Regarding methodological issues, we for example found works that determined the optimal number of training epochs on the test set and furthermore determined the best Hit Rate and MRR values across different optimization epochs. Regarding procedural issues, we found that while researchers seemingly rely on the same datasets as previous works, they sometimes apply different data pre-processing strategies. Furthermore, the choice of the baselines can make the results inconclusive. Most investigated works do not consider the SKNN method and its variants as a baseline. Some works only compare variants of one method and include a non-neural, but not necessarily strong other baseline. In many cases, little is also said about the optimization of the hyperparameters of the baselines. The SESSION-REC framework used in our evaluation should help to avoid these problems, as it contains all the code for data pre-processing, evaluation, and hyperparameter optimization. Such frameworks are generally important to ensure replicability and reproducibility of research results (Çoba and Zanker 2017). Furthermore, sharing the framework allows other researchers to inspect the exact details of how the algorithms are implemented and evaluated, which is important as no *de facto* standards exist in the literature, which can sometimes lead to inconclusive and inconsistent results (Said and Bellogín 2014).

Moreover, also from a methodological perspective, our analyses indicated that optimizing solely for accuracy can be insufficient also for session-based recommendation scenarios. Depending on the application domain, other quality factors such as coverage, diversity, or novelty should be considered besides efficiency, because they can be crucial for the adoption and success of the recommendation service. Given the insights from our controlled experiment, we furthermore argue that more user studies and field tests are necessary to understand the characteristics of successful recommendations in a given application domain.

Looking at future directions, in particular methods that leverage side information about users and items seem to represent a promising way forward; see de Souza Pereira Moreira et al. (2018, 2019), Huang et al. (2018), Hidasi et al. (2016b). In Hidasi et al. (2016b), the authors for example use a parallel RNN architecture to incorporate image and text information in the session modeling process. In de Souza Pereira Moreira et al. (2018, 2019), both item information and user context information are combined in a neural architecture for news recommendation. Huang et al. (2018), finally, combine RNNs with Key-Value memory networks to build a hybrid system that integrates information about item attributes in the sequential recommendation process.

A main challenge when trying to analyze and compare such methods under identical conditions, as was the goal of our present work, is that these works rely on largely different and often specific datasets, e.g., containing image information, or are optimized for a specific problem setting, e.g., cold-start situations in the news domain. An important direction for future work therefore lies in analyzing to what extent the benefits of such hybrid architectures generalize beyond individual application domains.

Acknowledgements We thank Liliana Ardissono for her valuable feedback on the paper.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Adomavicius, G., Kwon, Y.O.: Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans. Knowl. Data Eng. **24**(5), 896–911 (2012)

Armstrong, T.G., Moffat, A., Webber, W., Zobel, J.: 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, pp. 601–610 (2009)

Bonnin, G., Jannach, D.: Automated generation of music playlists: survey and experiments. ACM Comput. Surv. **47**(2), 26:1–26:35 (2014)

Carraro, D., Bridge, D.: Debiased offline evaluation of recommender systems: a weighted-sampling approach (extended abstract). In: Proceedings of the ACM RecSys 2019 Workshop on Reinforcement and Robust Estimators for Recommendation (REVEAL '19) (2019)

Chau, P.Y.K., Ho, S.Y., Ho, K.K.W., Yao, Y.: Examining the effects of malfunctioning personalized services on online users' distrust and behaviors. Decis. Support Syst. **56**, 180–191 (2013)

Chen, S., Moore, J.L., Turnbull, D., Joachims, T.: Playlist prediction via metric embedding. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pp. 714–722 (2012)

Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder–decoder approaches. CoRR, abs/1409.1259 (2014)

Çoba, L., Zanker, M.: Replication and reproduction in recommender systems research—evidence from a case-study with the rrecsys library. In: 30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE '17, pp. 305–314 (2017)

de Souza Pereira Moreira, G. Ferreira, Felipe, M., da Cunha, A.M.: News session-based recommendations using deep neural networks. In: Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems, DLRS '18, pp. 15–23 (2018)

de Souza Pereira Moreira, G., Jannach, D. da Cunha, A.M.: Contextual hybrid session-based news recommendation with recurrent neural networks. IEEE Access **7** (2019)

Ferrari Dacrema, M., Boglio, S., Cremonesi, P., Jannach, D.: A troubling analysis of reproducibility and progress in recommender systems research, CoRR, abs/2004.00646 (2019a)

Ferrari Dacrema, M., Cremonesi, P., Jannach, D.: Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, pp. 101–109 (2019b)

Garcin, F., Dimitrakakis, C., Faltings, B.: Personalized news recommendation with context trees. In: Proceedings of the 7th ACM Conference on Recommender Systems, RecSys '13, pp. 105–112 (2013)

Garcin, F., Zhou, K., Faltings, B., Schickel, V.: Personalized news recommendation based on collaborative filtering. In: Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology, WI-IAT '12, pp. 437–441 (2012)

Garg, D., Gupta, P., Malhotra, P., Vig, L., Shroff, G.: Sequence and time aware neighborhood for session-based recommendations: Stan. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '19, pp. 1069–1072 (2019)

Hariri, N., Mobasher, B., Burke, R.: Context-aware music recommendation based on latent topic sequential patterns. In: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, pp. 131–131 (2012)

Hariri, N., Mobasher, B., Burke, R.: Adapting to user preference changes in interactive recommendation. In: Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI '15, pp. 4268–4274. AAAI (2015)

He, R., McAuley, J.: Fusing similarity models with markov chains for sparse sequential recommendation. CoRR, abs/1609.09152 (2016)

Hidasi, B., Karatzoglou, A.: Recurrent neural networks with top-k gains for session-based recommendations. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM '18, pp. 843–852 (2018)

Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: Proceedings International Conference on Learning Representations, ICLR '16 (2016)

Hidasi, B., Quadrana, M., Karatzoglou, A., Tikk, D.: Parallel recurrent neural network architectures for feature-rich session-based recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, pp. 241–248 (2016)

Huang, J., Zhao, W.X., Dou, H., Wen, J.-R., Chang, E.Y.: Improving sequential recommendation with knowledge-enhanced memory networks. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '18, pp. 505–514 (2018)

Jannach, D., Lerche, L., Jugovac, M.: Adaptation and evaluation of recommendations for short-term shopping goals. In: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, pp. 211–218 (2015)

Jannach, D., Ludewig, M.: When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the 11th ACM Conference on Recommender Systems, RecSys '17, pp. 306–310 (2017)

Jannach, D., Ludewig, M., Lerche, L.: Session-based item recommendation in e-commerce: on short-term intents, reminders, trends, and discounts. User-Model. User-Adapted Interact. **27**(3–5), 351–392 (2017)

Jannach, D., Zanker, M.: Collaborative filtering: matrix completion and session-based recommendation tasks. In: Collaborative Recommendations: Algorithms, Practical Challenges and Applications, pp. 1–38 (2019)

Kabbur, S., Ning, X., Karypis, G.: FISM: factored item similarity models for top-n recommender systems. In: KDD '13, pp. 659–667 (2013)

Lerche, L., Jannach, D., Ludewig, M.: On the value of reminders within e-commerce recommendations. In: UMAP '16, pp. 27–35 (2016)

Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J.: Neural attentive session-based recommendation. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM '17, pp. 1419–1428 (2017)

Li, L., Chu, W., Langford, J., Wang, X.: Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM '11, pp. 297–306 (2011)

Liu, Q., Zeng, Y., Mokhosi, R., Zhang, H.: STAMP: short-term attention/memory priority model for session-based recommendation. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, pp. 1831–1839 (2018)

Ludewig, M., Jannach, D.: Evaluation of session-based recommendation algorithms. User-Model. User-Adapted Interact. **28**(4–5), 331–390 (2018)

Ludewig, M., Jannach, D.: User-centric evaluation of session-based recommendations for an automated radio station. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, pp. 516–520 (2019)

Ludewig, M., Kamehkhosh, I., Landia, N., Jannach, D.: Effective nearest-neighbor music recommendations. In: Proceedings of the ACM Recommender Systems Challenge 2018, RecSys Challenge '18, pp. 3:1–3:6 (2018)

Ludewig, M., Mauro, N., Latifi, S., Jannach, D.: Performance comparison of neural and non-neural approaches to session-based recommendation. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, pp. 462–466 (2019)

Makridakis, S., Spiliotis, E., Assimakopoulos, V.: Statistical and machine learning forecasting methods: concerns and ways forward. PloS one **13**(3), (2018)

Mi, F., Faltings, B.: Context tree for adaptive session-based recommendation. CoRR, abs/1806.03733 (2018)

Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Using sequential and non-sequential patterns in predictive web usage mining tasks. In: Proceedings of IEEE International Conference on Data Mining, ICDM '02, pp. 669–672 (2002)

Norris, J.R.: Markov Chains. Cambridge University Press, Cambridge (1997)

Pu, P., Chen, L., Hu, R.: A user-centric evaluation framework for recommender systems. In: Proceedings of the 5th ACM Conference on Recommender Systems, RecSys '11, pp. 157–164 (2011)

Quadrana, M., Cremonesi, P., Jannach, D.: Sequence-aware recommender systems. ACM Comput. Surv. **54**, 1–36 (2018)

Ragno, R., Burges, C.J.C., Herley, C.: Inferring similarity between music objects with application to playlist generation. In: Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval, MIR '05, pp. 73–80 (2005)

Ren, P., Chen, Z., Li, J., Ren, Z., Ma, J., de Rijke, M.: Repeatnet: a repeat aware neural recommendation machine for session-based recommendation. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI '19, pp. 4806–4813 (2019)

Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: UAI '09, pp. 452–461 (2009)

Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: WWW '10, pp. 811–820 (2010)

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, CSCW '94, pp. 175–186 (1994)

Romov, P., Sokolov, E.: RecSys Challenge 2015: ensemble learning with categorical features. In: Proceedings of the 2015 International ACM Recommender Systems Challenge, RecSys '15 Challenge, pp. 1:1–1:4 (2015)

Said, A., Bellogín, A.: Comparative recommender system evaluation: benchmarking recommendation frameworks. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pp. 129–136 (2014)

Shani, G., Gunawardana, A.: Evaluating recommendation systems. In: Recommender Systems Handbook, pp. 257–297 (2011)

Shani, G., Heckerman, D., Brafman, R.I.: An MDP-based recommender system. J. Mach. Learn. Res. **6**, 1265–1295 (2005)

Steck, H.: Training and testing of recommender systems on data missing not at random. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10, pp. 713–722 (2010)

Tan, Y.K., Xu, X., Liu, Y.: Improved recurrent neural networks for session-based recommendations. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS '16, pp. 17–22 (2016)

Tavakol, M., Brefeld, U.: Factored MDPs for detecting topics of user sessions. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pp. 33–40 (2014)

Wang, M., Ren, P., Mei, L., Chen, Z., Ma, J., de Rijke, M.: A collaborative session-based recommendation approach with parallel memory modules. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '19, pp. 345–354 (2019)

Wang, S., Cao, L., Wang, Y.: A survey on session-based recommender systems. CoRR, abs/1902.04864 (2019)

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI, pp. 346–353 (2019)

Yang, W., Lu, K., Yang, P., Lin, J.: 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, pp. 1129–1132 (2019)

Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J.M., He, X.: A simple convolutional generative network for next item recommendation. In: Proceedings of the 12th ACM International Conference on Web Search and Data Mining, WSDM '19, pp. 582–590 (2019)

Malte Ludewig is a Ph.D. candidate in Computer Science at TU Dortmund, Germany, from where he also received his M.Sc. degree. His research interests lie in the field of recommender systems—with a focus on session-based recommendations—and personalization in e-commerce environments in general.

Noemi Mauro is a Postdoctoral Researcher at the Computer Science Department of the University of Torino where she recently obtained a Ph.D. in Computer Science with Honors. Her research interests concern user modeling, recommender systems, information filtering and information visualization. She recently won the best paper award at UMAP 2020 and the outstanding program committee member award at HT 2020. She is a program committee member of the top conferences in her research areas and reviewer for several related journals.

Sara Latifi is a Ph.D. candidate in Computer Science at the University of Klagenfurt, Austria, and she received her M.Sc. degree from the University of Isfahan, Iran. Her research interests are data science, artificial intelligence, machine learning and its applications, including data mining and recommender systems.

Dietmar Jannach is a Professor of Computer Science at the University of Klagenfurt, Austria, and head of the department's information systems research group. Dr. Jannach has worked on different areas of artificial intelligence, including recommender systems, model-based diagnosis, and knowledge-based systems. He is the leading author of a textbook on recommender systems and has authored more than hundred research papers, focusing on the application of artificial intelligence technology to practical problems.

Affiliations

Malte Ludewig[1] · Noemi Mauro[2] · Sara Latifi[3] · Dietmar Jannach[3]

✉ Malte Ludewig
malte.ludewig@tu-dortmund.de

Noemi Mauro
noemi.mauro@unito.it

Sara Latifi
sara.latifi@aau.at

Dietmar Jannach
dietmar.jannach@aau.at

[1] TU Dortmund, Dortmund, Germany

[2] University of Torino, Turin, Italy

[3] University of Klagenfurt, Klagenfurt, Austria