IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 14, 2021, accepted January 22, 2022, date of publication February 1, 2022, date of current version February 10, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3148434

From Data Analysis to Intent-Based Recommendation: An Industrial Case Study in the Video Domain

CESARE BERNARDIS[1], MAURIZIO FERRARI DACREMA[1], FERNANDO BENJAMÍN PÉREZ MAURERA[1,2], MASSIMO QUADRANA[2], MARIO SCRIMINACI[2,3], AND PAOLO CREMONESI[1]

[1]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milano, Italy
[2]ContentWise, 20158 Milano, Italy
[3]Mostly AI, 1010 Vienna, Austria

Corresponding author: Cesare Bernardis (cesare.bernardis@polimi.it)

ABSTRACT This work presents a comprehensive study, from an industrial perspective, of the process between the collection of raw data, and the generation of next-item recommendation, in the domain of Video-on-Demand (VoD). Most research papers focus their efforts on analyzing recommender systems on already-processed datasets, but they do not face the same challenges that occur naturally in industry, e.g., processing raw interactions logs to create datasets for testing. This paper describes the whole process between data collection and recommendation, including cleaning, processing, feature engineering, session inferring, and all the challenges that a dataset provided by an industrial player in the domain posed. Then, a comparison on the new dataset of several intent-based recommendation techniques in the next-item recommendation task follows, studying the impact of different factors like the session length, and the number of previous sessions available for a user. The results show that taking advantage of the sequential data available in the dataset benefits recommendation quality, since deep learning algorithms for session-aware recommendation are consistently the most accurate recommenders. Lastly, a summary of the different challenges in the VoD domain is proposed, discussing on the best algorithmic solutions found, and proposing future research directions to be conducted based on the results obtained.

INDEX TERMS Industrial data, intent-based, session-based, user behavior.

I. INTRODUCTION

The benefits of recommender systems are clear. They guide users in the exploration of immense catalogs of products, and they leverage user behaviors to generate accurate, interesting, novel, and serendipitous recommendations [1]–[3]. One of the most important aspects in the design and evaluation of a recommender system is the data used to profile users and provide recommendations. In literature, many published research papers use *research datasets* such as MovieLens [4], or 30MUSIC [5] to perform offline experiments. However, these datasets hardly reflect the challenges posed by a real scenario. They rarely reach the magnitude of a true system in terms of number of interactions, users, and items, and they hide the complexity of building a dataset from raw collected data. On the other hand, *industrial datasets*, i.e., datasets constructed from the logs of real-world recommenders, pose different challenges in their construction and usage, and they can reproduce more accurately a real application scenario. For example, they introduce the impression bias, i.e., the fact that the pattern of interactions between users and items is influenced by how items are presented to the users [6]. Moreover, the size of industrial datasets might be orders of magnitude larger than research datasets and they are heavily noisy. Given these differences between research and industrial datasets, understanding the construction and processing steps that bring from a low-level interaction log to a high-level dataset ready for an algorithm is important for the research community. These constructions steps have a strong impact, equal to or even greater than that of the

The associate editor coordinating the review of this manuscript and approving it for publication was Le Hoang Son.

algorithm itself, on the final quality of the recommendation process [7], [8]. However, few works using industrial datasets [6], [9] have addressed these important aspects.

In this work, the construction, processing, and cleaning of an industrial dataset is described in detail. The dataset is composed by logs collected in a 6-months period from an Over-The-Top Media service (OTT), which provides Video-on-Demand (VoD) media content via the internet. Similarly to a real case study in an industrial environment, the new dataset is then employed to perform an offline comparative study of several intent-based recommendation techniques in next-item recommendation. Next-item recommendation is the typical task that an online recommender system is asked to accomplish, and it consists in predicting the item involved in the next action performed by a user. Intent-based recommenders exploit past user interactions, items and users metadata, and any other information available to assess the *intent* of a user, and generate recommendations accordingly. A wide range of approaches is confronted, from simple baselines to sophisticated session-based techniques for sequence-aware recommendation. Session-based recommenders, in particular, assume that the intent of the users can be inferred from the set of interactions that the user performed in the same session and they have proved to be particularly effective for next-item recommendation [1], [2], [10]. Note that the goal of this paper is not to perform a survey on the most recent state-of-the-art intent-based algorithms,[1] but to propose a real-world case study that discusses the whole path from raw data to recommendation, in order to fill the gap between research and industrial works, especially in the VoD domain.

The different steps that were preformed to accomplish this study are depicted in Figure 1. (i) Initially, data was gathered from the online system. A description of the collected data, that include user actions, such as clicks, views and ratings, and VoDs metadata, is provided in Section III-A. (ii) Collected data was then preprocessed, in order to remove noisy and inconsistent information. The various preprocessing steps that raw data collected from an online system required are discussed in Section III-B, covering the removal of repeated actions, the handling of key missing information, and the filtering of incoherent system interactions. (iii) Preprocessed data was analyzed to study user behavior, identify common habits, and infer user sessions. Sessions are crucial to deduce short-term intent of users, and lead to superior accuracy in the next-item recommendation task. In Section III-C, the results of the analysis are shown, and the methodology used to identify user sessions is discussed. (iv) User sessions were studied and filtered, in order to exclude data coming from bots or test users. In Section III-D, the results of the analysis are described. These results were then used to filter sessions as shown in Section III-E. (v) In order to perform the offline evaluation of common intent-based recommendation techniques (described in Section IV-A), the available data was partitioned in training, validation and test sets. The evaluation procedure, including the partitioning scheme, and the evaluation protocol for offline experiments, is discussed in detail in Section IV-E. (vi) Finally, the accuracy of the intent-based algorithms in the next-item recommendation task was compared. The whole testing procedure, including its results, are described in Section V. A thorough description of the hyperparameter optimization procedure adopted is also reported, as well as an analysis of the impact of within session (i.e., session length) and across session (i.e., the number of previous sessions available for a user) information.

To summarize, the main contributions of this paper are threefold. First, illustrate the construction, processing, and analysis of an industrial dataset. Second, perform behavioral analysis of users in the VoD domain. Third, study the accuracy of several intent-based techniques for VoD recommendation on the new, real-world dataset. To the best of our knowledge, this is the first study that presents a detailed analysis, in a real industrial environment, of the whole process between raw data, and next-item recommendation in the VoD domain.

II. RELATED WORKS

Generally, two types of datasets are used for research in the field, namely *research* and *industrial* datasets. Research datasets are usually small and publicly available. Examples of these are *30MUSIC* [5], [13] and *#NOWPLAYING* [13], [15] for music recommendations, and Movielens [4] for movies recommendations. The main drawback of research datasets is that they lack sufficient users, items, interactions, or metadata to model all user profiles. Moreover, research datasets hide from the research community the challenges related to the processing steps required to build a dataset from real interaction logs.

On the other hand, industrial datasets are extracted from logs of deployed recommender systems serving real users. *ContentWise Impressions* [6], *MIND* [16], *Plista* [16], [17], and *FINN.no* [18] are examples of research-formatted industrial datasets. The main benefit of industrial datasets is that they cover much more user profiles due to their enormous size. This advantage comes with the challenges of cleaning and processing real-world data, i.e., massive amount of information to process, users' privacy, and biases from previous or existing recommenders [19]. There are two main differences between the aforementioned works and this one. First, those datasets, with the exception of ContentWise Impressions, are not from the VoD domain. Second, they do not provide extensive user analysis and a comparison of several intent-based recommendation techniques.

Similar to this work, others have explored and analyzed user patterns, preferences and behaviors, although in different domains or recommendation scenarios. In the news domain, [19] collected, analyzed, and built a dataset for news recommendations on different countries and regions. They concluded that users have consistent interests with

[1] Please, refer to [11]–[14] for surveys on intent-based recommendation.

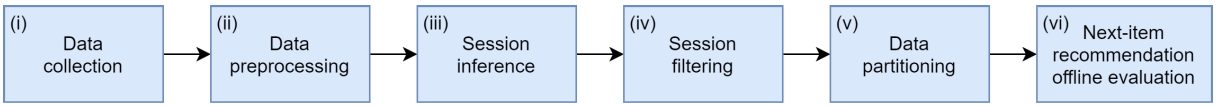


FIGURE 1. Flowchart representing the steps presented in this study.

specific news categories but might be influenced by local news trends. They used these findings to craft a more accurate recommendation technique.

Other case studies, like the one presented in this work, have encountered differences of user interests between and within domains. Therefore, it is important for the community to present several case studies in different industrial context and recommendation scenarios, so to better model user preferences. Differences of user interests have been discovered in particular within the news domain. For instance, [19] discovered differences between user preferences in web searches and news recommendations. Reference [20] stated that user preferences with everyday news recommendations cannot be extended to services providing highly-specialized news, like magazine articles, profiles of relevant people, and timeless articles. Lastly, [21] found differences between user preferences based on the target user of the service. For instance, user preferences cannot be generalized to users of another system if they target different types of users, e.g., from another country or anonymous users.

Other works in literature, similarly to this one, have explored the recommendation quality of state-of-the-art recommenders, but they present several differences. Reference [22] shows a description of Netflix's recommenders and their usual evaluation scenario, but it does not report a detailed analysis of data processing steps. Reference [12] presents a survey of the current state-of-the-art recommendation techniques, including *intent-based* ones. However, the authors solely describe these techniques and they do not perform an offline evaluation. References [11], [14] and [13] present evaluations of several recommendation techniques, including some *intent-based* ones against datasets of several domains. However, these works are not from the VoD domain, no user behavior analysis is performed, and their goal is to benchmark several recommendation techniques in the general case. For example: sessions are split using ''a commonly used 30-minute user inactivity threshold'' instead of analyzing user behaviors to select the inactivity threshold. Lastly, [9], [23]–[25] present evaluation studies similar to this work, but they are performed in a different context or recommendation scenario.

III. DATASET OVERVIEW

The dataset analyzed in this work is collected over a period of 6 months from an Over-The-Top Media service (OTT) which provides Video-on-Demand (VoD) media content via the internet. The dataset contains different types of user interactions with items and each item metadata. As the data is gathered from a real world system, several issues need to be addressed to ensure that it can be reliably used for such an analysis. Aligned with recent and important research about dataset building [26]–[28], the dataset used in this study is thoroughly described, including its content, features, preprocessing, and the strategy adopted to infer sessions. Note that building datasets is a complex task, this description might provide useful insights for researchers and practitioners.

A. DATA DESCRIPTION

The dataset contains user *actions* (clicks, views, ratings) as well as item metadata, that will be described in this section. This study considers the problem of session-based recommendation, in which a session refers to a group of user actions that occur in a continuous period of time, e.g., the user logs in to the platform and interacts with a few VoD before logging out. Sessions can have varying lengths, multiple sessions can occur within the same day or be separated by longer time periods. Note that the dataset does not contain information on the session in which each action was performed by the user. Since the intent of the user is usually bounded to the current session, session identifiers are generated based on the user actions (see Section III-C).

1) USER CLICKS

A *click* refers to the user accessing the details page of a VoD item. The dataset contains 87M click events generated by 908k distinct users on 41k distinct VoD items. Clicks frequently contain repetitions, i.e., consecutive clicks on the same item by the same user.

2) USER VIEWS

A *view* refers to the user viewing a VoD item. Only a fraction of the user click actions are converted into a view (i.e., users consume only a subset of the items they explore). Since views represent the actual VoDs consumed by the user, they are a strong indication of the user preferences. The dataset contains 129M view events generated by 1.12M distinct users on 41k distinct VoDs. Note that some users have a view for an item but no clicks for that item, due to the fact that they did not access the detail page. View events are categorized into: *started* (first play of the item), *paused* (every time the item has been paused by the user) and *ended* (the item has been watched by the user in its entirety). The detailed breakdown of the view events per category is reported in Table 1. It can be seen that the majority of view events are either *started* or *paused*. On average there is an equal number of *started* and *paused* events. The low number of *ended* events indicates that less than half of the started items are actually watched by the user entirely. It should be noted that *ended* events are

TABLE 1. User views statistics before data preprocessing.

View Event Type	Count (M)	%
Started	54.7	42.44
Paused	52.0	40.34
Ended	22.2	17.22
Total	128.9	100.00

registered only when the item has been completely watched by the user, credits included. However, most users likely skip the credits, and therefore the *ended* event is not properly logged. Upon closer inspection it could be observed that some *paused*/*ended* events do not refer to a preceding *started* event in the log of user activity. These events are likely due to logging mistakes and will be referred to as *dangling*.

3) USER RATINGS

Less than 5% of the overall actions are explicit loved/unloved feedback. This is expected since users seldom provide explicit feedback. While explicit feedback is a valuable asset for every traditional recommender system based on historical user preferences, its value has not been established yet for intent-based recommendation. This is due to both the scarcity of explicit feedback and to the low impact explicit feedback have for intent-based recommendations. In a recommender system that relies upon historical preferences, further feedback from past interactions allow to build a more complete long-term user profile. In an intent-based recommender, however, long-term preferences have a lower impact on the current intent of the user. Moreover, the user could be drawn to add explicit feedback for an item they interacted with in the past during a different session, hence, with an intent that could be different with respect to that of the current session. This type of ''out-of-place'' feedback could confuse the intent-based recommendation engine.

4) METADATA

Each item in the VoD catalogue is associated with metadata such as: episode name and title, genres, language, series identifier (e.g., for TV series episodes), episode number, season number, release date, director, actors, a summary, etc. For the purpose of this study it is of particular interest the item *availability window*. The availability window represents the period of time the VoD item was available to users and, therefore, recommendable to them [29].

The lack of interactions with an item outside its availability window could be erroneously interpreted as a strong negative signal for that item by any model not considering the availability window, thus introducing a strong bias in the model itself. To prevent this, items should not be considered outside their availability window during training. The metadata contains the start and end date of the license agreements that allows the OTT to provide the VoD item to the user. However, some VoD items can change availability window over time (e.g., in case of special offers from the broadcaster) and the metadata does not report this information. For this reason, the availability window is defined using the actual user actions, the beginning of the availability window will be at the first user interaction and end at the last interaction.

B. DATA PREPROCESSING

The data is gathered from the OTT in the form of logs, which contain several repeated events. In the scenario of interest for this study, recommending VoD items that the user has already seen is allowed. However, the presence of repeated consecutive interactions with the same item can bias some intent-based algorithms (sequence-based especially) to learn that, after a user interacts with a VoD item, they will likely interact with same item again, leading to trivial and uninteresting recommendations. To retain only the useful information from the user logs and to avoid such bias, the data is preprocessed to remove duplicated consecutive interactions of the same category:

- **User Clicks**: repeated clicks of a user on the same item (accessing the item's detail page) are compressed into a single click with the timestamp of the first one. In this way, the number of click events is reduced from 87M to 60M (-28%).
- **User Views**: one of the issues encountered is to establish whether a VoD item was viewed entirely or not. Unfortunately, the view time is not explicitly logged in the data. In principle, it should be possible to approximate the view time using the time that separates *started* events from *paused* or *ended* events. However, many *started* events do not correspond to any *paused* or *ended* event. Similarly, it also happens that some *paused* or *ended* events are not associated to any *started*. In all these cases, estimating the view time is not easily possible. For these reasons, the view time was not considered. If a user performed multiple view actions on the same VoD item, only the first one is kept, the following ones are removed. This preprocessing has the disadvantage of considering as a positive interaction even a VoD item watched just for a few seconds. Unfortunately, not much else can be done in the absence of further details on the actual viewing time. The result of this preprocessing is shown in Table 2. The distribution of event types in the dataset changes significantly. As can be expected, the majority of the events (i.e., the first event available for the interaction of the user with a VoD item) is of type *started*. However, it can also be seen that there is a significant number of *paused* and *ended* events. Those are referred to as *dangling* events, since the relative *started* is missing from the data.

C. INFERRING SESSIONS

Recommender systems literature has widely shown that sequentially-ordered user interaction logs represent a rich source of information, and they can be critical for the success of a recommender [30]. Indeed, being able to distinguish long-term preferences (from interactions farther

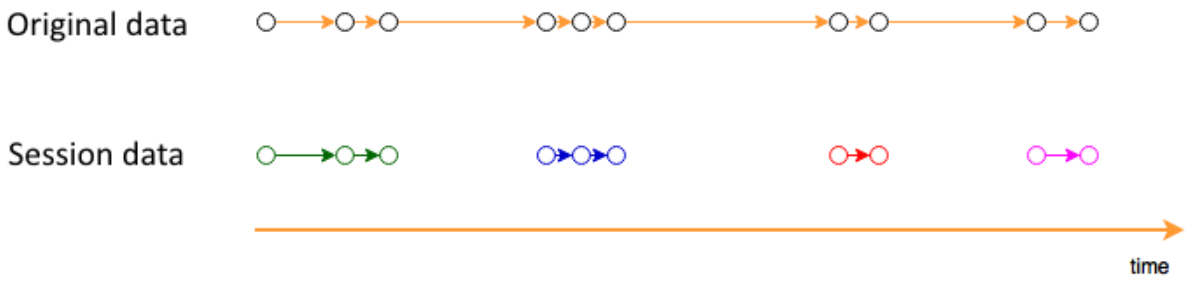


FIGURE 2. Example of session creation from a sequence of actions performed by a single user, based on idle-time. Circles represent actions. Arrows connect actions in the same session. Different sessions have different colors.

TABLE 2. User views statistics before and after data preprocessing.

View Event Type	Before Preprocessing Count (M)	%	After Preprocessing Count (M)	%
Started	54.7	42.44	25.7	85.67
Paused	52.0	40.34	2.8	9.33
Ended	22.2	17.22	1.5	5.00
Total	128.9	100.00	30	100.00

in time) and the short-term intent of a user (from the most recent interactions) can benefit the quality and the accuracy of recommendations. The short-term interest of a user, in particular, is highly relevant in intent-based recommendation. Being able to deduce what is the purpose of the user while he/she is using the service allows to personalize and contextualize the next recommendations generated by the system. Usually, short-term user intent is inferred through the interactions performed in a *session*, i.e., a relatively small number of interactions that are carried out consecutively by the user in a limited period of time. Past sessions of the same user, instead, provide information about his/her long-term preferences.

Generally, user sessions can be identified directly by the system when, for example, log-in and log-out events are recorded. Unfortunately, the dataset collected for this work did not contain this type of events, since logs were represented by independent user interactions without any session reference. However, the availability of the time when each interaction was recorded allowed to infer user sessions from data.

When log-in and log-out events are not available, a common assumption made in the literature is that a user session ends when the user has been idle for a sufficient amount of time. Defining the correct idle time to separate user history in sessions is an actively investigated research topic [1], [2]. Since a session is also associated to a user intent, if sessions are inferred too long then different user intents may be merged, while if they are too short the data may become more noisy. More sophisticated models, e.g., based on survival analysis [31], could be applied but comparing their strengths and weaknesses is beyond the scope of this work. For this study the idle time strategy is used, because it is simple, effective, and in the scenario of interest, as will be described shortly, the user behavior emerges rather clearly allowing to easily choose an appropriate threshold.

Figure 2 shows an example on how sessions can be created based on idle-time. In the example, the chain of events that constitutes the whole user history is represented as circles. Idle periods between subsequent events are represented by arrows. The sessions are created by breaking the chain into several sub-chains (4 in the example), i.e., the sessions, in correspondence of arrows longer than the selected idle-time. It is easy to see that by changing the minimum idle-time threshold different number of sessions can be generated, of different length.

The appropriate time threshold can be selected by analyzing the distribution of the temporal distance between clicks and views to obtain a better understanding of the temporal dynamics in this domain. From Figure 3 it can be seen that the distribution of the temporal gap between subsequent actions is tri-modal.

- **Short-term actions**: The first peak ($\approx$ 10 seconds), on the left of Figure 3, corresponds to *short-term* actions that occurred in the same user session and is mainly composed by click actions (e.g., the user is browsing the catalog before choosing which content to watch).
- **Mid-term actions**: The second peak ($\approx$ 40 minutes), the blue dashed line on the center of Figure 3, corresponds to *mid-term* actions that happen when the user interacts again with the interface, e.g., after they have watched an episode of a series or a movie. The user will likely look for content related to the previously watched one (e.g., if binge watching, the user will likely watch the following episode of the series). Consequently, it can be assumed that the intent of the user remains the same and therefore the session has not changed.
- **Long-term actions**: The third peak ($\approx$ 23 hours), the yellow dashed line on the right of Figure 3, corresponds to *long-term* actions, e.g., the user coming back to the service after some time (days or weeks). Interestingly, this distribution is peaked around 24h indicating that many users come back to the service after one day (e.g., the user interacts with the OTT at around the same time, which may be an effect of their daily routine).

Note that the distributions of clicks and views is different. In particular, the distribution of views is bi-modal and

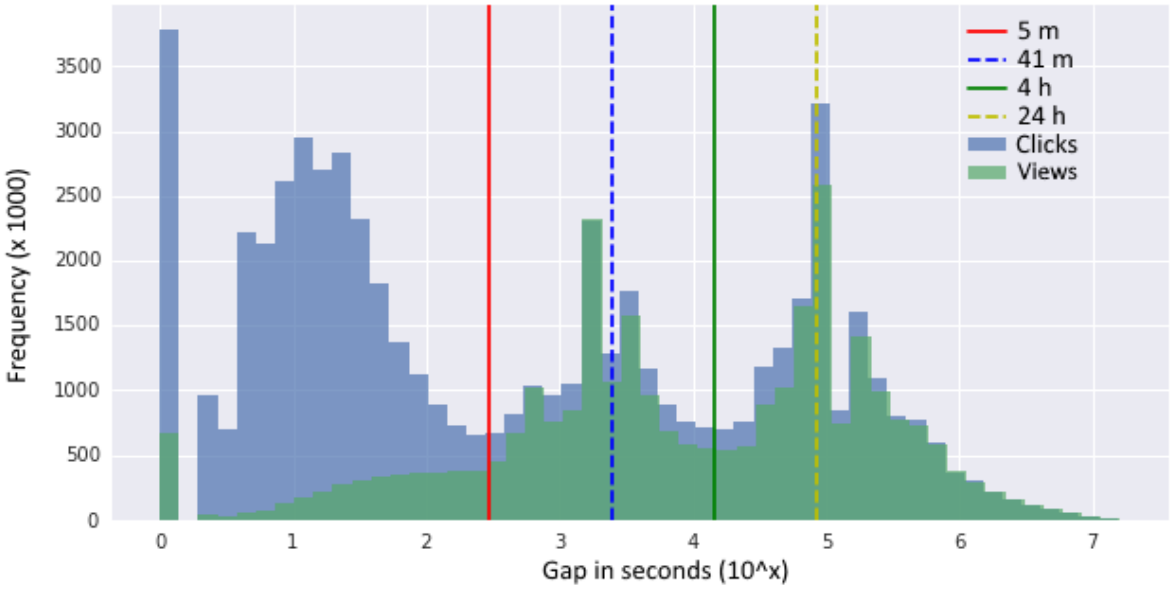


FIGURE 3. Distribution of the gap between subsequent clicks (in blue) and views (in green) considering all users. Notice that the x-axis is in log-scale.

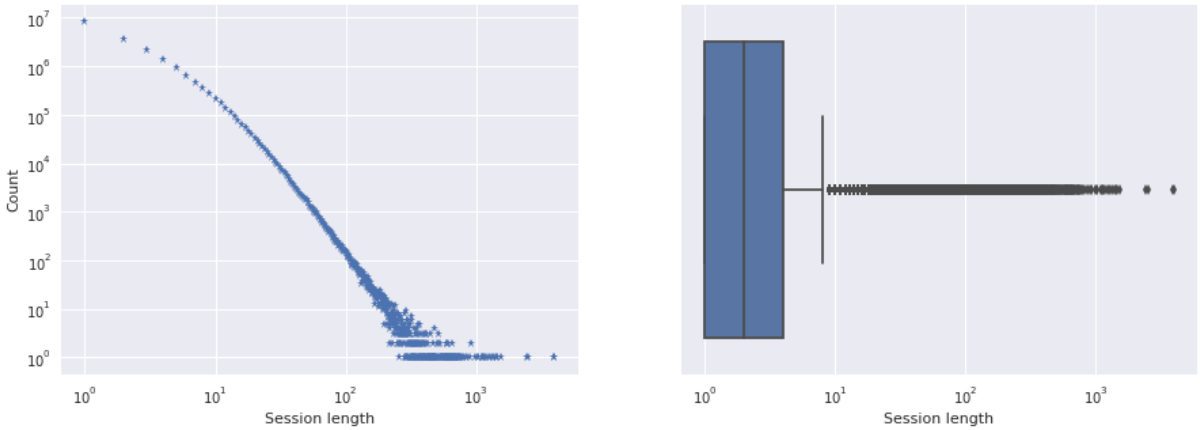


FIGURE 4. Number of sessions sorted by their length, i.e., the number of actions they contain, and the relative quantile box plot.

almost coincides with the mid-term and long-term action distributions. Furthermore, the temporal distribution between views is already contained in the distribution between clicks, i.e., almost every view is preceded by at least one click, but not the other way around. Since the goal is to obtain sessions composed by subsequent view events, a reasonable threshold to split sessions is approximately 4 hours, corresponding to the lowest point of the distribution between the mid-term and long-term components (green continuous line in the figure). Note that this threshold is larger than what is commonly used in other domains; e.g., typical values are around 30 minutes for e-commerce and music [13]. This difference can be explained because VoD consumption takes more time than music or e-commerce, so it is reasonable to use larger inactivity thresholds.

The resulting dataset contains 19.5M sessions of varying length, between 2 and more than 100 user actions. Sessions may contain repeated items if the user both clicked and viewed the same item. If the session contains both a view and a click for an item, the click event is removed. The view event over the click has been retained because views are a stronger positive signal.

D. SESSION DATA ANALYSIS

This Section analyzes the resulting session dataset. Figure 4 shows the number of sessions sorted according to their increasing length represented as the number of actions they contain. It is possible to see that session length follows a power-law distribution. Most sessions are very short in length, with a median length of 2 actions. This means that users typically browse few items before choosing what to watch. There are longer sessions, likely in correspondence with sessions where users browsed the catalog more extensively, or when the user watched several episodes of the same

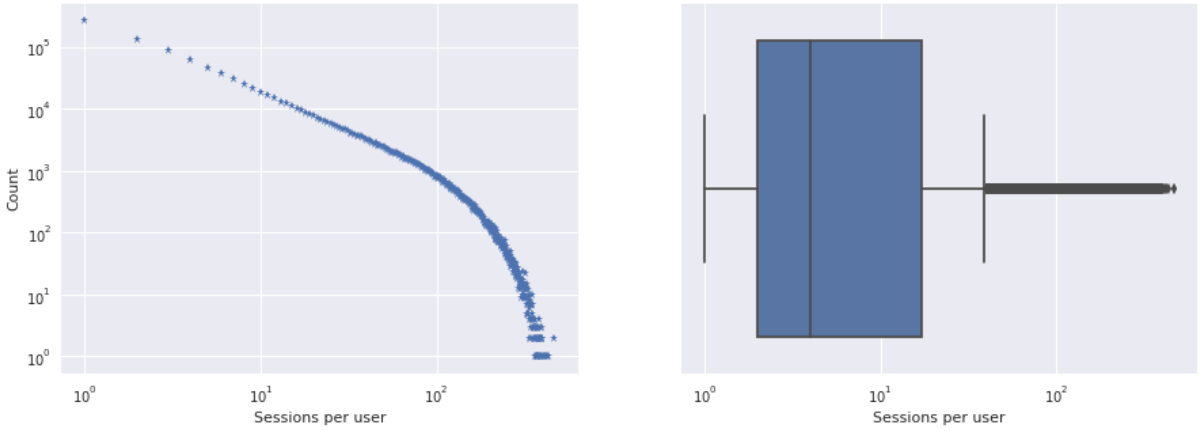


FIGURE 5. Number of sessions per user sorted by more active users, i.e., from the most active users to the less active, and the relative quantile box plot.

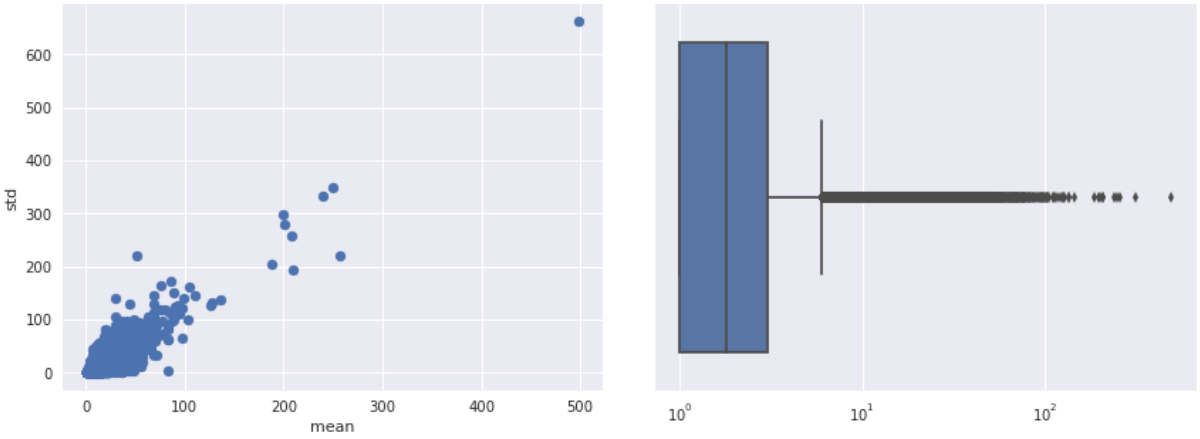


FIGURE 6. Mean session length per user. On the left the mean and standard deviation, on the right the mean session length quantile box plot.

series (i.e., binge watching). Finally, there are some outliers with a few sessions being longer than ∼100 actions, probably due to bot traffic or platform tests.

Figure 5 shows the number of sessions per user. While there's a large number of users with few (1-2 sessions), most users have between 3 and 11 sessions in the 6-month period covered by this dataset. Again, there are outliers with more than 100 sessions.

Figure 6 shows the average session length per user. Coherently with the previous findings, most users have, on average, less than 10 events per session. However, some users have sessions with a very high average number of actions, more than 50 or even 100. The Figure allows to identify clearly some outliers, that is users that have a high number of anomalously long sessions. These users are outside of the typical user behavior.

E. SESSION FILTERING

Based on the previous analysis, there exist some sessions that are either too short or likely generated by bots or test users. The session data is further processed to retain only the sessions that meet the following criteria:

- The minimum length of a session is 2 events.
- The average number of events per session of a user is less than 50.

The first criterion discards sessions of length 1 which are not interesting for intent-based recommendation. The second criterion aims at discarding bots or test users who systematically have a huge number of interactions per session. There are 206 users that did not comply with this rule, which is a negligible amount with respect to the overall number users in the final the dataset. After this further data cleaning, the number of sessions drops from 19.5M to 11.3M.

IV. NEXT-ITEM RECOMMENDATION EXPERIMENTS

The main goal of these experiments is to compare the accuracy performance, on the newly created dataset specifically, of the most common approaches for intent-based recommendation in the next-item recommendation task, that consists in predicting which is the item involved in the next action performed by the user. In particular, the

analysis focuses on session-based approaches, that represent an effective branch of intent-based techniques for next-item recommendation. Note that assessing the most accurate approach in the general case is out of the scope of this work.

The first part of this Section focuses on the definitions of session-based and session-aware recommendation reported in literature, arguing about the main differences between them. Then, a presentation of the most common algorithms for session-based recommendation adopted in these experiments follows. Finally, the entire evaluation procedure employed for the experiments on the next-item recommendation task is described in detail.

A. SESSION-BASED AND SESSION-AWARE RECOMMENDATION

In *session-based* recommendation, the sequence of events in the current session is the only information used by the recommender to suggest items to the user. The underlying assumption is that the intent of a user engaged in a session, can be mined directly from the sequence of actions performed so far in the session itself [30]. However, in many online systems, including the one studied in this paper, it is also possible to identify the users involved in the different sessions (e.g., through a login procedure, cookies, etc.). This allows to leverage also the taste of a user, inferred from past behavior, when new recommendations are required. This scenario is called *session-aware* recommendation: the recommender system can be built based on a combination of long-term and short-term interest models [30].

There is a significant amount of literature that has been recently developed in the context of session-based and session-aware recommender systems and a wide variety of different approaches have been proposed. These can be further classified into two main categories:

- **Sequence-free models**, that don't take into account the sequence of events in the current session (i.e., the events in the session are treated as a set).
- **Sequence-aware models**, that explicitly model the sequentiality in the user interactions.

In the next sections, the selection of approaches adopted in the experiments is described more in detail.

B. SEQUENCE-FREE MODELS

Sequence-free models are usually less complex and hence less computationally demanding than sequence-aware ones, since they don't require to model the sequential aspects of the session, which can be extremely difficult and, sometimes, domain dependent. However, several works have empirically proved that the accuracy of these models is competitive against that of sequence-aware techniques in the session-based recommendation scenario [13], [32]. In these experiments, two similarity-based models are considered as representative for this family of recommenders.

In similarity-based models, sessions are first represented into a session-item matrix. Each session is encoded into a fixed-length vector, as shown in Figure 7, which values can be:

- binary, if the intent is to keep track of whether in the session there was an interaction with an item or not;
- counts, if the intent is to keep track of how many times the user has interacted with an item in the session.

The binary encoding has been selected for our experiments, as it represents a well suited choice. Indeed, as shown in Section III-D, the average number of interactions in a single session of the dataset is between 2 and 3. This means that sessions are generally very short and that the repetitions of an item are low and not very relevant.

Once the sessions have been encoded into their vector form, the resulting session-item matrix can be processed to extract information about items and sessions (e.g., co-occurrence counts, or session similarities) in order to generate recommendations. In this work, two well known nearest neighbors approaches and a matrix factorization technique that belong to the session-based recommendation literature are compared: Session ItemKNN, ContextKNN and Session BPR.

1) SESSION ITEMKNN

In this technique, the interactions contained in the session-item matrix are modeled through item-to-item similarities. The similarity between two items is defined as the shrinked cosine between their occurrence vectors (i.e., the respective columns in the session-item matrix). For each item, only the k most similar neighbors are considered, while all the other similarities are set to 0. The next-item recommendation for a session is finally computed based only on the last item that occurs in it: the model recommends the most similar items to the one involved in the last interaction of the session. Note that this approach discards the previous actions of the user in the session. However, despite its simplicity, it works well in practice [33], [34].

2) CONTEXTKNN

Differently from Session ItemKNN, this technique exploits the similarity between sessions instead of the co-occurrence patterns between items [32], [35]. This approach has been originally described in the context of playlist generation [36] and it is very similar to a user-based collaborative similarity model where each session is treated as an independent user profile. Given a session, the model recommends items that frequently occur in the sessions that are most similar to the given one. In the experiments, both cosine and Jaccard similarity functions were considered. However, only the results related to cosine similarity are shown in the experimental section of this work, because cosine proved to be the best performing similarity during the hyperparameter optimization procedure.

3) SESSION BPR

It is a matrix factorization model in which sessions and items are represented by vectors of latent factors. This model uses

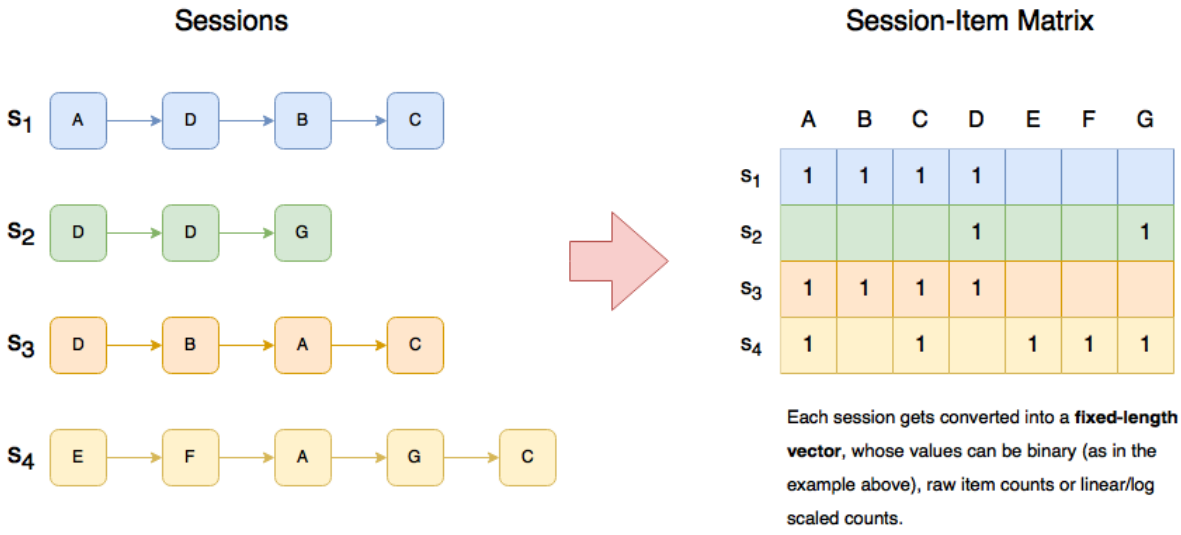


FIGURE 7. Example of session encoding as fixed-length vectors and generation of the session-item matrix.

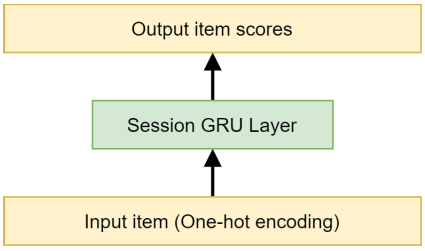


FIGURE 8. Graphical representation of the architecture of GRU4Rec.

Bayesian Pairwise Ranking loss to learn to rank items in a session. Note that matrix factorization cannot be directly applied to session-based recommendations, since it is not possible to learn latent representations for sessions that are not available during the training phase. It follows that during recommendation each session is represented by the average of the representations of the items involved in the interactions that compose it [1], [13].

C. SEQUENCE-AWARE MODELS

Sequence-aware models are more powerful than the sequence-free counterparts. They can adapt next-item recommendations for a session depending on the actual sequence of events. Contrarily to sequence-free approaches, two sequences composed by exactly the same items that appear in different order can result in different recommendations. These models have been recently proven to largely outperform every other approach in the session-based recommendation task [1], [13]. Among them, Recurrent Neural Networks (RNN) have shown to be particularly adapt for this task, thanks to their ability to model sequential data. In this work, the focus is on two well known state-of-the-art approaches based on Recurrent Neural Networks: GRU4Rec [1] and its more recent evolution H-GRU4Rec [2].

1) GRU4Rec

GRU4Rec is the implementation of the session-based, sequence-aware model proposed in [1]. It is a Recurrent Neural Network that takes as input the one-hot encoding of the active item in a session, and provides as output, for each item, the likelihood of being the next in the session. The prediction is based on both the input item and the state of the network, that represents the historical information about the current session and depends on the sequence of items that occur in the interactions that preceded the actual one in the session. Gated Recurrent Unit [37] layers are employed, in order to deal with the vanishing gradient problem that afflicts common RNN units. The implementation of GRU4Rec that was adopted had one single GRU layer composed by 100 neurons, and was trained using parallel mini-batches, as suggested in [1]. A graphical representation of the model architecture is shown in Figure 8. All the different losses proposed by the original authors (i.e., BPR, TOP1, BPR-max and TOP1-max) were tested, and BPR-max was selected, since it provided the highest accuracy. A popularity proportional probability was employed to select negative item samples in the BPR optimization procedure. In particular, the popularity of each item was exponentiated by a factor, and divided by the sum of all the transformed popularities, in order to soften the differences between very popular items and niche ones. The exponent factor was treated as a hyperparameter of the model and optimized accordingly.

2) H-GRU4Rec

H-GRU4Rec is the session-aware recommendation technique proposed in [2]. The aim of the model is to improve GRU4Rec by adding the ability to track the evolution of the user interests over time. This is performed adding a *user-level* GRU layer to the *session-level* GRU layer already present in GRU4Rec,

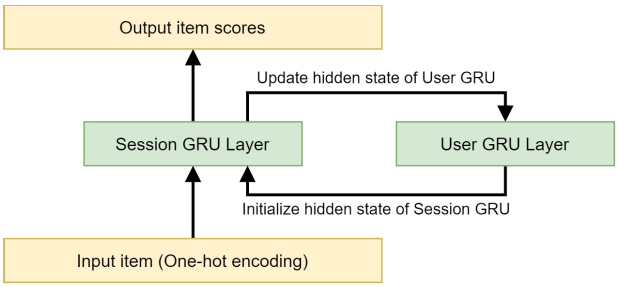


FIGURE 9. Graphical representation of the architecture of H-GRU4Rec. Differently from GRU4Rec, an additional GRU layer is in charge of keeping track of user interests over time, across user sessions.

in order to model information across user sessions. When a new session for a certain user is provided as input, the state of the user-level GRU layer is used as initial value for the state of the session-level GRU layer. Then the state of the session-level evolves as in GRU4Rec and its final state (i.e., the state of the GRU layer after all the items of the session have been given as input to the network) is used to update the state of the user-level GRU. The implementation of H-GRU4Rec that was employed had one user-level GRU layer and one session-level GRU layer, both composed by 100 neurons, and was trained using user-parallel mini-batches, as suggested in [2]. As for GRU4Rec, BPR-max proved to be the best performing among the different losses that were tested. The same popularity proportional approach for negative sampling proposed for GRU4Rec was adopted. The architecture of the implementation of H-GRU4Rec is shown in Figure 9.

D. OTHER MODELS

Besides sequence-related methods, some common baselines are considered.

- Global Pop is a non-personalized recommendation model. It recommends the items with the highest number of interactions. [1]
- User Pop is a semi-personalized technique that recommends the items with which the user engaged in the session has the highest number of interactions. It is based on the assumption that users will likely repeat actions done in the past [2].
- Session Pop is a semi-personalized approach that recommends the items that have the highest number of interactions in the session itself. It is based on the assumption that users will likely repeatedly inspect the same items within a session. [1]
- Collaborative ItemKNN is the traditional nearest neighbors, item-based approach with cosine similarity [3]. The algorithm builds user profiles using the respective view events during the training period and item similarities are computed based on these profiles. Given a session, the most similar items to those in the user profile and those he watched during the session are recommended.

TABLE 3. Statistics of the splitted data.

	Training	Validation	Test
length	17 weeks	2 weeks	2 weeks
events	39,297,814	5,516,505	6,445,285
sessions	7,107,555	968,657	1,183,853
users	592,595	257,102	288,594
items	33,933	15,725	17,609

E. EVALUATION PROCEDURE

In order to evaluate different intent-based approaches, this section defines the dataset partitioning, the evaluation protocol and the evaluation metrics.

1) DATA PARTITIONING

In order to retain the temporal ordering between interactions a temporal data partitioning is used, commonly found in the literature [1], [2], [38]. Specifically, given a partitioning timestamp, all the sessions that started before that timestamp are assigned to the training set and the remaining ones to the test set. The training set is further divided into a training and validation set to be used for hyperparameter optimization. Note that a session is never split. A session that begins before the splitting timestamp and ends after it, it is considered as a training session.

a: EVENTS PER DAY

The distribution over time of the number of events per date and day of the week are shown in Figure 10 and Figure 12. There is a relatively small volume of events in the first 2 weeks, then it drastically increases. The volume of events is then stable for the following 6 weeks, then it starts to increase and stays above 300k events per day for most of the remaining weeks. The volume of events drops quite notably in the last 3 weeks of the observed period. As one would expect, there are many more events during weekends than weekdays.

b: DAILY ACTIVE USERS

The distribution of daily active users is reported in Figure 11, it can be seen that it is coherent with the number of events per day (see Figure 10).

Following this analysis, the data of both the first 2 weeks and the last 3 weeks is discarded, because of the differences in distribution of events in these periods highlighted before. The dataset is split in the following way:

- Training: the first 17 weeks (from week 2 to 18)
- Validation: the following 2 weeks (week 19 and 20)
- Testing: the last 2 weeks (week 21 and 22)

This splitting strategy allows to keep about the 80% of all the events for the training set, the 10% for the validation and another 10% for the test set. The characteristics of each partition are shown in Table 3.

2) EVALUATION PROTOCOL AND METRICS

The evaluation of the models is performed under the next-event prediction task [13]. For each event in a session,

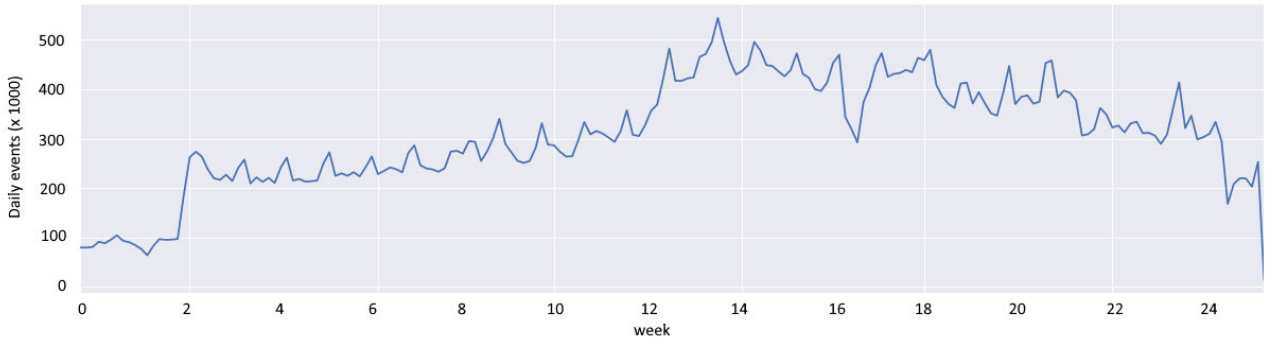


FIGURE 10. Number of events per day.

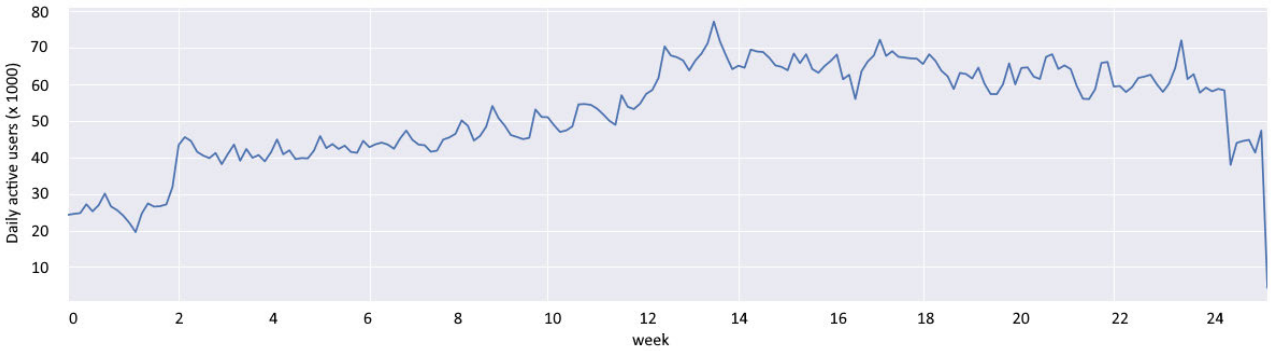


FIGURE 11. Number of active users per each day.

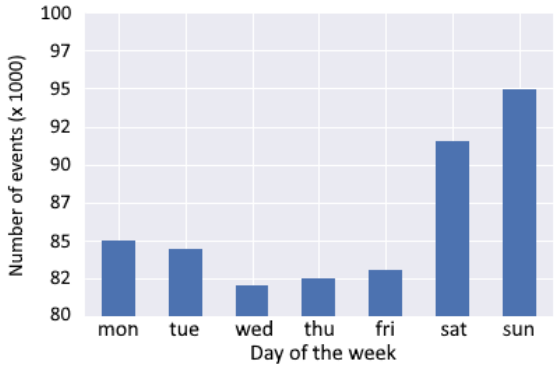


FIGURE 12. Number of events per day of the week.

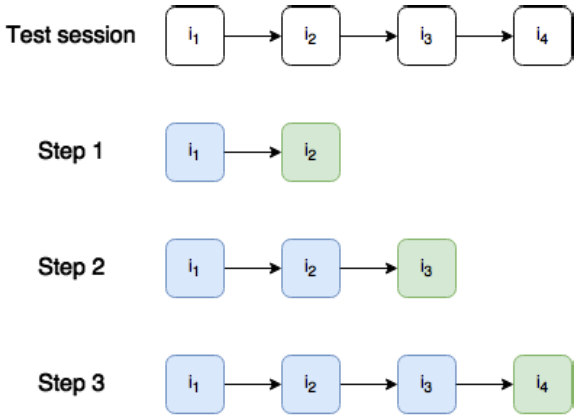


FIGURE 13. Steps of the evaluation under the next-item prediction scenario, given a test session.

the algorithm tries to predict the item involved in the next event in the session, generating recommendation lists of fixed length (i.e., the cutoff). Results are reported with traditional information retrieval metrics such as Recall and MRR at three different cutoffs, 1, 5 and 20. Formally, given the set of all the users $\mathcal{U}$, the set $\mathcal{P}_u$ of relevant recommendations for $u \in \mathcal{U}$, and the ordered set $\mathcal{R}_u^k$ of the top-k recommendations generated for $u \in \mathcal{U}$, the two metrics are defined as:

$$\text{Recall@}k = \frac{1}{|\mathcal{U}|} \sum_{\mathcal{U}}^{u} \frac{|\mathcal{P}_u \cap \mathcal{R}_u^k|}{|\mathcal{P}_u|}$$

$$\text{MRR@}k = \frac{1}{|\mathcal{U}|} \sum_{\mathcal{U}}^{u} \frac{1}{\min_{i \in \mathcal{P}_u \cap \mathcal{R}_u^k} \text{rank}(\mathcal{R}_u^k, i)}$$

where $\text{rank}(\mathcal{R}_u^k, i)$ is the function that returns the position of item i in the recommendation list $\mathcal{R}_u^k$ generated for user u, and k represents the cutoff at which the metrics are computed.

Figure 13 exemplifies the evaluation protocol for a given test session where each event is represented by an identifier. At each step of the evaluation, the sequence of events in blue is used to predict the immediately next item, in green. The evaluation metrics computed at each step are averaged to

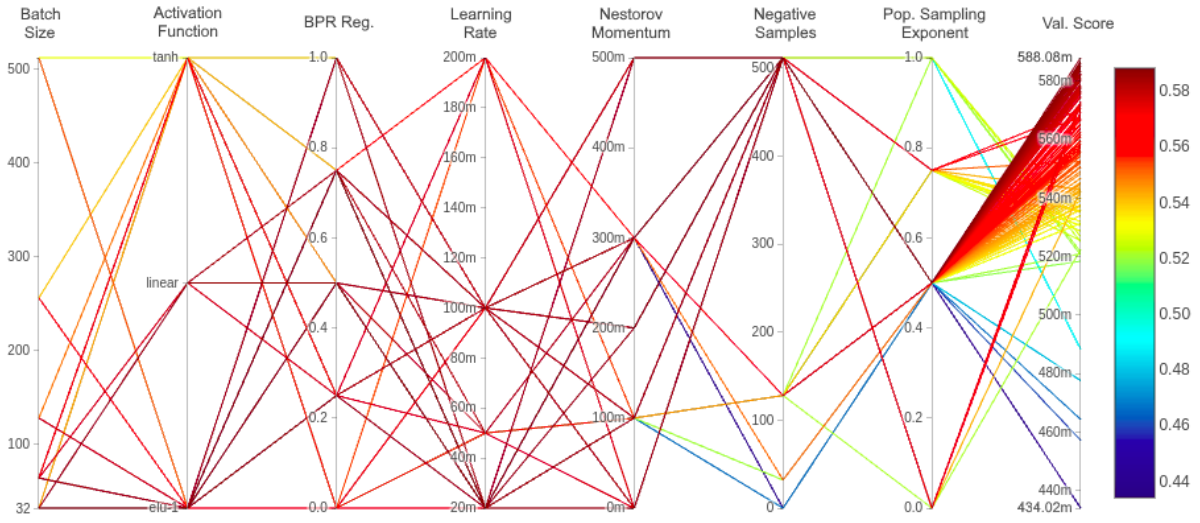


FIGURE 14. Parallel coordinate plot for GRU4Rec hyperparameter tuning. Recall@20 on the validation set is reported as metric score. A negative sampling probability distribution too similar to the item popularity distribution, an excessively small number of negative samples, and the hyperbolic tangent as activation function result in a degradation of accuracy. Small batch sizes, small learning rate and a high number of negative samples prove to be the best choices.

compute the performance of the recommender over the whole test session. The evaluation metrics of each session are then averaged.

A further experiment was done to take into account the availability window of each item. The evaluation was performed by excluding the items that were no longer available when the recommendation was required. However, there were not relevant differences between this and the original evaluation procedure, so, for brevity, only the results obtained without the availability filter are reported.

V. EXPERIMENTAL RESULTS

The results of the experiments are shown in the following sections. First, the hyperparameter optimization procedure is described in detail. Second, the accuracy results of the different algorithms presented in Section IV are reported and discussed. Finally, two analyses that assess the efficacy of session-aware and RNN-based models in exploiting within and across session information are performed.

A. HYPERPARAMETER TUNING

Recent works [39], [40] show that properly tuned simple baselines can outperform even more complex models in various recommendation tasks. These findings emphasize the importance of an appropriate selection of hyperparameters values, in order to get fair and reliable comparisons among different approaches. Indeed, it is known from literature that different values of hyperparameters can dramatically impact the performance of a model and using non-optimal values will lead to underestimating its quality, and resulting in a meaningless comparison. It follows that tuning a model's hyperparameters to obtain its best performance is a fundamental step in any evaluation procedure.

Two different optimization approaches were used for these experiments. For non-RNN based methods, a simple grid-search on the most important hyperparameters was employed. RNN based methods, instead, are deep learning approaches based on neural networks, that, compared to simpler techniques, have a wider range of fine grained hyperparameters to tune, where small variations of the values can lead to substantial changes in performance. For this reason, they are traditionally more sensitive to the choices of the hyperparameters and require a more thorough tuning process. Therefore, an extensive hyperparameter search on those models based on 100 iterations of random search was performed [41]–[43]. Due to the computational cost of such methods, the hyperparameter optimization was done on a subsampled training and validation dataset. Both were obtained by retaining only 50% of the users, chosen uniformly at random.

As expected, all methods exhibited some variability in the search space, with simpler models being more stable in their performance across the various combinations of the hyperparameters. In Figure 14 and 15 a detailed parallel coordinate plot of the hyperparameter tuning procedure of, respectively, GRU4Rec and H-GRU4Rec is shown. The models are evaluated on the validation set, using the Recall@20 as reference metric. In both scenarios, there are specific values of some parameters that degrade the performance of the model. For example, the selection of the hyperbolic tangent as activation function of the last layer usually leads to a poor accuracy of the model. The same occurs with small amounts of negative samples and a negative sampling probability too close to the popularity distribution. However, it is interesting to notice that overall the performance is affected by the global combination of the hyperparameters values instead of single parameters, except those mentioned above. This confirms that finding the optimal configurations for these two techniques is not an easy task and requires special attention.

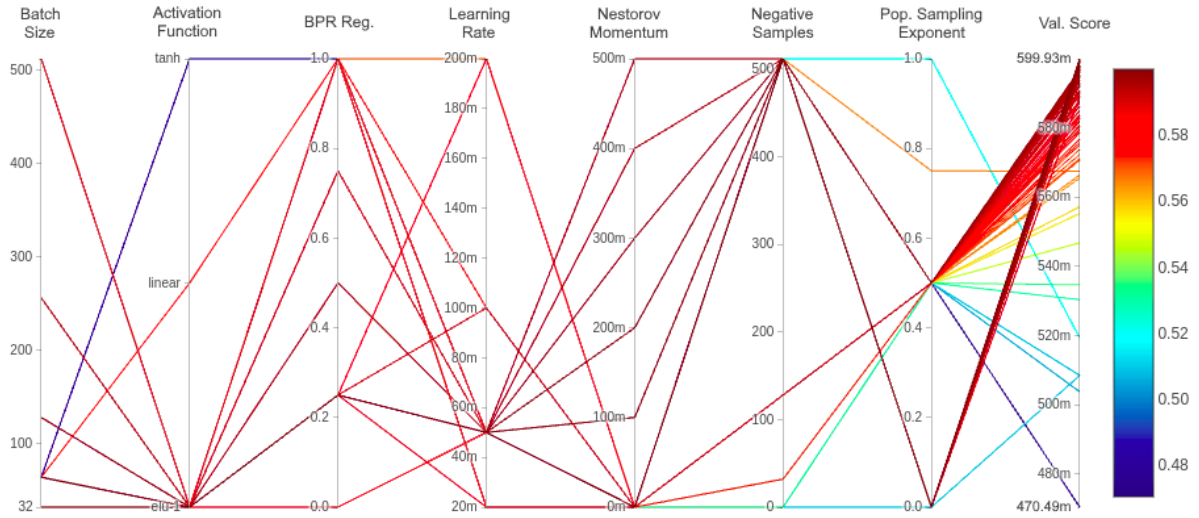


FIGURE 15. Parallel coordinate plot for H-GRU4Rec hyperparameter tuning. The metric reported is Recall@20 computed on the validation set. Similarly to GRU4Rec, a negative sampling probability distribution too similar to the item popularity distribution, an excessively small number of negative samples, and the hyperbolic tangent as activation function result in a degradation of accuracy. Also in this case, small batch sizes, small learning rate, ELU as activation function and a high number of negative samples prove to be the best choices.

It is also important to point out that the results of the hyperparameter search for RNN-based models is an extremely time-consuming process and the results are tightly bound to the training data, which can be seen as a strong limitation to the actual implementation in a production environment of such models. Nevertheless, showing the peak performance achievable by these complex models is a very valuable contribution, in the light of the possible future enhancements in hyperparameter search procedures that will potentially alleviate these issues.

B. NEXT-ITEM RECOMMENDATION ACCURACY

Table 4 shows the accuracy performance of the tested algorithms in the next-item recommendation task, according to Recall and MRR at 20. The results highlight that popularity-based baselines perform poorly on this dataset. User Pop, which has the highest degree of personalization among the popularities in the pool, since it looks at the past behavior of the considered user, is the best performing one. Anyway its performance is easily overtaken by all the other competing methods.

Collaborative ItemKNN outperforms simple semi-personalized baselines in both metrics as expected, confirming the results extensively reported in the literature [39]. However, this collaborative method falls short in next-item recommendation performance with respect to all the other algorithms. The main reasons for its poor performance are two. The first is that this algorithm considers only viewed items, which represent a small fraction of all user interactions, therefore a lot of information is lost. The second is that the evaluation protocol favors algorithms that can update their output frequently and dynamically, depending on user actions. Collaborative ItemKNN, instead, updates the user's profile and the respective recommendations only when a new view event occurs, and it does not adapt to user activity within the session.

Session BPR obtains much better results with respect to Collaborative ItemKNN thanks to its superior ranking capabilities, but it cannot attain the performance of Session ItemKNN. The main problem of the model is that it is not fully asymmetric, since the item representations are treated differently between training and testing. During training, this model considers sessions equivalently to users in a common matrix factorization approach and learns their latent representations. However, during the recommendation phase the model has to deal with sessions that were not available during the training process, and the model has not learnt their representations. It follows that new sessions must be represented with the average of the representations of the items that compose them [1], which is surely a sub-optimal solution.

Similarity-models are among the most competitive methods. Both of them largely outperform all the other non session-based recommenders and achieve an accuracy close to deep learning models' one. In particular, ContextKNN outperforms all the other item-based variants in terms of Recall, while attaining an MRR similar to Session ItemKNN. This result highlights the importance of the information contained in the examined session. Even if most of the sessions are composed by a small number of interactions, as shown in Section III-D, exploiting their content leads to sensible improvements in accuracy, also with simple methods like Session ItemKNN and ContextKNN. The last evident result is that RNN-based methods outperform all non-deep learning based alternatives, independently from the metric or the cutoff applied. However, the comparison between GRU4Rec and H-GRU4Rec is quite interesting and requires a more detailed analysis and discussion.

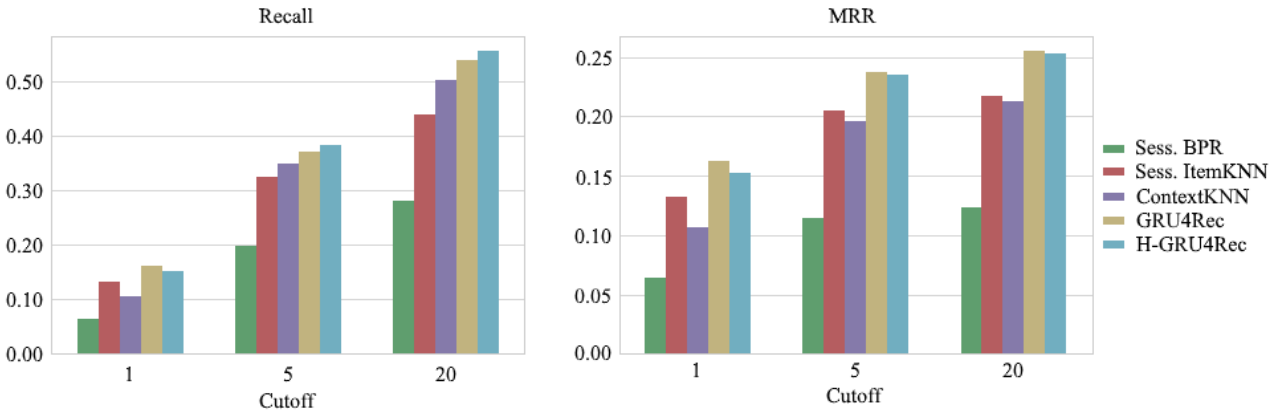


FIGURE 16. Accuracy performance of the different approaches in the next item recommendation task. On the horizontal axis, different cutoffs are reported. On the vertical axis, the absolute value of the specified metric is shown. Only competitive session-based approaches are included in the comparison.

TABLE 4. Comparison of the algorithms' performance @20 in the next-item recommendation task.

Method	Recall@20	MRR@20
GLOBAL POP	0.0111	0.0034
USER POP	0.0156	0.0036
SESSION POP	0.0130	0.0034
COLLABORATIVE ITEMKNN	0.0628	0.0157
SESSION BPR	0.2772	0.1227
SESSION ITEMKNN	0.4352	0.2174
CONTEXTKNN	0.4972	0.2121
GRU4REC	0.5389	**0.2548**
H-GRU4REC	**0.5545**	0.2531

In Figure 16, the best performing session-based methods at different cutoffs are confronted. Popularity based methods and Collaborative ItemKNN are not reported, because they did not obtain competitive results. Interestingly, H-GRU4Rec obtains superior Recall with respect to GRU4Rec, especially at longer cutoffs, while the latter gets a slightly better MRR overall and a slightly higher Recall@1. Combining these two results, it is clear that the correctly predicted items in the recommendation lists generated by GRU4Rec are usually ranked higher than those generated by H-GRU4Rec, while H-GRU4Rec is able to oftener include the desired item in its recommendations. A deeper investigation over this small margin between RNN-based methods is conducted in the following sections.

In summary, it is evident the advantage of session-based models over traditional recommendation methods (Popularity, Item-based collaborative filtering) in predicting the sequence of future user actions. A further step is represented by sequence-aware over sequence-free techniques, while there is not an evident winner between session-based and session-aware methods, since their accuracies are very close. In particular, deep learning methods outperform similarity based ones in this scenario by a good margin. However, the improvements come at a non negligible cost. From an industrial point of view, similarity based models present far less challenges than the RNN counterparts. They are simple to implement and maintain, the optimal point in the hyperparameter space is easier to find and their training time is quite low. Moreover, in their kNN variants, they also scale well with the amount of data available. Deep learning models, instead, are computationally expensive and time consuming and require an appropriate tuning of the hyperparameters. They also have multiple technical challenges that obstacle their implementation and their adaptation to large scale and online scenarios. However the large improvement in performance they provide can justify the engineering effort needed to introduce them in a production system.

C. WITHIN SESSION ANALYSIS

In Figure 17, the performance evolution of session-based techniques in function of the length of the session is shown.[2] In other words, how the accuracy of the recommendations changes as the user interacts with the system is analyzed, taking into account the evaluation methodology adopted, described in Section IV-E. Recall@20 is shown in the left plot, while MRR@20 is shown in the right one.

One of the most important and evident results is that Session ItemKNN, GRU4Rec and H-GRU4Rec have very similar trends. The Recall clearly improves as the session evolves. The MRR, instead, has a spike with very short sessions, composed by 2 or 3 interactions, then stabilizes for medium long sessions and finally slightly drops for very long sessions with more than 10 interactions. However, it is evident that deep learning models outperform Session ItemKNN independently from the session length, that confirms that these models can leverage the information gathered during the session by effectively representing the intent of the user.

On the other hand, ContextKNN obtains similar performance compared to complex RNN-based models at the very beginning of the session, but quickly loses track as the session evolves. The performance drop can be explained

[2]Session BPR is omitted, because it did not obtain competitive results compared to other session-based algorithms, as shown in Table 4.

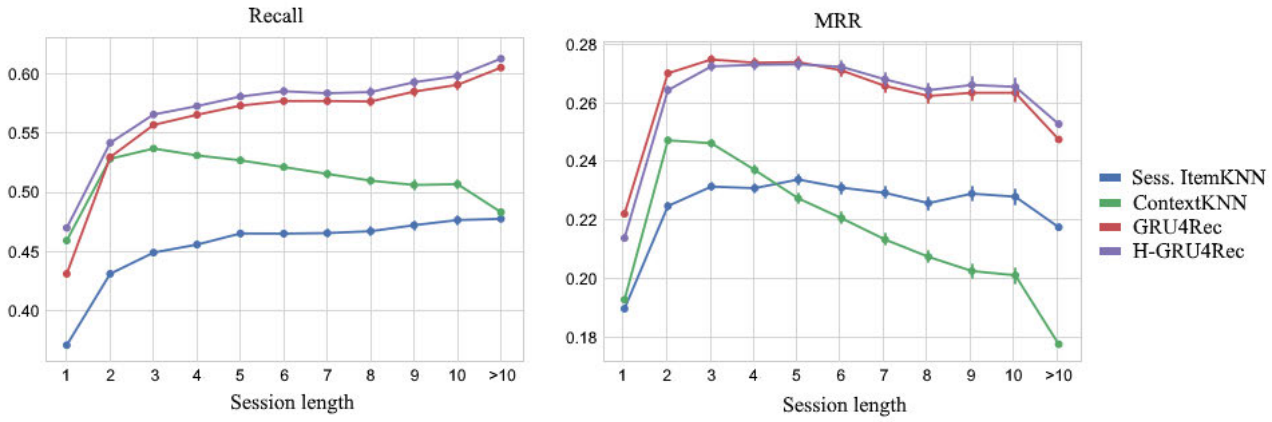


FIGURE 17. Performance @20 of session-based approaches per session length. Non competitive techniques are omitted.

with two insights. The first is that while short sessions are several and can easily repeat themselves, since the combinations are fewer, it is hard to find similar sequences for long sessions and consequently provide reliable and coherent recommendations. The second is that they do not take into account the positional importance of the interactions. Intuitively, the latest interactions before the required prediction should be the most important to take into account when providing a recommendation, an information not exploited by ContextKNN. This insight is confirmed by the performance of Session ItemKNN, that simply taking into account the item involved in the single latest interaction is able to outperform ContextKNN in longer sessions.

Another interesting detail that can be captured from the plots in Figure 17, is the difference in the Recall and MRR between GRU4Rec and H-GRU4Rec. It has been already highlighted in Figure 16 and Table 4 that while H-GRU4Rec has better Recall, especially at higher cutoffs, GRU4Rec outperforms its counterpart in terms of MRR. In Figure 17 the superiority of H-GRU4Rec over GRU4Rec in terms of Recall is evident and confirms the summary results. On the contrary, GRU4Rec shows better MRR for short sessions, while H-GRU4Rec performs better with longer ones, partially contrasting the results reported in Section V-B. It's worth noting that most of user sessions are short, as extensively discussed in Section III-D. Therefore, the traditional accuracy metrics are strongly skewed towards the beginning of each sequence, failing at capturing the difference between the two approaches with longer sessions.

D. ACROSS SESSION ANALYSIS

As last experiment, the difference in performance between GRU4Rec and H-GRU4Rec depending on the number of sessions in the user profile is analyzed. In Section IV-C, it has been pointed out that the main difference between GRU4Rec and H-GRU4Rec is that the latter is also able to infer and exploit the taste of a user, based on the previous sessions. The intent of this experiment is to assess if H-GRU4Rec is actually able to combine short and long term information about a user in order to improve its recommendation accuracy.

In Figure 18, the performance of the two deep learning based methods at different numbers of available sessions for a user is shown. Clearly the gap between H-GRU4Rec and GRU4Rec grows with the number of sessions, indicating that H-GRU4Rec can effectively leverage past user sessions for user profiling and that this type of information can benefit the performance of the model. However, it is interesting to notice that GRU4Rec achieves a better MRR on cold-start, or anonymous, users (i.e., users that have no referable past sessions in the system) while H-GRU4Rec has a slightly higher Recall. Another evident aspect highlighted by the plots is the descending trend in accuracy, accompanied with higher variance in the performance, that both the techniques show.

VI. LIMITATIONS AND FUTURE RESEARCH DIRECTIONS

This study addresses the entire pipeline required to build a recommendation model on a real-world dataset. The study is however done only on a specific case of a VoD service, hence the findings and guidelines reported should be considered in this perspective.

Regarding the data preprocessing phase, inferring sessions is a particularly important step. The analysis done to choose an appropriate idle time relied on the clear user behavior emerging from the clicks, i.e., short, mid and long-term actions. This was partially due to some highly regular patterns, e.g., 24-hour peaks, that may be of particular importance in the VoD domain but may not be present in others. In those cases, it may be more difficult to choose an appropriate idle time and more advanced strategies may be advisable. Furthermore, while the strategy based on idle time is simple and effective, it may be too rigid and unable to effectively capture more nuanced user behaviors. For example, it cannot capture rapid changes in user intent. If two sessions are separated by a shorter time distance than the idle time, they will be merged even if the user interacted with the system with a different intent. Due to this, for some domain it may not be an advisable strategy.

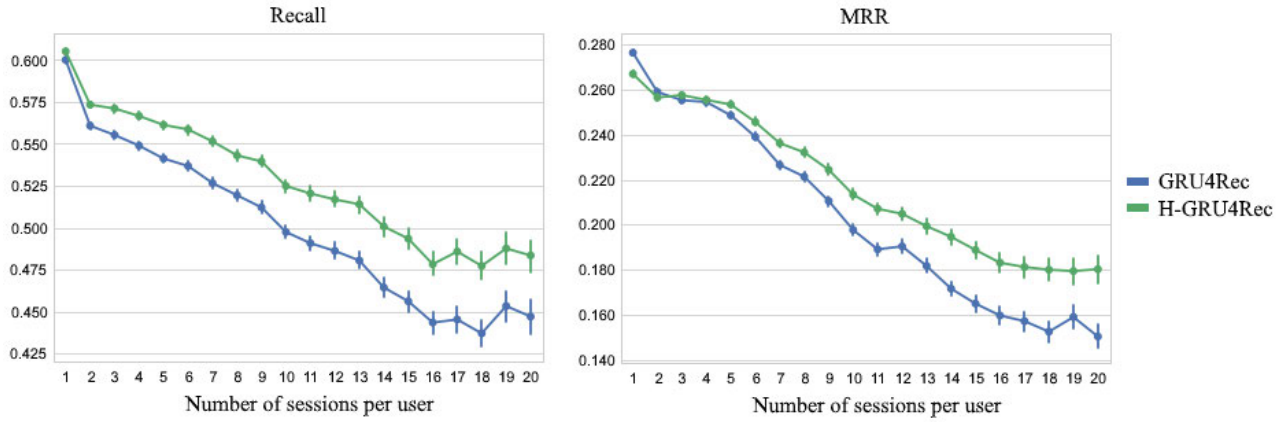


FIGURE 18. Performance @20 of GRU4Rec and H-GRU4Rec per number of sessions available for a user.

Choosing both the domain or the different strategy to infer sessions would also impact how sessions are filtered. For example, in the VoD domain it is expected the user will interact with a limited number of relatively long items while in other scenarios, e.g., music-on-demand or news recommendation, the number of items the user consumes is much higher and therefore so is the expected session length.

This work has shown a number of choices that can be made in the preprocessing of the data, all of them based on arguments pertaining the domain, user behavior analysis and the limitations of the available data, e.g., the so called *dangling paused/ended* that do not refer to a preceding *started* event. These choices affect the final dataset by approximating certain types of user behavior as well as removing data sources or inferring signals based on incomplete or noisy information. An open research question is to see how those choices affect the recommendation quality different families of models, e.g., session-free and session-aware. In particular, whether certain preprocessing steps could penalize or advantage a specific family of models. A particularly interesting question would be to assess whether currently used research dataset were preprocessed in such a way as to introduce unwanted biases that would affect the significance of the results obtained.

A further limitation of this study is that, while the importance of good data cleaning and processing from an industrial perspective is discussed at length, other aspects of recommendations in online systems like training times, memory consumption and recommendation time are not studied, and could be considered in future works.

Another open research question, from the point of view of the practical applicability of a model touches two particularly important and interconnected aspects pertain the robustness of the recommendation models to newly added data as well as their sensitivity to the hyperparameters. Although offline evaluation in research papers is usually done with a static dataset, in a real case the data is not static, rather it changes continuously over time as new interactions are collected. This is known as the *dataset shift* problem [44], [45]. It is desirable for a recommendation model to be robust to new data and able to use it effectively, without needing to be retrained often, which can be a very time consuming process. Another desirable property is for the models to be relatively robust to the choice of their hyperparameters. A model that is robust can be retrained on newly gathered data with the same hyperparameter configuration for a while before a new hyperparameter search step becomes necessary, while a model that tends to be unstable may need new hyperparameter searches very often, severely increasing the overall computational cost.

Other future directions might also include replicating this study in the same domain by other industrial actors or using different industrial datasets, thus enriching the recommenders' knowledge base in VoD recommendations. Moreover, this work can be compared with others performed in other domains, in order to establish similarities and differences in recommendations, user modeling, biases, and restrictions that might occur.

VII. CONCLUSION

In this work, a comprehensive study of the entire process between raw data collected from an online service and session-based recommendations has been presented.

First, the dataset used, which was collected from an industrial source for a period of 6 months, has been described. The industrial source is an Over-The-Top Media service that provides VoD content via the internet. The dataset contained user clicks, views, and ratings of VoDs but had no information on users' sessions; thus, the main challenges faced were the intense cleaning, processing, and feature engineering processes required. The effects of short, mid, and long-term actions in the dataset, and how they were used to infer sessions, have been presented and discussed, which is an important aspect of this work. Lastly, topics of interest regarding this dataset have been discussed, such as biases from existing recommenders and expiring items, i.e., those who could be interacted with at training time but could not be recommended afterwards due to expired licenses.

In the second part of the paper, the accuracy of several intent-based recommenders on the new dataset has been

compared. It has been shown that, for the next-item recommendation task, the best performing algorithms were two Deep Learning algorithms, namely GRU4Rec and H-GRU4Rec. This trend was also consistent with different metrics and cutoff lengths, and deeper analysis confirmed that GRU4Rec and H-GRU4Rec were the most accurate algorithms also for all session lengths. Lastly, it has been shown that H-GRU4Rec is better than GRU4Rec at capturing and exploiting user tastes based on previous sessions, combining short and long-term information about the user. It has been also highlighted that Recall and MRR dropped for both recommenders when the number of sessions per user increased, indicating that older information actually decreased recent recommendations' accuracy.

REFERENCES

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

[2] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," in *Proc. 11th ACM Conf. Recommender Syst.*, P. Cremonesi, F. Ricci, S. Berkovsky, and A. Tuzhilin, Eds., Como, Italy, Aug. 2017, pp. 130–137, doi: 10.1145/3109859.3109896.

[3] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 1–35.

[4] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, p. 19, Jan. 2016, doi: 10.1145/2827872.

[5] R. Turrin, M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi, "30music listening and playlists dataset," in *Proc. 9th ACM Conf. Recommender Syst.*, vol. 1441, P. Castells, Ed. Vienna, Austria: CEUR-WS.org, Sep. 2015, pp. 1–2. [Online]. Available: http://ceur-ws.org/Vol-1441/recsys2015_poster13.pdf

[6] F. B. Pérez Maurera, M. Ferrari Dacrema, L. Saule, M. Scriminaci, and P. Cremonesi, "ContentWise impressions: An industrial dataset with impressions included," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, M. d'Aquin, S. Dietze, C. Hauff, E. Curry, and P. Cudré-Mauroux, Eds., Ireland, Oct. 2020, pp. 3093–3100, doi: 10.1145/3340531.3412774.

[7] G. Adomavicius and J. Zhang, "Impact of data characteristics on recommender systems performance," *ACM Trans. Manage. Inf. Syst. (TMIS)*, vol. 3, no. 1, pp. 1–17, Apr. 2012.

[8] O. S. Shalom, S. Berkovsky, R. Ronen, E. Ziklik, and A. Amihood, "Data quality matters in recommender systems," in *Proc. 9th ACM Conf. Recommender Syst.*, H. Werthner, M. Zanker, J. Golbeck, and G. Semeraro, Eds., Vienna, Austria, Sep. 2015, pp. 257–260, doi: 10.1145/2792838.2799670.

[9] P. Kouki, I. Fountalis, N. Vasiloglou, X. Cui, E. Liberty, and K. Al Jadda, "From the lab to production: A case study of session-based recommendations in the home-improvement domain," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 140–149, doi: 10.1145/3383313.3412235.

[10] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-K gains for session-based recommendations," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, A. Cuzzocrea, J. Allan, N. W. Paton, D. Srivastava, R. Agrawal, A. Z. Broder, M. J. Zaki, K. S. Candan, A. Labrinidis, A. Schuster, and H. Wang, Eds., Turin, Italy, Oct. 2018, pp. 843–852, doi: 10.1145/3269206.3271761.

[11] S. Latifi, N. Mauro, and D. Jannach, "Session-aware recommendation: A surprising quest for the state-of-the-art," *Inf. Sci.*, vol. 573, pp. 291–315, Sep. 2021, doi: 10.1016/j.ins.2021.05.048.

[12] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, "A survey on session-based recommender systems," *ACM Comput. Surveys*, vol. 54, no. 7, pp. 1–38, Sep. 2022, doi: 10.1145/3465401.

[13] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," *User Model. User-Adapted Interact.*, vol. 28, pp. 331–390, Dec. 2018, doi: 10.1007/s11257-018-9209-6.

[14] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Empirical analysis of session-based recommendation algorithms," *User Model. User-Adapted Interact.*, vol. 31, no. 1, pp. 149–181, Mar. 2021, doi: 10.1007/s11257-020-09277-1.

[15] E. Zangerle, M. Pichl, W. Gassler, and G. Specht, "#nowplaying music dataset: Extracting listening behavior from Twitter," in *Proc. 1st Int. Workshop Internet-Scale Multimedia Manage.*, New York, NY, USA, 2014, pp. 21–26, doi: 10.1145/2661714.2661719.

[16] F. Wu, Y. Qiao, J.-H. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou, "MIND: A large-scale dataset for news recommendation," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*. Stroudsburg, PA, USA: ACL, Jul. 2020, pp. 3597–3606. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-main.331

[17] B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz, "The plista dataset," in *Proc. Int. News Recommender Syst. Workshop Challenge*, New York, NY, USA, 2013, pp. 16–23, doi: 10.1145/2516641.2516643.

[18] S. Eide, D. S. Leslie, A. Frigessi, J. Rishaug, H. Jenssen, and S. Verrewaere, "FINN.no slates dataset: A new sequential dataset logging interactions, all viewed items and click responses/no-click for recommender systems research," in *Proc. 15th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2021, pp. 556–558, doi: 10.1145/3460231.3474607.

[19] J. Liu, P. Dolan, and E. R. Pedersen, "Personalized news recommendation based on click behavior," in *Proc. 15th Int. Conf. Intell. User Interfaces*, New York, NY, USA, 2010, pp. 31–40, doi: 10.1145/1719970.1719976.

[20] E. Kirshenbaum, G. Forman, and M. Dugan, "A live comparison of methods for personalized article recommendation at forbes.com," in *Machine Learning and Knowledge Discovery in Databases*, P. A. Flach, T. De Bie, and N. Cristianini, Eds. Berlin, Germany: Springer, 2012, pp. 51–66.

[21] F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber, "Offline and online evaluation of news recommender systems at swissinfo.ch," in *Proc. 8th ACM Conf. Recommender Syst.*, New York, NY, USA, 2014, pp. 169–176, doi: 10.1145/2645710.2645745.

[22] C. A. Gomez-Uribe and N. Hunt, "The Netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 4, p. 13, 2016, doi: 10.1145/2843948.

[23] P. Symeonidis, A. Janes, D. Chaltsev, P. Giuliani, D. Morandini, A. Unterhuber, L. Coba, and M. Zanker, "Recommending the video to watch next: An offline and online evaluation at YOUTV.De," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 299–308, doi: 10.1145/3383313.3412257.

[24] S. Liu and Y. Zheng, "Long-tail session-based recommendation," in *Proc. 14th ACM Conf. Recommender Syst.*, R. L. T. Santos, L. B. Marinho, E. M. Daly, L. Chen, K. Falk, N. Koenigstein, and E. S. de Moura, Eds., Brazil, Sep. 2020, pp. 509–514, doi: 10.1145/3383313.3412222.

[25] T. Moins, D. Aloise, and S. J. Blanchard, "RecSeats: A hybrid convolutional neural network choice model for seat recommendations at reserved seating venues," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 309–317, doi: 10.1145/3383313.3412263.

[26] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé, III, and K. Crawford, "Datasheets for datasets," 2018, *arXiv:1803.09010*.

[27] M. Miceli, T. Yang, L. Naudts, M. Schuessler, D. Serbanescu, and A. Hanna, "Documenting computer vision datasets: An invitation to reflexive data practices," in *Proc. ACM Conf. Fairness, Accountability, Transparency*, New York, NY, USA, Mar. 2021, pp. 161–172, doi: 10.1145/3442188.3445880.

[28] S. Holland, A. Hosny, S. Newman, J. Joseph, and K. Chmielinski, "The dataset nutrition label: A framework to drive higher data quality standards," 2018, *arXiv:1805.03677*.

[29] R. Turrin, A. Condorelli, P. Cremonesi, and R. Pagano, "Time-based TV programs prediction," in *Proc. 1st Workshop Recommender Syst. Telev. Online Video ACM (RecSys)*, vol. 14, 2014.

[30] M. Quadrana, P. Cremonesi, and D. Jannach, "Sequence-aware recommender systems," *ACM Comput. Surv.*, vol. 51, no. 4, p. 66:1–66:36, 2018, doi: 10.1145/3190616.

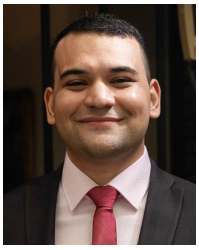
[31] T. Vasiloudis, H. Vahabi, R. Kravitz, and V. Rashkov, "Predicting session length in media streaming," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, N. Kando, T. Sakai, H. Joho, H. Li, A. P. de Vries, and R. W. White, Eds., Tokyo, Japan, Aug. 2017, pp. 977–980, doi: 10.1145/3077136.3080695.

[32] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, New York, NY, USA, Aug. 2017, pp. 306–310, doi: 10.1145/3109859.3109872.

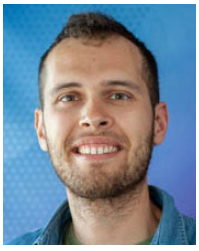
[33] J. Davidson, B. Livingston, D. Sampath, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, and M. Lambert, "The YouTube video recommendation system," in *Proc. 4th ACM Conf. Recommender Syst.*, Barcelona, Spain, X. Amatriain, M. Torrens, P. Resnick, and M. Zanker, Eds., 2010, pp. 293–296, doi: 10.1145/1864708.1864770.
[34] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. de Rijke, "A collaborative session-based recommendation approach with parallel memory modules," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, B. Piwowarski, M. Chevalier, É. Gaussier, Y. Maarek, J. Nie, and F. Scholer, Eds., Paris, France, Jul. 2019, pp. 345–354, doi: 10.1145/3331184.3331210.
[35] G. Bonnin and D. Jannach, "Automated generation of music playlists: Survey and experiments," *ACM Comput. Surveys*, vol. 47, no. 2, pp. 1–35, Jan. 2015.
[36] D. Jannach, L. Lerche, and I. Kamehkhosh, "Beyond 'Hitting the hits': Generating coherent music playlist continuations with the right tracks," in *Proc. 9th ACM Conf. Recommender Syst.*, H. Werthner, M. Zanker, J. Golbeck, and G. Semeraro, Eds., Vienna, Austria, Sep. 2015, pp. 187–194, doi: 10.1145/2792838.2800182.
[37] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl.*, D. Wu, M. Carpuat, X. Carreras, and E. M. Vecchi, Eds., Doha, Qatar, 2014, pp. 103–111. [Online]. Available: https://www.aclweb.org/anthology/W14-4012/
[38] E. Smirnova and F. Vasile, "Contextual sequence modeling for recommendation with recurrent neural networks," in *Proc. 2nd Workshop Deep Learn. Recommender Syst.*, B. Hidasi, A. Karatzoglou, O. S. Shalom, S. Dieleman, B. Shapira, and D. Tikk, Eds., Como, Italy, Aug. 2017, pp. 2–9, doi: 10.1145/3125486.3125488.
[39] M. Ferrari Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? A worrying analysis of recent neural recommendation approaches," in *Proc. 13th ACM Conf. Recommender Syst.*, T. Bogers, A. Said, P. Brusilovsky, and D. Tikk, Eds., Copenhagen, Denmark, Sep. 2019, pp. 101–109. [Online]. Available: https://github.com/MaurizioFD/RecSys2019_DeepLearning_Evaluation
[40] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Performance comparison of neural and non-neural approaches to session-based recommendation," in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 462–466, doi: 10.1145/3298689.3347041.
[41] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.
[42] T. Yu and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," 2020, *arXiv:2003.05689*.
[43] E. D'Amico, G. Gabbolini, D. Montesi, M. Moreschini, F. Parroni, F. Piccinini, A. Rossettini, A. R. Introito, C. Bernardis, and M. Ferrari Dacrema, "Leveraging laziness, browsing-pattern aware stacked models for sequential accommodation learning to rank," in *Proc. Workshop ACM Recommender Syst. Challenge*, P. Knees, Y. Deldjoo, F. B. Moghaddam, J. Adamczak, G. P. Leyson, and P. Monreal, Eds., Copenhagen, Denmark, Sep. 2019, p. 7:1–7:5, doi: 10.1145/3359555.3359563.
[44] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.
[45] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognit.*, vol. 45, no. 1, pp. 521–530, Jan. 2012.

CESARE BERNARDIS received the M.Sc. degree from Politecnico di Milano, with a thesis on feature weighting for item cold-start recommendation, where he is currently pursuing the Ph.D. degree. His research interests include machine-learning applications, recommender systems, and evaluation and the implementation of new techniques.

MAURIZIO FERRARI DACREMA received the Ph.D. degree *(cum laude)* in information technology focused on the assessment of reproducibility and methodological issues in neural recommender systems research from Politecnico di Milano, in 2020. He is a Postdoctoral Researcher at the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano. His research interests include recommender systems, with a particular focus on the independent evaluation and reproducibility of published research, and the application of currently available quantum computing technology for optimization and machine learning tasks. In 2019, he was awarded the Best ACM Long Paper Award for the paper "Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches."



FERNANDO BENJAMÍN PÉREZ MAURERA received the degree *(cum laude)* in computer engineering at Universidad Simón Bolívar, Venezuela, in December 2018. He is currently pursuing the Ph.D. degree in information technology with the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano. His undergrad thesis "Improving collaborative filtering techniques by the use of co-training in recommender systems" got an Outstanding Mark and received recognition from the Venezuelan Computing Society. From April 2018 to October 2019, he was a Software Engineer at Mahisoft Inc. His research interests include recommender systems, with a particular focus on industry recommenders, user behavior analysis, online evaluation, and evaluation and reproducibility of published research.



MASSIMO QUADRANA received the Ph.D. degree from Politecnico di Milano, Italy, with a thesis on algorithms for sequence-aware recommender systems. He was a Senior Scientist at Pandora Media LLC, where he worked on music recommender systems.

MARIO SCRIMINACI was born in Sicily, in 1984. He received the master's degree in computer science for intelligent systems from Università Degli Studi di Palermo, in 2009. He is the Chief Product Officer with Mostly AI. From 2006 to 2009, he was a Collaborator of the ICAR CNR. In 2010, he joined ContentWise as a Software Developer and a Data Scientist first and as a Product Manager in the last years. He has more than ten years of experience in applying machine learning and AI in B2B products.

PAOLO CREMONESI is a Professor of computer science engineering at Politecnico di Milano (Polimi), Italy, and Co-Founder of Moviri—one of the first and most-successful start-ups from the Politecnico di Milano Accelerator. He is an author of more than 200 papers and a co-inventor of five granted patents. His current research interests include the design and evaluation of machine-learning algorithms and recommender systems.