Exploring Session-Based Recommender Systems with Reminders

Douglas Rorie Tanno, Vinicius Sylvestre Simm, Gabriel Libardi Lulu and Marcos Aurelio Domingues
Department of Informatics
State University of Maringá
Maringá, Paraná, Brazil
{pg404806, pg404811, ra134728, madomingues}@uem.br

***Abstract*—Online services, such as e-commerce platforms, streaming services, and social networks, provide users with vast amounts of information, often leading to information overload and making it challenging to discover items of interest. Recommender systems have been developed to mitigate this issue by personalizing user experiences and facilitating the discovery of relevant content. These systems leverage user preference data, interaction history, and demographic information to generate accurate recommendations. In this work, we investigate the effectiveness of Session-Based Recommender Systems (SBRS) with *reminders*, which incorporate previously viewed items to improve recommendation precision and relevance. We conducted experiments in the legal, music, employment, and e-commerce domains to evaluate the generalizability of *reminder* strategies beyond the e-commerce setting, where they have been most widely adopted. Results show that *reminders* consistently improve performance in employment and legal scenarios, corroborate the existing literature with strong gains in e-commerce, and reveal a more modest impact in the music domain, suggesting potential for further research in this area.**

***Index Terms*—Recommender Systems, Reminders, Session-Based Recommender Systems.**

I. Introduction

Online services, such as e-commerce platforms, streaming services, and social networks, provide vast amounts of information, leading to information overload for users. This challenge hinders timely access to relevant content. To address this, recommender systems have become essential for enhancing user experience across various online services [1], [2].

The primary function of a recommender system is to predict products that potential consumers may be interested in purchasing. To provide a personalized user experience, enhancing their journey and facilitating the discovery of new content or products of interest, recommendations are generated by considering various sources. These include users' self-declared preferences, purchase history, data from other customers with similar tastes or demographic profiles, as well as browsing information. Furthermore, recommender systems can leverage a variety of data types, such as text, images, and multimedia content [1]–[4].

This work was supported by the National Council for Scientific and Technological Development (CNPq), Brazil, under the Master's scholarship (Process No. 170391/2023-0); and financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

A widely adopted strategy involves recommending items that closely align with products the user has previously viewed. This approach aims to further personalize recommendations, guiding users toward products that align with their historical browsing and viewing interests. Not only does this enhance the user experience by offering more relevant suggestions, but it also encourages potential transactions by capturing users' attention with items they have already shown interest in [5], [6].

Session-Based Recommender Systems (SBRS) are designed to capture users' short-term preferences by analyzing the sequence of interactions (e.g., clicks) within a session that has a distinct start and end. A session is defined as a sequence of related user actions occurring within a specific, often brief, time frame, which typically reflects a coherent, goal-directed activity. Unlike traditional recommender systems, which rely on the user's entire historical interaction data, SBRS focuses solely on the interactions within a current session to predict the next item (or all subsequent items) the user may be interested in. This approach is particularly effective in contexts where user preferences can change rapidly, such as during last-minute decisions or in dynamic media consumption environments [5].

Beyond traditional recommendations, the use of *reminders* in recommender systems has proven effective. *Reminders* reuse items viewed by the user during the current session, recalling products or content of recent interest. Online platforms frequently incorporate *reminders* into their recommendation lists to assist decision-making, reduce cognitive effort, and increase user trust in recommendations. Additionally, *reminders* help build future shopping lists and act as a temporary solution in cold-start scenarios, where little user information is available. They can be particularly useful in domains such as music and video, encouraging the revisit of previously consumed items that may have been forgotten [7], [8].

Furthermore, *reminders* may contribute to user retention and engagement by facilitating the resumption of purchase decisions or content consumption, strengthening continuous interaction with the platform [7].

Thus, our work explores recommender systems incorporating *reminders* across different application domains. This work is based on the hypothesis that the use of *reminders* can improve the relevance of recommendations, while also increasing user satisfaction. To this end, we conducted exper-

iments with SBRS employing *reminders* in the legal, music, and employment domains, as well as in e-commerce platforms, where this strategy has been more extensively explored. By investigating these diverse contexts, our goal was to assess whether the benefits of incorporating *reminders* extend beyond e-commerce and can also enhance recommendation outcomes in other domains. The performance of SBRS with *reminders* was compared against baseline models without this strategy.

The remainder of this paper is structured as follows. Section II reviews related work. Section III details the methodology, covering datasets, data preprocessing, training and test set splitting, SBRS models, and evaluation setup. Section IV presents and discusses the experimental results. Finally, Section V concludes the paper with final remarks and future research directions.

II. Related Work

Research exploring the use of *reminders* SBRS is still scarce, with most studies focusing on the e-commerce domain.

Lerche et al. [9] compared various recommendation strategies using e-commerce datasets such as Zalando, China-Gadgets, and TMall. Their *reminder* strategy *IRec* was used as a baseline and compared with strategies like *IInt*, *ISim*, and *SSim*. *IRec* recommends items based on the timestamp of the user's last interaction (view, purchase, or cart event). *IInt* relies on the frequency of item views in the user's history. *ISim* finds items aligned with the current purchase goal using cosine similarity between item descriptions. *SSim* searches past sessions sharing the same purchase goal, also using cosine similarity.

Their evaluation considered, for each target purchase, the p sessions preceding a time interval of d days, with p fixed at 6 sessions and d varying from 0 to 3 days. Additionally, to improve recommendation quality, they introduced a *Feature Filter* (FF), which dynamically excludes *reminders* from previously purchased categories within a session.

For $d = 0$, *IRec* performed well on Hit Rate (HR), as users often view items shortly before buying. *IInt* and *ISim* had lower performance. The *SSim* strategy excelled for Zalando's high-engagement users, indicating users often abandon sessions and return later. For $d = 3$, HR and MRR values dropped overall, but combining *ISim* with FF yielded the best results by filtering *reminders* from already purchased categories. For China-Gadgets, *IRec* performed well due to the temporal order of interactions.

Outside e-commerce, Domingues et al. [8] explored *reminders* in the legal domain using the JusBrasilRec dataset. They employed a hybrid *reminder* strategy combining *IRec* and *SSim* with SBRS models. Their results showed the hybrid did not consistently improve performance and sometimes worsened it, highlighting the importance of parameter tuning for *reminders*.

The limited research on *reminders* in SBRS suggests that this is an emerging area. Most existing works are restricted to single-domain e-commerce settings, with little investigation into their applicability in other contexts. Our work aims to address this gap by evaluating *reminder* strategies across multiple application domains.

III. Methodology

This section describes the methodology employed to conduct the experiments in this work. The main steps are summarized in Figure 1 and are further detailed throughout the following subsections.

A. Datasets

For the experiments, we used four public datasets adopted in SBRS research:

- **JusBrasilRec:** Contains 30 days of user interactions on the JusBrasil legal platform (Feb 23 to Mar 24, 2021). Each record includes session/user IDs, user type (logged-in or not), document ID and type, and timestamp [10]. We split it into 5 slices of 5 days each.
- **Music4All:** Includes 15,602 users and 109,269 songs, with listening histories, metadata (artist, album, release year), lyrics, 30-second audio clips, user and genre tags, and Spotify audio features such as danceability, energy, and popularity [11]. Divided into 5 slices of 21 days.
- **Xing:** From ACM RecSys Challenge 2016, with about 1.5 million users, 12 million interactions (clicks, bookmarks, replies), and 327,000 job postings. Job metadata includes title, description, career level, industry, and location; user profiles have anonymized experience and region info [12]. Split into 5 slices of 16 days.
- **RetailRocket:** Collected from a real-world e-commerce platform, this dataset includes user behavior data and item attributes such as price, category, vendor, and product type. It contains 212,182 actions in 59,962 sessions across 31,968 items [13]. We divided it into 5 slices of 27 days.

B. Data Preprocessing

To ensure data quality, all datasets were preprocessed prior to the experiments.

Sessions containing only a single interaction were discarded, as they provide limited information for training recommendation models. Sessions with more than 50 interactions were also excluded, as they may reflect atypical user behavior that does not align with typical platform usage. Additionally, duplicated sessions were removed to eliminate redundancies and preserve the uniqueness of user interaction sequences.

Table I provides a comprehensive overview of the datasets following the preprocessing stage, reporting the total interactions, distinct items, sessions, session length statistics, and data collection timespan.

C. Training and Test Set Splitting

Each dataset was split into temporal slices, with the window size d (in days) chosen to fit the overall timespan of the dataset. For each slice, a user-level split was applied, where the most recent session of each user was assigned to the test set and the earlier sessions were used for training, as shown in Figure 2. This approach maintains the chronological order

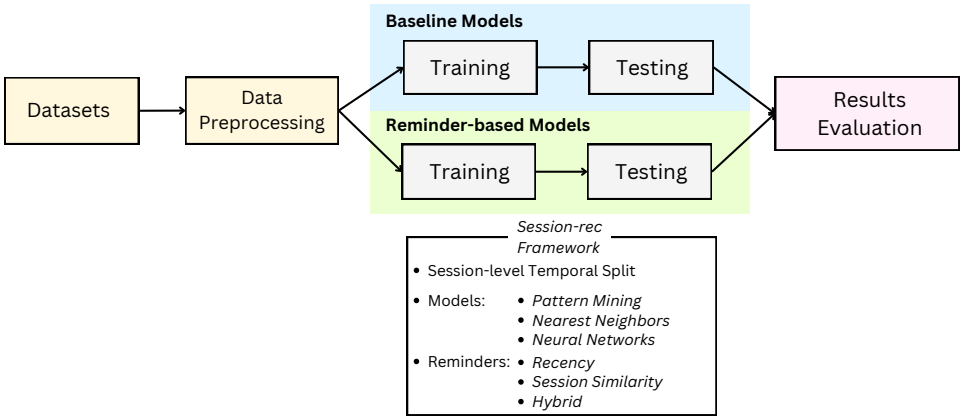

Fig. 1. Overview of the main steps involved in the development of this work.

TABLE I
PREPROCESSED DATASETS STATISTICS

Statistics	JusBrasilRec	Music4All	Xing	RetailRocket
Number of Interactions	3,637,949	3,998,499	255,291	187,341
Number of Items	100,052	48,957	19,898	11,980
Number of Sessions	905,824	522,387	52,850	42,732
Items per Session (min.)	2	2	2	2
Items per Session (mean)	4.02	7.65	4.83	4.38
Items per Session (max.)	50	50	50	47
Timespan in Days	29	109	79	134

of interactions, reflecting realistic settings for recommender systems.

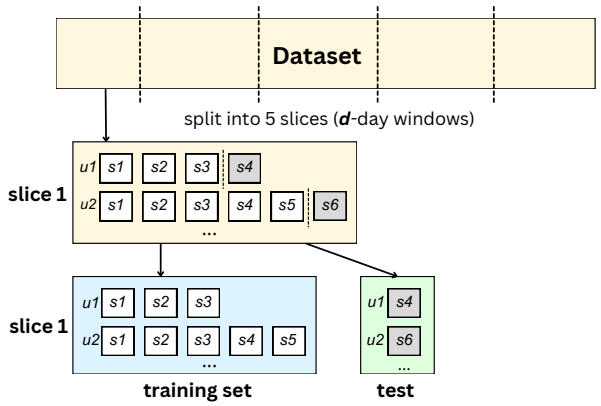

Fig. 2. Illustration of the data splitting setup into 5 slices (d-day windows). For each user u_n within a slice, the last session s_n^{last} is allocated to the test set, while the previous sessions $s_n^1, \ldots, s_n^{k-1}$ are used for training.

D. SBRS Models

The experiments were conducted using the *session-rec* recommendation framework[1], which provides a variety of models employed in SBRS with and without *reminders*—the latter referred to as baseline models.

The recommender models used in our work can be grouped into three main categories based on their underlying principles: *Pattern Mining*, *Nearest Neighbors*, and *Neural Networks* [7], [10].

Pattern Mining-Based Models explore item co-occurrence patterns to identify frequent associations in user interaction data. These models recommend items that often appear together, leveraging recurrent behaviors to enhance the relevance of recommendations [7], [14]. The following model was included in this category:

- `sr`: Based on the *Sequential Rules*, this model builds upon the classical *Association Rules* (`ar`) approach, which identifies frequent item co-occurrences in the form of rules $i \rightarrow j$, indicating that item j is likely to follow item i. The `sr` model extends `ar` by incorporating the sequential order of items within sessions. It applies a decay function that penalizes larger distances between items, thus giving more weight to short-term sequential dependencies observed in training sessions [14].

Nearest Neighbors-Based Models, despite their simplicity, have demonstrated competitive performance in various session-based recommendation tasks [14]–[16]. Three variants were selected:

- `vsknn`: The *Vector Multiplication Session-Based KNN* is a variant of `knn` that emphasizes recent items when calculating session similarity. The current session is encoded as a real-valued vector, where only the last item receives weight 1, and the others are weighted by a linear decay function based on their position. Similarity is computed using the dot product, which gives more influence to recent interactions [17].
- `stan`: The *Sequence and Time-aware Neighborhood* model incorporates three factors: (1) item position in the current session, (2) recency of past sessions, and (3) position of recommendable items in similar sessions. Decay functions are applied to assign higher weights to more recent interactions [18].
- `vstan`: This model integrates the mechanisms of both `stan` and `vsknn`. In addition to considering session recency and item position, it introduces a sequence-aware scoring scheme, and applies an IDF-based weighting

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

strategy to penalize overly frequent items, where IDF (Inverse Document Frequency) is commonly used to reduce the influence of popular items in information retrieval [19].

Neural Network-Based Models represent the most recent and complex approaches for session-based recommendation. Two models were selected for our work:

- `gru4rec`: The first neural model proposed for session-based recommendation, employing Gated Recurrent Units (GRUs) to mitigate the vanishing gradient problem in sequential learning tasks. The model predicts the next likely item based on the sequential dynamics of the session [20].
- `narm`: The *Neural Attentive Recommendation Machine* extends `gru4rec` by introducing a hybrid encoder equipped with an attention mechanism. This mechanism captures both sequential signals and item-level similarity to the last interaction. Candidate item scores are computed using a bilinear matching scheme over the unified session representation [21].

All models were initially evaluated using their default hyperparameter settings. Subsequently, a comprehensive set of experiments was conducted, where each model was tested with a wide range of parameter configurations. The best-performing setup for each model and dataset was selected from the validation set. In some cases, default values were retained when no significant gains were observed with alternative configurations.

One of the key parameters adjusted in the framework was `remind_mode`, which determines the position of *reminder* items in the recommendation list. This parameter can be set to *top* or *end* (default). When set to *top*, *reminder* items are placed at the top of the recommendation list, giving them higher priority. In contrast, when set to *end*, *reminder* items are added to the bottom of the list, after the main recommendations. The `remind_mode` was set to *top* for JusBrasilRec, Xing, and RetailRocket, as it performed better in these datasets. For Music4All, however, `remind_mode` was set to *end*, where it produced the best results.

Another important parameter adjusted was `remind_sessions_num`, which controls how many of the user's most recent sessions are used to select *reminder* items. While the default value is 6, for the Xing dataset we observed that using only the most recent session (`remind_sessions_num` = 1) yielded better results, likely due to its users' shorter histories.

Finally, for `gru4rec`, the number of training epochs was adapted per dataset based on validation performance and convergence speed: 10 epochs for JusBrasilRec, 50 for Xing, and 30 for Music4All and RetailRocket. The batch size was also varied, with a smaller value of 6 used for Xing and 32 for the remaining datasets.

E. Reminder Strategies

In our work, we have adopted the *reminder* strategies available in the *session-rec* framework: **recency**, **session similarity**, and **hybrid**. These *reminders* follow the concept of Adaptive Reminders [9]. It is important to note that the session similarity and hybrid strategies are only available for Nearest Neighbors-Based Models within the framework, as they rely on computing similarity scores between sessions based on `knn`.

To formalize these techniques, we define the following notation. Let H_u represent the recent interaction history of a user u, extracted from their previous sessions. A set of candidate items H_I^u is sampled from H_u, which consists of sessions composed of interaction tuples H_V^u in the form of ⟨item, action, timestamp⟩. The type of action may vary according to the domain—such as clicks, viewing an item, adding a song to a playlist, or applying for a job—and provides contextual information for identifying meaningful reminders [9].

The **recency** *reminder* strategy is based on the *IRec* (Item Recency) concept, which prioritizes items that have been recently interacted with by the user. The underlying idea is that items from recent interactions are more likely to be relevant for future recommendations, as they reflect the user's current interests and preferences. This approach assumes that user preferences evolve rapidly, and by prioritizing recent interactions, the recommendations stay aligned with the user's up-to-date preferences [9].

To quantify the recency of interactions, the *timestamps* of interactions are used to calculate a *ranking* score score_{ui}^{IRec} for a user u and an item i. This score reflects the temporal relevance of an item based on the user's interaction history. Specifically, the score is determined by identifying the most recent interaction for the item, as defined in Equation 1:

$$\text{score}_{ui}^{IRec} = \max_{t \in H_u^V(i)} (time(t)) \tag{1}$$

where $H_u^V(i)$ is the set of interaction tuples in H_u^V related to item i, and $time()$ is a function that returns the timestamp of a past interaction tuple t.

The **session similarity** *reminder* strategy is based on the *SSim* (Session Similarity) technique, which aims to identify past sessions of a user that exhibit similar intent to the current session. This approach assumes that if previous sessions contain items semantically close to those in the current session, then other items from those sessions may also be of interest [9].

To compute session similarity, the cosine similarity between items in the current session C_u and items in a past session V_s is averaged, as shown in Equation 2:

$$\text{sim}_{session}(u, s) = \frac{\sum_{i \in V_s} \sum_{j \in C_u} \text{sim}_{cos}(i, j)}{|V_s| \cdot |C_u|} \tag{2}$$

where s is a past session of user u, V_s is the set of items in session s, and C_u is the set of items in the current session.

The top k most similar sessions, denoted $\text{top}_k^S(u)$, are selected. The union of items from these sessions, $items(\text{top}_k^S(u))$, forms the candidate set. To rank the candidate items, the *IInt* (Interaction Intensity) score is used,

which reflects how frequently an item was viewed in the user's interaction history:

$$\text{score}_{ui}^{SSim} = \text{score}_{ui}^{IInt} \cdot \mathbf{1}_{items(\text{top}_k^S(u))}(i) \tag{3}$$

The *IInt* score for a user u and an item i is defined as the number of past interactions with item i, as given in Equation 4:

$$\text{score}_{ui}^{IInt} = |H_u^V(i)| \tag{4}$$

This scoring assumes that a higher frequency of interaction with an item indicates a stronger user interest. In cases where multiple items receive the same *IInt* score, the *IRec* score can be used to break ties based on the most recent interaction timestamps [9].

The **hybrid** *reminder* strategy combines the strengths of both *recency*-based and *session similarity*-based approaches by integrating their respective *reminder* scores into a unified recommendation score. This fusion aims to capture not only the immediate context of the user's current session but also historical behavioral patterns across previous sessions [9].

Internally, the hybrid strategy computes separate *reminder* scores using the *IRec* and *SSim* techniques. Each score is then integrated with the base recommendation score (from the main model) using a weighted sum, as shown in Equation 5:

$$\text{score}_{ui}^{Hybrid} = \sum_{x \in \{base, IRec, SSim\}} w_x \cdot \text{score}_{ui}^{x} \tag{5}$$

where each component $x \in \{base, IRec, SSim\}$ is associated with a weight w_x that controls its contribution to the final score. To ensure comparability across components, all scores score_{ui}^{x} are normalized by their respective maximum values prior to aggregation.

This strategy provides a flexible mechanism to adapt the recommendation list to both short-term user interests (via recency) and longer-term preferences (via session similarity), which can be particularly effective in dynamic environments with recurrent users [9].

F. Evaluation Setup

SBRS aim to generate a ranked list of items based on a specific session, and their effectiveness is evaluated by how well they predict items within that session. To simulate the user's journey, we withhold all items and reveal them iteratively. Specifically, the task focuses on predicting the next item based on the first n interactions in the session [17].

Two evaluation scenarios were adopted: **Next Item** and **Rest of Session**.

In the **Next Item** scenario, the model receives the first n interactions of a session and must predict the $(n+1)$-th item. The value of n is incrementally increased to reflect different points within the session, simulating how the model performs at various stages of a user's navigation. For each prediction, we compute Hit Rate and Mean Reciprocal Rank (MRR). Hit Rate@k indicates how often the correct next item appears among the top-k recommended items, while MRR@k considers the position of the correct item in the ranking, rewarding higher placements [17].

In the **Rest of Session** scenario, the task is to predict all remaining items in the session after the first n interactions. Here, we evaluate the ranked output using Precision, Recall, NDCG (Normalized Discounted Cumulative Gain), and MAP (Mean Average Precision). Precision@k captures the proportion of recommended items that are actually relevant, while Recall@k measures the fraction of relevant items that were retrieved. NDCG@k assigns higher weight to relevant items that appear earlier in the list, and MAP@k aggregates precision scores across all correct predictions, offering a comprehensive view of ranking quality [17].

Additionally, we consider **Coverage** and **Popularity Bias**. Coverage measures the fraction of unique items from the catalog appearing in recommendation lists, while Popularity Bias checks whether higher accuracy is linked to recommending popular items. Popularity is measured as the average normalized frequency which each item appears across training sessions, normalized between 0 and 1 [7].

All evaluations are conducted at a cutoff of $k = 10$, following common practice in recommender systems to limit cognitive load and reflect realistic interface constraints [17].

IV. Results and Discussion

This section presents the experimental results obtained on the four datasets used in our work.

Throughout this section, we refer to the *reminders* versions of the models using a suffix that indicates the strategy employed: `-recency` for models that incorporate the recency strategy, `-session` for those based on session similarity, and `-hybrid` for models that combine the hybrid strategy.

Tables II, III, IV and V summarize the results for the JusBrasilRec, Xing, Music4All, and RetailRocket datasets, respectively. The best-performing values for both the Next Item and Rest of Session evaluation scenarios, as well as for the metrics Coverage and Popularity Bias, are highlighted in bold. In case of ties, the value with the smallest standard deviation was chosen. If both the mean and the standard deviation are tied, no value was highlighted. All results are averaged across multiple temporal slices and computed at a cutoff of $n = 10$ (i.e., the top 10 recommendations).

Across all datasets, the use of *reminders* frequently achieved competitive or superior performance compared to their respective baseline models.

On the JusBrasilRec dataset, models using *reminder* strategies showed consistent improvements, particularly in Coverage. Notably, `vstan-session` and `narm-recency` achieved the highest Coverage scores (0.952 and 0.965, respectively), indicating their ability to expose users to a broader range of relevant items. Additionally, *Reminder*-based versions of `uvsknn` also stood out, with `uvsknn-recency` and `uvsknn-session` outperforming their baseline model in Hit Rate, Precision, Recall, and MAP.

However, gains were not uniform across all metrics. The baseline model `vstan` still achieved the highest MRR (0.611),

TABLE II
RESULTS FOR JUSBRASILREC DATASET

Models	Hit Rate	MRR	Precision	Recall	NDCG	MAP	Coverage	Popularity
stan	0.747 ± 0.002	**0.578 ± 0.002**	0.105 ± 0.000	0.481 ± 0.007	**0.460 ± 0.005**	0.054 ± 0.001	0.794 ± 0.009	0.083 ± 0.008
stan-recency	0.754 ± 0.005	0.349 ± 0.129	**0.106 ± 0.001**	0.488 ± 0.009	0.357 ± 0.055	0.054 ± 0.001	0.885 ± 0.047	**0.086 ± 0.008**
stan-session	**0.755 ± 0.003**	0.277 ± 0.008	**0.106 ± 0.001**	**0.489 ± 0.007**	0.325 ± 0.006	**0.055 ± 0.001**	**0.940 ± 0.023**	0.085 ± 0.008
stan-hybrid	0.747 ± 0.005	0.517 ± 0.139	0.105 ± 0.001	0.482 ± 0.009	0.432 ± 0.061	0.054 ± 0.001	0.827 ± 0.068	0.083 ± 0.008
vstan	0.743 ± 0.003	**0.611 ± 0.005**	0.097 ± 0.000	**0.494 ± 0.004**	**0.479 ± 0.003**	0.054 ± 0.000	0.448 ± 0.121	0.018 ± 0.001
vstan-recency	**0.754 ± 0.001**	0.292 ± 0.004	**0.106 ± 0.000**	0.488 ± 0.007	0.332 ± 0.004	**0.055 ± 0.001**	0.908 ± 0.006	0.073 ± 0.006
vstan-session	0.751 ± 0.002	0.274 ± 0.004	0.105 ± 0.001	0.486 ± 0.007	0.323 ± 0.004	0.054 ± 0.001	**0.952 ± 0.003**	**0.075 ± 0.006**
vstan-hybrid	0.741 ± 0.002	0.589 ± 0.002	0.103 ± 0.000	0.475 ± 0.007	0.458 ± 0.005	0.053 ± 0.001	0.804 ± 0.009	0.068 ± 0.006
vsknn	0.745 ± 0.002	0.563 ± 0.002	0.105 ± 0.000	0.481 ± 0.007	0.456 ± 0.005	0.054 ± 0.001	0.796 ± 0.010	0.087 ± 0.009
vsknn-recency	**0.757 ± 0.002**	0.289 ± 0.004	**0.108 ± 0.001**	**0.493 ± 0.007**	0.332 ± 0.004	**0.055 ± 0.001**	**0.908 ± 0.006**	**0.088 ± 0.008**
vsknn-session	0.755 ± 0.002	0.541 ± 0.002	0.107 ± 0.000	0.489 ± 0.007	**0.458 ± 0.006**	**0.055 ± 0.001**	0.803 ± 0.009	0.086 ± 0.009
vsknn-hybrid	0.745 ± 0.002	**0.564 ± 0.002**	0.105 ± 0.000	0.481 ± 0.007	0.456 ± 0.005	0.054 ± 0.001	0.796 ± 0.010	0.087 ± 0.009
gru4rec	0.680 ± 0.002	**0.523 ± 0.004**	0.093 ± 0.000	0.434 ± 0.006	**0.414 ± 0.005**	0.048 ± 0.001	0.920 ± 0.005	0.042 ± 0.004
gru4rec-recency	**0.708 ± 0.001**	0.278 ± 0.003	**0.099 ± 0.000**	**0.458 ± 0.006**	0.314 ± 0.003	**0.051 ± 0.001**	**0.955 ± 0.003**	**0.055 ± 0.004**
narm	0.718 ± 0.003	**0.541 ± 0.006**	0.100 ± 0.000	0.461 ± 0.006	**0.437 ± 0.004**	0.051 ± 0.001	0.937 ± 0.006	0.081 ± 0.008
narm-recency	**0.735 ± 0.003**	0.284 ± 0.003	**0.104 ± 0.000**	**0.477 ± 0.006**	0.323 ± 0.003	**0.053 ± 0.001**	**0.965 ± 0.004**	**0.082 ± 0.008**
sr	0.665 ± 0.003	**0.523 ± 0.004**	0.090 ± 0.000	0.425 ± 0.007	**0.409 ± 0.006**	0.047 ± 0.001	0.772 ± 0.010	0.072 ± 0.008
sr-recency	**0.699 ± 0.002**	0.275 ± 0.003	**0.097 ± 0.000**	**0.453 ± 0.006**	0.311 ± 0.003	**0.050 ± 0.001**	**0.901 ± 0.006**	**0.077 ± 0.007**

TABLE III
RESULTS FOR XING DATASET

Models	Hit Rate	MRR	Precision	Recall	NDCG	MAP	Coverage	Popularity
stan	0.311 ± 0.019	**0.184 ± 0.012**	0.053 ± 0.003	0.218 ± 0.017	0.176 ± 0.013	0.024 ± 0.002	0.985 ± 0.004	0.097 ± 0.068
stan-recency	0.329 ± 0.026	0.140 ± 0.009	0.059 ± 0.004	0.239 ± 0.023	**0.191 ± 0.012**	**0.027 ± 0.003**	**0.990 ± 0.004**	0.098 ± 0.068
stan-session	**0.332 ± 0.024**	0.142 ± 0.008	**0.059 ± 0.003**	**0.240 ± 0.023**	**0.191 ± 0.012**	**0.027 ± 0.003**	**0.990 ± 0.004**	**0.099 ± 0.068**
stan-hybrid	0.311 ± 0.019	**0.184 ± 0.012**	0.053 ± 0.003	0.218 ± 0.017	0.176 ± 0.013	0.024 ± 0.002	0.985 ± 0.004	0.097 ± 0.068
vstan	0.310 ± 0.017	**0.193 ± 0.011**	0.053 ± 0.003	0.216 ± 0.016	0.177 ± 0.012	0.024 ± 0.002	0.991 ± 0.002	0.088 ± 0.065
vstan-recency	0.345 ± 0.020	0.128 ± 0.009	0.065 ± 0.003	**0.258 ± 0.017**	0.189 ± 0.012	**0.029 ± 0.002**	**0.995 ± 0.002**	**0.093 ± 0.064**
vstan-session	**0.349 ± 0.018**	0.133 ± 0.008	**0.065 ± 0.002**	**0.258 ± 0.017**	**0.191 ± 0.011**	**0.029 ± 0.002**	0.994 ± 0.002	**0.093 ± 0.064**
vstan-hybrid	0.310 ± 0.017	**0.193 ± 0.011**	0.053 ± 0.003	0.216 ± 0.016	0.177 ± 0.012	0.024 ± 0.002	0.991 ± 0.002	0.088 ± 0.065
vsknn	0.311 ± 0.015	0.175 ± 0.011	0.057 ± 0.002	0.223 ± 0.015	0.178 ± 0.012	0.025 ± 0.002	0.989 ± 0.002	**0.113 ± 0.077**
vsknn-recency	**0.341 ± 0.016**	0.137 ± 0.029	**0.066 ± 0.005**	**0.256 ± 0.016**	**0.190 ± 0.012**	**0.029 ± 0.002**	**0.994 ± 0.003**	0.111 ± 0.076
vsknn-session	0.322 ± 0.012	**0.175 ± 0.010**	0.059 ± 0.002	0.232 ± 0.012	0.188 ± 0.008	0.026 ± 0.001	0.989 ± 0.002	0.111 ± 0.076
vsknn-hybrid	0.311 ± 0.015	0.175 ± 0.011	0.057 ± 0.002	0.223 ± 0.015	0.178 ± 0.012	0.025 ± 0.002	0.989 ± 0.002	**0.113 ± 0.077**
gru4rec	0.211 ± 0.004	**0.112 ± 0.004**	0.038 ± 0.001	0.150 ± 0.004	0.121 ± 0.003	0.016 ± 0.000	0.980 ± 0.001	0.080 ± 0.059
gru4rec-recency	**0.277 ± 0.017**	0.106 ± 0.009	**0.065 ± 0.004**	**0.236 ± 0.013**	**0.172 ± 0.011**	**0.029 ± 0.002**	**0.997 ± 0.001**	**0.098 ± 0.063**
narm	0.172 ± 0.013	0.087 ± 0.007	0.033 ± 0.002	0.125 ± 0.010	0.097 ± 0.006	0.014 ± 0.001	0.928 ± 0.010	0.107 ± 0.074
narm-recency	**0.261 ± 0.019**	**0.103 ± 0.008**	**0.062 ± 0.004**	**0.226 ± 0.013**	**0.167 ± 0.011**	**0.027 ± 0.002**	**0.992 ± 0.003**	**0.109 ± 0.070**
sr	0.211 ± 0.005	**0.114 ± 0.003**	0.040 ± 0.001	0.153 ± 0.006	0.125 ± 0.004	0.017 ± 0.001	0.966 ± 0.004	0.091 ± 0.067
sr-recency	**0.278 ± 0.017**	0.105 ± 0.008	**0.065 ± 0.004**	**0.236 ± 0.013**	**0.171 ± 0.011**	**0.029 ± 0.002**	**0.995 ± 0.001**	**0.103 ± 0.067**

with `vstan-hybrid` following closely (0.589). This suggests that while *reminders* can increase overall engagement and diversity, they may not always improve fine-grained ranking performance.

The impact of *reminders* was even more pronounced in the Xing dataset. The `vstan-session` model achieved the highest Hit Rate (0.349), while both `vstan-recency` and `vstan-session` obtained the best Recall score (0.258), significantly outperforming the baseline models.

Further improvements were observed in Precision and MAP, with `vstan-recency` yielding the highest MAP (0.029). These findings may highlight the effectiveness of *reminder* strategies in scenarios characterized by recurring user intents and extended temporal dependencies.

The Music4All dataset presented a more balanced scenario, with competitive performance observed across both baseline and *reminder*-based models. *Reminder* strategies yielded modest gains in some metrics but did not dramatically alter the ranking of top-performing methods.

The `gru4rec` model achieved the highest MRR (0.322), suggesting its strong capability for ranking quality even without *reminders*. Likewise, `sr` and `sr-recency` reached the best NDCG (0.285) and MAP (0.046), indicating that sequential modeling and *reminder*-based variations are both effective in this context.

Interestingly, the three `vsknn` variants that incorporate *reminders* achieved the highest Precision (0.171) and Recall (0.257), alongside the best Popularity scores, indicating their strength in recommending both accurate and diverse items. However, the improvements over their baselines were relatively small, suggesting that Music4All—possibly due to its large volume and rich item variety—already provides sufficient

TABLE IV
RESULTS FOR MUSIC4ALL DATASET

Models	Hit Rate	MRR	Precision	Recall	NDCG	MAP	Coverage	Popularity
stan	0.478 ± 0.032	0.230 ± 0.016	**0.160 ± 0.020**	0.251 ± 0.028	0.249 ± 0.023	0.043 ± 0.006	0.828 ± 0.029	**0.096 ± 0.055**
stan-recency	**0.479 ± 0.033**	**0.232 ± 0.016**	0.159 ± 0.020	0.251 ± 0.028	**0.249 ± 0.022**	0.043 ± 0.006	**0.831 ± 0.029**	0.088 ± 0.048
stan-session	**0.479 ± 0.033**	**0.232 ± 0.016**	0.159 ± 0.020	0.251 ± 0.028	**0.249 ± 0.022**	0.043 ± 0.006	**0.831 ± 0.029**	0.088 ± 0.048
stan-hybrid	**0.479 ± 0.033**	**0.232 ± 0.016**	0.159 ± 0.020	0.251 ± 0.028	**0.249 ± 0.022**	0.043 ± 0.006	0.830 ± 0.029	0.088 ± 0.048
vstan	0.481 ± 0.032	0.241 ± 0.016	0.158 ± 0.019	0.252 ± 0.027	0.253 ± 0.021	0.043 ± 0.006	0.892 ± 0.023	0.061 ± 0.040
vstan-recency	0.481 ± 0.032	**0.241 ± 0.015**	0.158 ± 0.019	0.252 ± 0.027	0.253 ± 0.021	0.043 ± 0.006	0.892 ± 0.023	0.061 ± 0.040
vstan-session	0.481 ± 0.032	**0.241 ± 0.015**	0.158 ± 0.019	0.252 ± 0.027	0.253 ± 0.021	0.043 ± 0.006	0.892 ± 0.023	0.061 ± 0.040
vstan-hybrid	0.481 ± 0.032	0.241 ± 0.016	0.158 ± 0.019	0.252 ± 0.027	0.253 ± 0.021	0.043 ± 0.006	0.892 ± 0.023	0.061 ± 0.040
vsknn	0.461 ± 0.026	0.211 ± 0.014	0.170 ± 0.021	0.255 ± 0.022	0.252 ± 0.021	0.045 ± 0.005	0.754 ± 0.033	**0.121 ± 0.073**
vsknn-recency	**0.464 ± 0.025**	**0.212 ± 0.013**	**0.171 ± 0.021**	**0.257 ± 0.022**	**0.253 ± 0.020**	**0.046 ± 0.005**	0.759 ± 0.032	0.114 ± 0.070
vsknn-session	**0.464 ± 0.025**	**0.212 ± 0.013**	**0.171 ± 0.021**	**0.257 ± 0.022**	**0.253 ± 0.020**	**0.046 ± 0.005**	**0.760 ± 0.032**	0.114 ± 0.070
vsknn-hybrid	**0.464 ± 0.025**	**0.212 ± 0.013**	**0.171 ± 0.021**	**0.257 ± 0.022**	**0.253 ± 0.020**	**0.046 ± 0.005**	0.759 ± 0.032	0.114 ± 0.070
gru4rec	0.460 ± 0.031	0.322 ± 0.025	0.139 ± 0.013	0.227 ± 0.018	0.256 ± 0.018	0.037 ± 0.003	0.949 ± 0.010	0.036 ± 0.026
gru4rec-recency	0.460 ± 0.031	0.322 ± 0.025	0.139 ± 0.013	0.227 ± 0.018	0.256 ± 0.018	0.037 ± 0.003	0.949 ± 0.010	0.036 ± 0.026
narm	**0.445 ± 0.032**	**0.296 ± 0.026**	0.153 ± 0.020	0.234 ± 0.023	**0.263 ± 0.023**	0.040 ± 0.006	**0.812 ± 0.022**	0.082 ± 0.049
narm-recency	0.445 ± 0.033	0.295 ± 0.025	0.153 ± 0.020	**0.235 ± 0.023**	0.263 ± 0.024	0.040 ± 0.006	0.811 ± 0.025	0.082 ± 0.049
sr	0.463 ± 0.031	0.313 ± 0.024	0.160 ± 0.021	0.252 ± 0.025	0.285 ± 0.026	**0.046 ± 0.006**	0.884 ± 0.017	**0.080 ± 0.049**
sr-recency	0.463 ± 0.031	**0.313 ± 0.024**	0.160 ± 0.021	0.252 ± 0.025	0.285 ± 0.026	0.046 ± 0.006	**0.885 ± 0.017**	0.080 ± 0.050

TABLE V
RESULTS FOR RETAILROCKET DATASET

Models	Hit Rate	MRR	Precision	Recall	NDCG	MAP	Coverage	Popularity
stan	0.751 ± 0.020	**0.561 ± 0.028**	0.120 ± 0.004	0.543 ± 0.018	0.514 ± 0.019	0.063 ± 0.002	0.978 ± 0.002	0.073 ± 0.008
stan-recency	**0.782 ± 0.022**	0.509 ± 0.035	**0.128 ± 0.004**	**0.571 ± 0.019**	**0.522 ± 0.024**	**0.067 ± 0.002**	0.989 ± 0.003	0.073 ± 0.008
stan-session	0.782 ± 0.023	0.508 ± 0.032	0.127 ± 0.004	0.570 ± 0.020	0.520 ± 0.023	**0.067 ± 0.002**	**0.992 ± 0.002**	**0.074 ± 0.008**
stan-hybrid	0.751 ± 0.020	**0.561 ± 0.028**	0.120 ± 0.004	0.543 ± 0.018	0.514 ± 0.019	0.063 ± 0.002	0.978 ± 0.002	0.073 ± 0.008
vstan	0.748 ± 0.019	**0.561 ± 0.028**	0.119 ± 0.004	0.540 ± 0.017	0.511 ± 0.019	0.063 ± 0.002	0.981 ± 0.002	0.070 ± 0.007
vstan-recency	**0.781 ± 0.022**	0.509 ± 0.035	**0.127 ± 0.004**	**0.569 ± 0.019**	**0.521 ± 0.024**	**0.067 ± 0.002**	0.990 ± 0.002	0.070 ± 0.008
vstan-session	**0.781 ± 0.022**	0.508 ± 0.032	**0.127 ± 0.004**	**0.569 ± 0.019**	0.520 ± 0.023	**0.067 ± 0.002**	**0.993 ± 0.002**	**0.071 ± 0.008**
vstan-hybrid	0.748 ± 0.019	**0.561 ± 0.028**	0.119 ± 0.004	0.540 ± 0.017	0.511 ± 0.019	0.063 ± 0.002	0.981 ± 0.002	0.070 ± 0.007
vsknn	0.748 ± 0.018	**0.552 ± 0.029**	**0.120 ± 0.004**	0.540 ± 0.016	0.515 ± 0.018	0.063 ± 0.002	0.981 ± 0.002	**0.080 ± 0.009**
vsknn-recency	**0.782 ± 0.021**	0.508 ± 0.035	**0.129 ± 0.004**	**0.571 ± 0.018**	0.521 ± 0.024	**0.068 ± 0.002**	**0.992 ± 0.002**	0.079 ± 0.010
vsknn-session	0.764 ± 0.019	0.548 ± 0.030	0.124 ± 0.004	0.554 ± 0.017	**0.529 ± 0.020**	0.065 ± 0.002	0.983 ± 0.001	0.079 ± 0.009
vsknn-hybrid	0.748 ± 0.018	**0.552 ± 0.029**	0.120 ± 0.004	0.540 ± 0.016	0.515 ± 0.018	0.063 ± 0.002	0.981 ± 0.002	**0.080 ± 0.009**
gru4rec	0.641 ± 0.017	0.455 ± 0.027	0.099 ± 0.003	0.467 ± 0.015	0.431 ± 0.018	0.052 ± 0.001	0.990 ± 0.002	0.059 ± 0.006
gru4rec-recency	**0.720 ± 0.022**	**0.489 ± 0.035**	**0.116 ± 0.003**	**0.531 ± 0.019**	**0.492 ± 0.024**	**0.061 ± 0.002**	**0.997 ± 0.001**	**0.061 ± 0.006**
narm	0.666 ± 0.029	0.489 ± 0.034	0.104 ± 0.004	0.485 ± 0.024	0.458 ± 0.026	0.055 ± 0.003	0.977 ± 0.003	0.072 ± 0.007
narm-recency	**0.730 ± 0.028**	**0.493 ± 0.037**	**0.117 ± 0.004**	**0.538 ± 0.023**	**0.498 ± 0.026**	**0.062 ± 0.003**	**0.989 ± 0.003**	0.072 ± 0.007
sr	0.612 ± 0.020	0.451 ± 0.025	0.093 ± 0.003	0.446 ± 0.018	0.422 ± 0.019	0.050 ± 0.002	0.959 ± 0.005	0.069 ± 0.007
sr-recency	**0.703 ± 0.025**	**0.482 ± 0.035**	**0.112 ± 0.004**	**0.518 ± 0.020**	**0.483 ± 0.025**	**0.060 ± 0.002**	**0.984 ± 0.004**	**0.070 ± 0.008**

context for effective recommendations.

For the RetailRocket dataset, we observe a consistent improvement in performance for models enhanced with the recency strategy across various metrics. Notably, `vsknn-recency` achieved the highest values in Hit Rate (0.782), Precision (0.129), Recall (0.571), and MAP (0.068), outperforming not only its baseline counterpart but also the other *reminder*-based variants, possibly due to the temporal dynamics and the short lifespan of item relevance in user sessions of this dataset. Similarly, other models such as `stan-recency`, `vstan-recency`, and `gru4rec-recency` also showed meaningful gains compared to their baseline versions, particularly in terms of Recall and MAP.

The hybrid strategy did not lead to improvements over the baseline models. For example, `stan-hybrid`, `vstan-hybrid`, and `vsknn-hybrid` reported identical or slightly lower results compared to their non-enhanced counterparts across almost all metrics. In contrast, the session similarity *reminder*-based models achieved intermediate results: better than the baseline models in Recall and Precision, but not surpassing the `-recency` variants, indicating that, while session similarity contributed to relevance, the temporal aspect played a more decisive role.

In terms of Coverage and Popularity, most models maintained high coverage levels, often exceeding 0.98. The `gru4rec-recency` model stood out with the highest coverage (0.997), suggesting that the recency mechanism also supports item diversity. Regarding Popularity, `vsknn` variants generally achieved the highest scores, particularly `vsknn-recency` and `vsknn-hybrid` (both 0.080).

V. Final Remarks and Future Directions

In this work, we investigated the effectiveness of Session-Based Recommender Systems (SBRS) with *reminders*, which incorporate previously viewed items to improve recommendation precision and relevance.

Overall, the experimental results highlight the effectiveness of incorporating *reminder* strategies into SBRS. Performance improvements were particularly consistent in domain-specific scenarios, such as the Xing and JusBrasilRec datasets, suggesting that *reminders* may be especially effective in fields like employment and law. In the RetailRocket dataset, *reminder* strategies yielded the most significant improvements across all domains, consistently outperforming the baseline on multiple metrics. These findings align with prior literature, which has already highlighted the effectiveness of *reminders* in the e-commerce domain, where users often revisit previously viewed items when making a purchase. In contrast, the Music4All dataset showed comparatively smaller gains, indicating that *reminders* may have less impact in domains with inherently rich and diverse interaction patterns, such as music. This also suggests potential for further exploration of *reminder* strategies in music recommender systems to better understand their effects in such contexts.

The recency, session similarity, and hybrid *reminder* strategies may prove effective in enriching the contextual representation of user sessions. These mechanisms have the potential to leverage latent temporal dynamics and inter-session relationships, potentially improving model responsiveness to user behavior patterns. As such, they represent a promising direction for advancing the performance and adaptability of SBRS.

Future work could explore alternative *reminder* strategies and investigate their applicability across a wider variety of domains and session characteristics. Further studies are also needed to assess the integration of *reminders* into different types of recommendation models, as well as to better understand how different *reminder* configurations impact recommendation quality. Additionally, conducting statistical significance tests on the results would help verify whether the observed improvements are meaningful and consistent across datasets, especially in cases where performance differences are subtle.

References

[1] G. Adomavicius, J. Bockstedt, S. P. Culey, J. Zhang, and S. Ransbotham, "The hidden side effects of recommendation systems", in MIT Sloan Management Review, vol. 60, n. 2, pp. 13–15, 2019.

[2] X. Chen, L. Yao, J. J. McAuley, G. Zhou, and X. Wang, "A survey of deep reinforcement learning in recommender systems: A systematic review and future directions", in Journal of the Association for Computing Machinery, vol. 37, n. 4, pp. 1–32, 2021.

[3] P. S. Jacobs, "Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval", L. Erlbaum Associates Inc., USA, 1992.

[4] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, "A review on deep learning for recommender systems: challenges and remedies", Artificial Intelligence Review, vol. 52, pp. 1–37, 2018.

[5] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks", arXiv preprint arXiv:1511.06939, 2015.

[6] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, "A survey on session-based recommender systems", ACM Computing Surveys (CSUR), vol. 54, n. 7, pp. 1–38, 2021.

[7] D. Jannach, M. Ludewig, and L. Lerche, "Session-based item recommendation in e-commerce: on short-term intents, reminders, trends and discounts", User Modeling and User-Adapted Interaction, vol. 27, pp. 351–392, 2017.

[8] M. A. Domingues, E. S. de Moura, L. B. Marinho, and A. da Silva, "Benchmarking Session-based and Session-aware Recommender Systems for Jusbrasil", Anais Estendidos do XXVIII Simpósio Brasileiro de Sistemas Multimídia e Web, pp. 145–148, 2022.

[9] L. Lerche, D. Jannach, and M. Ludewig, "On the value of reminders within e-commerce recommendations", in Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, pp. 27–35, 2016.

[10] M. A. Domingues, E. S. de Moura, L. B. Marinho, and A. da Silva, "A large scale benchmark for session-based recommendations on the legal domain", in Artificial Intelligence and Law, pp. 1–36, 2023.

[11] I. A. Pegoraro Santana, F. Pinhelli, J. Donini, L. Catharin, R. B. Mangolin, Y. M. G. da Costa, V. D. Feltrim, and M. A. Domingues, "Music4All: A new music database and its applications", in *2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 399–404, 2020.

[12] F. Abel, A. Benczúr, D. Kohlsdorf, M. Larson, and R. Pálovics, "RecSys Challenge 2016: Job Recommendations", in Proceedings of the 10th ACM Conference on Recommender Systems, Boston, Massachusetts, USA, pp. 425–426, 2016.

[13] R. Zykov, "Retailrocket Recommender System Dataset," Kaggle, 2022. [Online]. Available: https://www.kaggle.com/datasets/retailrocket/ecommerce-dataset. [Accessed: May 1, 2025].

[14] I. Kamehkhosh, D. Jannach, and M. Ludewig, "A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation", in Proceedings of the Workshop on Recommendation in Temporal Environments (RecTemp) at RecSys, pp. 50–56, 2017.

[15] K. Verstrepen and B. Goethals, "Unifying nearest neighbors collaborative filtering", in Proceedings of the 8th ACM Conference on Recommender Systems, pp. 177–184, 2014.

[16] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation", in Proceedings of the 11th ACM Conference on Recommender Systems, pp. 306–310, 2017.

[17] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms", User Modeling and User-Adapted Interaction, vol. 28, pp. 331–390, 2018.

[18] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "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, pp. 1069–1072, 2019.

[19] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Empirical analysis of session-based recommendation algorithms: a comparison of neural and non-neural approaches", User Modeling and User-Adapted Interaction, vol. 31, no. 1, pp. 149–181, 2021.

[20] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations", in Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 843–852, 2018.

[21] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation", in Proceedings of the 2017 ACM Conference on Information and Knowledge Management, pp. 1419–1428, 2017.