MobileRec: A Large-Scale Dataset for Mobile Apps Recommendation

M.H. Maqbool
University of Central Florida
Orlando, FL, USA
hasanmaqbool@knights.ucf.edu

Umar Farooq
Independent Researcher
Mountain View, CA, USA
ufarooq.cs@gmail.com

Adib Mosharrof
University of Kentucky
Lexington, KY, USA
amo304@g.uky.edu

A.B. Siddique
University of Kentucky
Lexington, KY, USA
siddique@cs.uky.edu

Hassan Foroosh
University of Central Florida
Orlando, FL, USA
hassan.foroosh@ucf.edu

ABSTRACT

Recommender systems have become ubiquitous in our digital lives, from recommending products on e-commerce websites to suggesting movies and music on streaming platforms. Existing recommendation datasets, such as Amazon Product Reviews and MovieLens, greatly facilitated the research and development of recommender systems in their respective domains. While the number of mobile users and applications (aka apps) has increased exponentially over the past decade, research in mobile app recommender systems has been significantly constrained, primarily due to the lack of high-quality benchmark datasets, as opposed to recommendations for products, movies, and news. To facilitate research for app recommendation systems, we introduce a large-scale dataset, called MobileRec. We constructed MobileRec from users' activity on the Google play store. MobileRec contains 19.3 million user interactions (i.e., user reviews on apps) with over 10K unique apps across 48 categories. MobileRec records the sequential activity of a total of 0.7 million distinct users. Each of these users has interacted with no fewer than five distinct apps, which stands in contrast to previous datasets on mobile apps that recorded only a single interaction per user. Furthermore, MobileRec presents users' ratings as well as sentiments on installed apps, and each app contains rich metadata such as app name, category, description, and overall rating, among others. We demonstrate that MobileRec can serve as an excellent testbed for app recommendation through a comparative study of several state-of-the-art recommendation approaches. The quantitative results can act as a baseline for other researchers to compare their results against. The MobileRec dataset is available at https://huggingface.co/datasets/recmeapp/mobilerec.

KEYWORDS

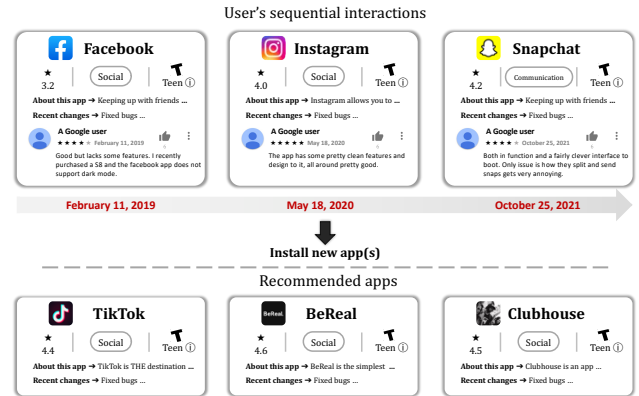Sequential Recommendation, GooglePlay Dataset, App Recommendation Dataset.

Figure 1: An Example of a sequence of the user activity. Based on past user interactions (e.g., app installations), the app recommendation system recommends new apps to install.

1 INTRODUCTION

Mobile apps have seen exponential growth in the last decade and over 5 billion users [8] utilize them for a variety of reasons, including social media, entertainment, news, productivity, and ride-sharing, among others. As a result of this boom, Google Play [16] and Apple App store [5] host more than 3.5 and 2.2 million apps, respectively [6]. The increasingly crowded app marketplaces pose a significant challenge for users to discover apps that align with their preferences effectively. Personalized app recommendations can relieve users' cognitive overload and improve the app installation experience. As illustrated in Figure 1, an app recommendation system has the capability to suggest new applications to users based on their previous app installations and interactions. Although Google Play and the App Store employ app recommendation techniques for suggesting apps to their users potentially leveraging user data collected internally, the research in app recommendation is almost nonexistent.

M.H. Maqbool, Umar Farooq, Adib Mosharrof, A.B. Siddique, and Hassan Foroosh

Table 1: Comparison of existing mobile apps datasets with MobileRec.

Dataset features	Top 20 Apps [3]	RRGen [15]	AARSynth [13]	Srisopha et al. [38]*	PPrior [14]†	MobileRec
Number of Reviews	200K	309K	2.1M	9.3M	2.1M	19.3M
Number of Apps	20	58	103	1,600	9,869	10,173
Number of App Categories	9	15	23	32	48	48
Multiple Reviews by a Single User	✗	✗	✗	✗	✗	✓
App Metadata	✗	✗	✓	✗	✗	✓
Reviews Rating	✓	✓	✓	✓	✓	✓
Review Text	✓	✓	✓	✓	✓	✓
Review Timestamp	✗	✗	✗	✓	✓	✓

* [38] is not publicly available. † [14] contains only negative user reviews.

Recommendation systems have demonstrated remarkable effectiveness in a wide range of domains, such as news [46], movies [10], products [42, 48], fashion [21], fitness [34], toys, beauty, CDs [18, 31, 33], to name a few [37]. The availability of datasets for specific domains that contain user-item interactions [1, 26, 30] have played a critical role in the development and improvement of recommendation systems. In earlier research items and users were typically represented using their unique identifiers, and their respective interactions, such as rating scores, were used to learn useful relationships that could aid in recommendation tasks. These relationships were typically learned using collaborative filtering techniques or similar methods [26]. Recently, incorporating sequential user interactions into the development of recommendation systems has led to significant improvements in their performance [23]. By taking into account the sequential patterns of user interactions with items [33] (e.g., the order in which they viewed or purchased items), these methods can better capture the user's preferences and provide more accurate recommendations [23]. Furthermore, the development of novel techniques such as recurrent neural networks (RNNs) [11, 19] and attention-based models [43] that effectively handle sequential data has shown exceptional performance in recommendation tasks [42, 48].

There are several notable datasets that focus on mobile applications, such as the Top 20 apps dataset [3], RRGen [15], AARSynth [13], Srisopha et al.[38], and PPrior[14], among others. However, Top 20 Apps and RRGen only contain a limited number of reviews, typically in the hundreds of thousands. Moreover, they only cover fewer than 100 apps from less than 20 categories. Despite providing larger datasets with millions of reviews from thousands of apps, datasets such as AARSynth, Srisopha et al., and PPrior are not amenable to building effective app recommendation systems, since they lack unique user identifiers. Furthermore, the dataset from Srisopha et al. [38] is not publicly available and PPrior only provides negative user reviews. A comparison of mobile app datasets is presented in Table 1. This work attempts to fill this research gap by providing a large-scale, rich, and diverse benchmark dataset, we call MobileRec, to facilitate researchers in developing app recommendation systems. To the best of our knowledge, *this is the only mobile app dataset that offers sequential and multiple interactions per user* with apps. We also present a comparison of our proposed dataset with the latest versions of the well-established recommendation datasets in various domains in Table 2.

MobileRec is a large-scale dataset constructed from users' activity on the Google Play Store, containing 19.3 million user interactions (i.e., user reviews along with rating) across 10K unique apps in 48 categories. It records the sequential activity of 0.7 million unique users, each of whom interacted with at least five apps. Every user interaction within our dataset consists of essential information, such as the user's rating, textual review, and the review date, pertaining to installed apps. Moreover, every individual app in MobileRec is equipped with abundant metadata, such as the app's title, category, long-form textual description, overall average rating, developer information, content rating, and additional pertinent details. Table 3 presents important features of the dataset. To stimulate research in app recommendation systems, MobileRec is on par with well-established and widely recognized datasets from various other domains. On top of app recommendation, our comprehensive dataset can offer significant insights and serve as a valuable resource for diverse research and data analysis pursuits, such as sentiment analysis, app development and optimization, market research, and fraud detection, among others.

We also show that MobileRec can serve as a valuable experimental testbed for app recommendation research through a comparative evaluation of several cutting-edge recommendation techniques. Specifically, we employed several existing general and sequential recommendation systems, including Pop (i.e., the popularity of items), SASRec [23], ELECRec [7], BERT4Rec [39], HGN [28], SINE [40], LightSANs [12], GRU4Rec [41], and GCSAN [47] using MobileRec. Our analysis of the results obtained by these methods on the MobileRec has revealed some intriguing trends. Notably, we observed that ELECRec [7] had performed significantly better than GRU4Rec [41] on the Amazon-Beauty dataset, achieving over 300% improvement in the Hit@5 metric. However, when tested on MobileRec, ELECRec exhibited a considerable drop in performance compared to GRU4Rec on the same metric. Similarly, we observe that LightSANs [12] had achieved a 3.91% and 2.65% increase in Hit@10 and NDCG@10 metrics, respectively, compared to SASRec [23]. However, on MobileRec, SASRec [23] outperforms LightSANs [12]. This signifies the inherent dynamism and fleeting nature of user-item interactions and user interests in the MobileRec dataset, which gives rise to complex user interaction history. This observation has important implications for state-of-the-art recommendation models. The numerical results presented in this work can also serve as a baseline for fellow researchers to evaluate their

Table 2: MobileRec's comparison with the latest versions of well-known recommendation datasets in different domains. We used the Amazon Reviews dataset from 2018, Yelp's 2022 version, and the latest release of ML-25M from 12/2019 to produce these statistics.

Datasets	Amazon Reviews	Yelp	ML-25M	MobileRec (Ours)
Total interactions (in millions)	233.1	6.99	25.0	19.3
Users with at least five interactions (in millions)	10.6	0.29	0.16	0.70
Items with at least 15 interactions (in thousands)	2072	77.58	20.59	10.17
Maximum number of interactions by a single user	446	3048	32202	256
Minimum number of interactions by a single user	1	1	20	5
Average number of interactions per user	5.32	3.52	153.81	27.56
Maximum number of interactions on a single item	13,560	7,673	81,491	14,345
Minimum number of interactions on a single item	1	5	1	20
Average number of interactions per item	15.45	46.49	423.39	1896.88

own results and objectively assess the quality and effectiveness of different approaches. Moving forward, numerous natural language processing techniques, such as advanced text representation methods and pre-trained language models [9, 35], have the potential to unlock new research avenues for app recommendations using MobileRec. These techniques hold great promise for enhancing the quality and effectiveness of app recommendation systems, and as such, are of particular interest to the research community.

Specifically, this work makes the following contributions:

- We present MobileRec, the most extensive collection of sequential user-app interactions to date, comprising over 19 million interactions across a diverse range of over 10 thousand distinct apps from Google Play, spanning all categories. Notably, this is the only mobile app dataset that features multiple interactions per user.
- Our experimental study investigates the dynamics of user-app interactions and demonstrates the practical utility of MobileRec by employing several state-of-the-art recommendation systems and establishing baseline results for the research community.

2 RELATED WORK

2.1 App Datasets and Recommendation

There are several existing datasets for user reviews of mobile apps as listed in Table 1. In an early effort to collect user reviews, Iacob and Harrison [20] collected 3,279 reviews for 161 mobile apps and analyzed feature requests from users. Khalid et al. [25] and [24] focused on iOS apps and prepared a dataset of 6,390 user reviews for 20 apps. McIlroy et al. [32] used 601,221 user reviews for 12,000 mobile apps to study negative reviews on app stores. Maalej and Nabil [29] collected a much larger dataset with 1.3 million reviews for 1,186 apps. These datasets focus on user complaints and user-developer dialogue understanding. Moreover, these datasets are not publicly available.

Top 20 Apps [3] is available on Kaggle, which contains 200 thousand reviews for 20 apps spanning 9 categories. This dataset provides rating scores and text for the reviews. RRGᴇɴ [15] contains more than 309 thousand reviews from 58 apps. Similar to the Top 20 Apps, RRGᴇɴ provides only rating scores and text of reviews. Both datasets do not provide app metadata, the timestamp of review, and a unique identifier for the user.

AARSʏɴᴛʜ [13] provides more than two million user reviews spanning over a hundred apps, including app metadata. Reviews of this dataset also miss out on key information similar to the above-mentioned datasets. The dataset by Srisopha et al. [38] includes over 9 million user reviews from 1,600 apps. This dataset has review timestamps, which can help to understand reviews in the context of the time period. However, this dataset does not include the user's unique identifier and app metadata. Please note that dataset by Srisopha et al. [38] is not publicly available.

More recently, the PPʀɪᴏʀ [14] dataset provided more than 2 million reviews for over 9 thousand apps spanning 48 categories from Google play. This dataset provides rating scores, review text, and timestamps of reviews. Nonetheless, user identifier on interactions (i.e., reviews) and app metadata is not provided. Furthermore, PPʀɪᴏʀ only provided negative user reviews (i.e., reviews with ratings 1 and 2 only). Hence, making PPʀɪᴏʀ not suitable for building robust app recommendation systems. In this work, we provide a larger dataset, called MobileRec, than all of the above-mentioned datasets for mobile apps, including each user's unique identifier and timestamps with interactions. Furthermore, each user has a least five interactions with apps and all the included apps have at least 15 interactions, which makes it an ideal testbed for mobile recommendation systems.

2.2 Existing Recommendation Datasets

Recommendation datasets are available across a diverse array of domains, encompassing everything from e-commerce platforms like Amazon, to the entertainment industry with datasets focused on movies, and even extending to online gaming platforms and beyond. These diverse datasets offer a wealth of opportunities for researchers to investigate and improve the effectiveness of recommendation systems across a broad range of use cases.

Aᴍᴀᴢᴏɴ Pʀᴏᴅᴜᴄᴛ Rᴇᴠɪᴇᴡs [18, 31, 33] is a large-scale dataset that contains 233.1 million reviews in the updated version of 2018. There are 29 categories in the Aᴍᴀᴢᴏɴ Pʀᴏᴅᴜᴄᴛ Rᴇᴠɪᴇᴡs dataset. Note that, an earlier version of this dataset which was released in 2014 had 142.8 million interactions and 24 categories. This dataset provides review text, rating, and helpfulness votes along with descriptions, category information, price, and brand, among others, as product metadata. Aᴍᴀᴢᴏɴ Pʀᴏᴅᴜᴄᴛ Rᴇᴠɪᴇᴡs dataset contains several categories including All Beauty, Books, CDs and Vinyl.

Table 3: Description of the various important features in the MobileRec **dataset.**

Feature	Description
uid	16-characters alphanumeric uid represents a user uniquely. It also anonymizes the user. Example: Aj0Sm6myfh6YN3Rn, 3pZUhksFIcLjEXtl, dvx0dqXTKtHUmY3O
app_package	Represents android package name of an application and uniquely identifies an application. Example: com.google.android.calculator, com.king.crash, org.wikipedia
app_name	Title of the app as displayed on Google Play. Example: Candy Crush Saga, MONOPOLY - Classic Board Game
app_category	The category of the app. Example: Entertainment, Finance, Productivity
review	Text review given by a user to an application
rating	Numeric rating the user has given to an application. Example: 5, 2, 1, 4
votes	The number of users who found this review helpful. Example: 1, · · · , 6, · · ·
date	The date of the specific user/item interaction, i.e., review date. Example: October 21, 2018, November 4, 2021, January 16, 2021
formated_date	The date of the specific user/item interaction. (review date in YYYY-MM-DD format). Example: 2018-10-21, 2021-11-04, 2021-01-16
unix_timestamp	The review date converted into unix timestamp. Example: 1.540094e+09 1.547788e+09, 1.610773e+09

Each of these categories contains thousands to millions of reviews. For example, `All Beauty` has 0.37 million reviews and 32K unique products, `Books` has 51 million reviews and 2 million products, and `CDs and Vinyl` contains 1 million reviews and 544K products. 5-core is a small version of AMAZON PRODUCT REVIEWS dataset that contains at least 5 user interactions, which makes it useful for building robust recommendation systems. Since AMAZON PRODUCT REVIEWS is a very large dataset, several of the existing works[7, 23] just employ some of the categories from the dataset such as `Books`, `Beauty`, `CDs`, and `Games` to benchmark their proposed methods.

YELP [4] dataset has several releases starting from 2018 to 2022. The latest version, YELP2022 has 1.9 million unique users, 150K distinct items, and 6.9 million reviews. Similarly, MovieLens [1] offers ML-25M [17], which comes with 25 million ratings and 1 million tag applications across 62K movies. The past versions of this dataset include ML-100K and ML-1M.

In addition to the large-scale datasets mentioned above, a wide variety of datasets from diverse domains have played a vital role in advancing research within their respective fields. In the following, we briefly discuss other recommendation datasets. STEAM contains reviews and game information crawled from stream [23]. BOOK-CROSSING [50] and GOODREADS [44, 45] provide user-item interactions where interaction type is rating. DIGINETICA[49] contains 0.2 million users, 184K items and about 1 million interactions, where interaction type is user-clicks. This dataset has been compiled from user sessions extracted from e-commerce search engine logs. TWITCH [36] offers 474 million interactions by 15 million users on 6 million items, and the interaction type is user-click. These datasets have enabled the development of several recommendation systems [10, 18, 21, 31, 33, 34, 37, 42, 48] in their respective domains. We expect that MobileRec plays a similar role to ignite research in building robust app recommender systems.

3 MOBILEREC **DATASET**

3.1 Dataset Construction

Google Play [16] serves as the default app store for android users to install the apps and express their opinion about the apps. Consequently, it hosts an enormous amount of user interactions through user reviews. The user reviews are dynamically loaded upon scrolling the web page. Therefore, traditional web scraping tools cannot facilitate downloading of large-scale user reviews data. To automate the scrolling of the page and perform other click events (e.g., longer reviews are not displayed fully by default), we used Selenium Web-Driver [2].

To obtain the data, the first step was to collect the package names (i.e., unique identifiers) of all the apps directly accessible from Google Play by navigating through all the app category pages as well as top charts. Then, we used this information to download app metadata. The metadata about an app included details such as the app name, developer, category, and textual description, among others. The user reviews associated with each app were extracted by accessing the user review section of the app, recursively scrolling the page, and extracting the text of the review along with the user's rating of the app, user's information (i.e., anonymized later on), and timestamp until no more reviews were available for the given app. Table 3 presents important features of the dataset. We used several filters and sorting mechanisms (e.g., newest, rating) to allow for downloading the maximum number of user reviews.

To ensure the quality of the data, several checks were put in place to eliminate duplicate entries, incorrect information, and other errors that may have occurred during the crawling process. We removed the duplicates from the data before processing the data further for converting it to a 5-core. We also anonymized the users by introducing a 16-character alphanumeric *uid*. We only
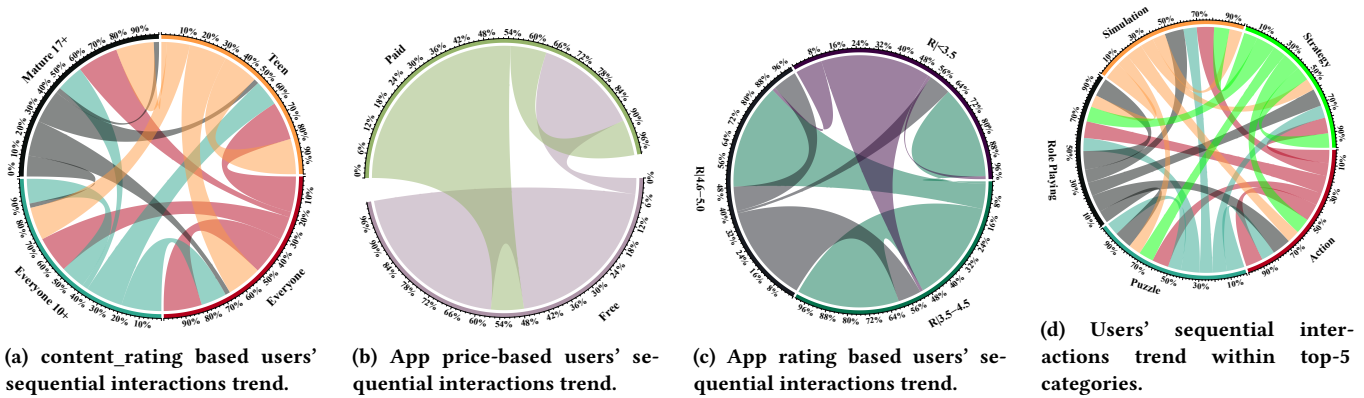
(a) content_rating based users' sequential interactions trend.

(b) App price-based users' sequential interactions trend.

(c) App rating based users' sequential interactions trend.

(d) Users' sequential interactions trend within top-5 categories.

Figure 2: Users' sequential interactions trend with respect to conten_rating, price tier, rating and within top-5 categories.

kept those users who have more than 5 reviews. The items having less than 15 reviews were removed from the final dataset. The end result was a large-scale 5-core dataset of user reviews (i.e., timestamped interactions) for apps available on Google Play, providing valuable insights into user opinions and experiences with various apps. Table 1 and 2 provide important statistics about the dataset.

3.2 Dataset Features

We introduce various features (also presented in Table 3) in the dataset which will be helpful in discussing the dataset trends. The *uid* is the unique 16-character alphanumeric id of a user, which also serves as anonymization of the users, since MobileRec does not provide the actual user IDs for privacy reasons. The *review* is the actual text review by a user. The *app_package* is the unique android package name of an application. The *rating* represents the actual rating given to an application (identified by app_package) by a user. The *formated_date* is used for sorting the user interactions in chronological order. For further reference, Table 3 can be consulted.

3.3 Dataset Analysis

In this section, we present various trends in the MobileRec dataset, e.g., how do users migrate from one category to the other with respect to their review-based interactions, how do users' behavior evolve with respect to application pricing, how do users migrate from application to application with respect to application's content rating (i.e., Mature 17+, Teen, Everyone 10+, and Everyone).

We capture an interaction-based snapshot of users' dynamic behavior in Figure 2(a), 2(b), 2(c) & 2(d). We consider the first two user interactions for capturing the migration behavior. For example, in Figure 2(a), it can be noticed that the majority of users with their interaction in Mature17+ categories at a given timestep t, ended up interacting with Everyone and the Teen content-rated apps in their interaction at timestep $t + 1$. Similarly, an expected trend is revealed in Figure 2(b) which illustrates price-based sequential user interactions. Figure 2(b) points out that very few users who interact with free apps at timestep t, interact with paid apps at their timestep $t + 1$. Conversely, numerous users who interact with paid apps, quickly migrate to free apps. Furthermore, it can be noticed from Figure 2(c) that users who interact with top-rated apps
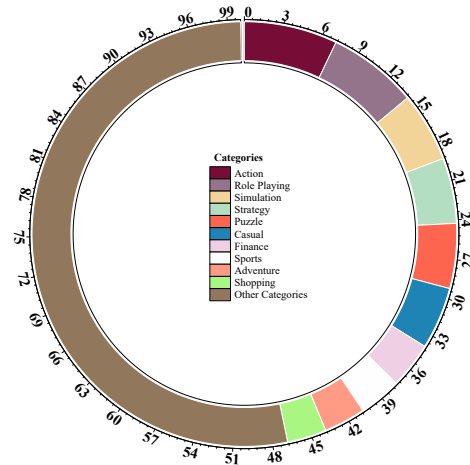

Figure 3: Top-10 categories with respect to the number of reviews. The numbering around the dial depicts the percentage, e.g., apps belonging to the action category add up to ≈ 7% of total reviews in the MobileRec **dataset.**

mostly do not compromise on app quality (i.e., the overall average rating of the app), which is depicted by very low migration to low-rated apps by those users. It is also clear that users, with a liking for highly rated apps, may migrate to apps with rating scores between *3.5-4.5*. A very dynamic user migration pattern can be observed in Figure 2(d). Users migrate from one category to another among the top-5 categories quite dynamically. This highly fluctuating pattern depicts the complexity of modeling users' behavior in recommender systems, especially in cold-start settings such dynamic behavior becomes very challenging. It is important to note that MobileRec has 48 categories while in Figure 2(d), we show only top-5 categories with respect to the number of reviews. This dynamically fluctuating migration pattern can be even more complex if all the categories are considered.

Figure 3 summarizes the number of reviews in the top-10 categories. It can be observed that the Action category has the most number of reviews among all the 48 categories in the MobileRec dataset. It is also interesting to note that the top-10 categories
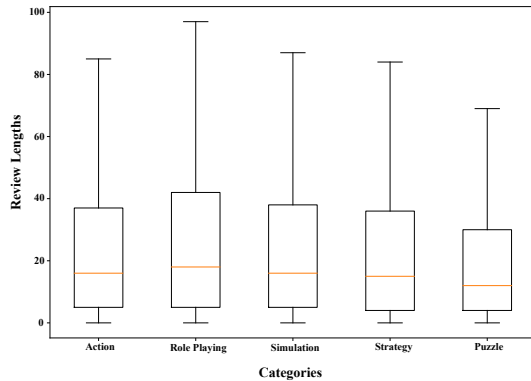
Figure 4: Review length for Top-5 categories based on the number of reviews.

amount to roughly 50% of the reviews in comparison with the rest of the categories.

Figure 4 summarizes the review length in the top-5 categories: `Action`, `Role Playing`, `Simulation`, `Strategy`, and `Puzzle`. We turn off the whiskers for better visibility in the box plot in Figure 4, but we provide some statistics about the outliers in each category. For example, in the `Action` category, there are $1,373,484$ total reviews with $10,357$ reviews having more than 100 words. Hence, the outliers amount to nearly 0.754% of the total reviews in the `Action` category. Similarly, in the `Role Playing` category, there are $1,321,861$ reviews in total with $12,187$ having more than 100 words giving rise to 0.922% outliers. The `simulation` category has $1,023,366$ reviews with $8,056$ having more than 100 words, which is 0.787% of the total reviews in the category. The `Strategy` category has 0.771% reviews with more than 100 words, which amounts to 7430 reviews out of $963,360$ total reviews belonging to this category. Finally, the `Puzzle` category has $954,909$ total reviews with $4,832$ reviews having more than 100 words which are a marginal 0.506% of the total reviews in the category. From Figure 4, it can be noticed that most of the reviews are less than 100 words long in the top-5 categories.

3.4 Additional Usage Scenarios

On top of building recommender systems, this large-scale dataset can be used for a variety of purposes. From app development and optimization to market research and fraud detection, the insights provided by this data can help businesses and researchers make informed decisions and create more effective strategies. In this work, we focus on providing baseline results for a wide range of recommender systems and leave other uses of this dataset to the research community.

As with any dataset, there are potential bad use cases for this dataset. As such, the data does not contain any information about the users. Specifically, we de-identified and anonymized data to prevent individual identification. For completeness, we consider it important to highlight potential bad uses that could arise. For example, a company could pay for fake reviews or ratings to artificially boost their app's standing, or they could manipulate the app metadata to make their app be recommended more often by a certain recommender system. Similarly, attackers could use the

data to identify apps more favorable to be recommended and then create fake versions that include malicious code.

On balance, we believe that this dataset is more useful than the potential harm. For example, this dataset has the potential to provide many useful insights and benefits for individuals and organizations. Specifically, we expect that this dataset will inspire research and development of app recommender systems, among other good uses.

4 BASELINES

We consider a wide range of recommender systems for benchmarking and establish baseline results for the MobileRec dataset.

4.1 General Recommendations

Pop. It is a simple popularity-based model. In this model, the popularity of items is recorded, and the most popular items are recommended to the user. For example, utilizing this model on Youtube, most watched videos will be recommended to users.

4.2 Sequential Recommendations

SASRec [23]. Markov chains assume that a user's next action can be determined by their recent actions (just one or few recent actions). On the other hand, RNN-based methods can consider long-term user-item interaction history for uncovering the hidden interests of a user. SASRec combines the best of Markov Chain (MC) based methods and Recurrent Neural Network (RNN) based methods in a unified design by employing the attention mechanism. MC-based approaches can be effective where the sparse dataset is involved, while RNN-based approaches are good for situations involving denser datasets. SASRec proposes to balance the power of MC-based approaches and RNN-based approaches to get the best of both worlds.

ELECRec [7]. Next item prediction task is generally modeled as a generative task. ELECRec proposes to employ a discriminator in the recommender system for the next item prediction task. The discriminator is responsible for deciding if the next sampled item is a true target item. A generator G and discriminator D are trained jointly. The generator G is trained to generate high-quality samples for the discriminator.

BERT4Rec [39]. Building upon the observation that sequential neural networks are sub-optimal and limited. This sub-optimality can be attributed to restrictive unidirectional (left to right) encoding of the user's behavior and their assumption is that the user-interaction sequence is ordered. BERT4Rec is a sequential recommendation model which employs the bidirectional self-attention for encoding the user's interaction sequence. This baseline adopts the cloze objective to the sequential recommendation and predicts the masked items by joint conditioning on their bidirectional context.

HGN [28]. Considering the importance of recent chronological user-item interaction, the hierarchical gating network proposes a method to capture both long-term and short-term user interests by integrating Bayesian Personalized Ranking (BPR). The proposed method consists of a feature gating module, item gating module, and item-item product module. The gating module is responsible to decide which features are to be forwarded to downstream layers from the feature and instance layers. The item-item product module

captures the item relations between items that the user interacted with in the past and will interact with in the future.

SINE [40]. Realizing the importance of multiple conceptually distinct items in a user's behavior sequence, SINE suggests that capturing a unified embedding for a user's behavior sequence is affected primarily by the most recent user interactions. Building upon this observation, SINE proposes to use multiple embeddings to capture various aspects of a user's interaction behavior. Since the concept pool can be large, SINE has the ability to infer a sparse set of concepts from the large concept pool. Having multiple embeddings, an interest aggregation module is employed for predicting a user's current intention, which is also used for modeling next-item prediction.

4.3 Transformer Based Recommendation

LightSANs [12]. Self-attention networks (SANs) are limited due to quadratic complexity, and vulnerability to over-parameterization. Modeling inaccuracy in sequential relations between items is also a limiting factor in SANs because of implicit position encoding. To this end, LightSAN proposes a low-rank decomposed self-attention network. Low-rank decomposed self-attention projects users' historical items into a few latent interests. Item-to-interest interaction is used to generate the context-aware representation.

4.4 Session Based Recommendation

GRU4Rec [41]. This baseline uses an RNN-based method for the session-based recommendation. Data augmentation and method to account for the distribution shift in the input data is employed in the proposed technique.

4.5 GNN Based Recommendation

GCSAN [47]. This method proposes a graph-contextualized self-attention model which employs both graph neural networks and self-attention networks. In GCSAN, rich local dependencies are captured with graph neural networks, while long-range dependencies are captured with the self-attention network. Each session is represented as a combination of global preference and the current interest of the session.

5 EXPERIMENTAL SETUP

5.1 Experimental Settings

For sequential recommendation baselines, we keep a consistent batch size of 4096 and the maximum interaction length to be 50. We employ a leave-one-out strategy for validation and testing, and the full item set is used for evaluation. We use early stopping patience of 10. We consider the sequential recommendation task as a multiclass classification task and use the cross-entropy loss for training the models. SASRec [23] is trained with adam optimizer and a learning rate of 0.001. The number of layers and attention heads is 2. The dropout rate is 0.5 and gelu is used as an activation function. HGN [28] and SINE [40] are trained with embedding size 64, learning rate 0.001, and adam optimizer. LigthSANs [12] uses the latent interest dimension to be 5, 2 attention heads and 2 transformer layers. Training is done with a learning rate of 0.001 and adam optimizer. GCSAN [47] has 2 transformer encoder layers,

2 attention heads, 64 is the hidden state features size, feed-forward layers' hidden size is 256, weight is set to be 0.6 and the number of layers in graph neural network is 1, adam is used for optimization with a learning rate of 0.001. GRU4Rec [41] is trained with 1 layer with an embedding size of 64, hidden size of 128, and dropout to be 0.3. We employ RecBole [49] for establishing the baselines for MobileRec.

5.2 Evaluation Metrics

We employ standard metrics, Hit@K and NDCG@K where $K \in \{1, 5, 10, 15, 20\}$, as our evaluation metrics for the benchmark methods. Hit@k considers the number of times the predicted item appears in the top K list and can be represented as in [27]:

$$HR@K = \frac{1}{M} \sum_{u=1}^{M} \mathbb{1}_{R_u \leq K} = \sum_{R=1}^{N} W_R \cdot \mathbb{1}_{R \leq K}$$

$$W_R = \frac{1}{M} \sum_{u=1}^{M} \mathbb{1}_{R_u=R}$$

$\mathbb{1}_X$ represents the indicator random variable, M are the total number of users and $N = |I|$ are the total items. R is the integer rank position of an item in the range $[1, N]$. R_u is the rank of item i_u among I items, for user u. W_R captures the users with item i_u at position R.

$NDCG@K$ can be represented as [22, 49]:

$$\frac{1}{|U|} \sum_{u \in U} \left(\frac{1}{\sum_{i=1}^{\min(|R(u)|,K)} \frac{1}{\log_2(i+1)}} \sum_{i=1}^{K} \delta(i \in R(u)) \frac{1}{\log_2(i+1)} \right)$$

All the item set is considered for ranking the prediction.

6 RESULTS AND DISCUSSION

Table 4 presents the performance analysis of various baselines on MobileRec. Next, we discuss these results and provide further details. As discussed earlier, Pop is a popularity-based model which relies on the popularity of items. Pop is the most naive model and, as expected, its performance is the worst compared to almost all of the other baselines. SASRec had previously reported better results than Pop according to Hit@10 and NDCG@10 performance metrics. In SASRec, 100 negative items were randomly sampled, and Hit@10 and NDCG@10 was calculated against these 101 items, including the ground truth. Using MobileRec, we employ a full item set for evaluation, which is a stricter evaluation criterion for recommendation systems. We observe that on all the performance metrics, SASRec outperforms Pop except Hit@1. For example, SASRec manages to get 19.86% improved Hit@10 in comparison with pop. On the same lines, a 16.34% performance improvement is observed in Hit@15. Similarly, SASRec achieves 15.23% improved Hit@20 in comparison with Pop on MobileRec.

Considering NDCG metric, SASRec had reported 41.37% improved NDCG@10 compared to pop, when evaluation is done on Beauty. On MobileRec, we observe only 14.28% improvement in NDCG@10 compared to pop. We believe that this difference in the improvement on NDCG@10 between SASRec and Pop on these two datasets (Beauty and MobileRec) can be explained by keeping in view the comparative dataset statistics of the Beauty and

M.H. Maqbool, Umar Farooq, Adib Mosharrof, A.B. Siddique, and Hassan Foroosh

Table 4: Performance Analysis of various baselines on MobileRec. **Baselines belonging to various categories, such as General baselines (e.g. Pop), Sequential baselines (e.g. SASRec, ELECRec, BERT4Rec, HGN, SINE), Session-based baselines (e.g. GRU4Rec GCSAN), Graph Neural Network based baselines (e.g. GCSAN), Transformer based baselines (e.g. LigthSANs).**

Method ↓ Metric →	Hit@1	Hit@5	Hit@10	Hit@15	Hit@20	NDCG@5	NDCG@10	NDCG@15	NDCG@20
Pop	0.0027	0.0086	0.0151	0.0208	0.0256	0.0056	0.0077	0.0092	0.0103
SASRec	0.0026	0.0098	0.0181	0.0242	0.0295	0.0061	0.0088	0.0104	0.0117
ElecRec	0.0020	0.0094	0.0174	0.0237	0.0293	0.0056	0.0082	0.0098	0.0112
Bert4Rec	0.0024	0.0083	0.014	0.0183	0.0221	0.0054	0.0072	0.0083	0.0092
HGN	0.0012	0.0054	0.0096	0.0132	0.0165	0.0033	0.0046	0.0056	0.0064
SINE	0.0022	0.0087	0.0163	0.0228	0.028	0.0054	0.0078	0.0095	0.0107
LightSANs	0.0024	0.0102	0.0172	0.0227	0.028	0.0062	0.0085	0.0099	0.0112
GRU4Rec	0.0021	0.0086	0.0153	0.021	0.0261	0.0053	0.0074	0.0089	0.0102
GCSAN	0.0024	0.0094	0.0161	0.0214	0.0266	0.0059	0.0081	0.0095	0.0107

MobileRec. There are $52,024$ users in Beauty while MobileRechas $700,111$ users. Secondly, Beauty dataset has $57,289$ items compared to MobileRec which has $10,173$ apps. Beauty has $0.4M$ interactions, with 7.6 average interaction per user and 6.9 average interaction per item. Whereas MobileRec has $19.3M$ interactions with 27.56 interactions per user and $1,896.88$ interactions per item (i.e., app). For example, MobileRec has 27.56 interaction per user, which is more than 200 times more interactions per user than Beauty. Likewise, MobileRec also has a considerably larger number of average interactions per item in comparison with Beauty. Keeping these factors in view, MobileRec presents a more dynamic recommendation scenario. The high degree of user and item interactions in MobileRec can be attributed to being the reason for the comparatively smaller improvement in NDCG@10 by SASRec compared to the improvement SASRec had reported over Pop on Beauty. Secondly, since we consider a full item set for evaluation compared to the ranking strategy by SASRec, this may also be a potential reason for smaller performance gains by SASRec in comparison with pop. On Beauty, SASRec reported better results than GRU4Rec. We observe the same pattern of SASRec outclassing the GRU4Rec when training and evaluation are performed on MobileRec.

On Beauty dataset, ELECRec reports better Hit@5, Hit@10, NDCG@5 and NDCG@10 than pop. We notice the same pattern when training and evaluation are done on MobileRec with Hit@1 to be the only exception, where Pop performs better than ELECRec. We assume that a very dynamic user-item interaction history might be the reason. Similarly, ELECRec had reported 329.87% improved Hit@5 and 242.04% improvement in Hit@10 compared to GRU4Rec on Beauty. Moreover, for NDCG@5 and NDCG@10 metrics, ELECRec achieved 412.12% and 331.38% improvements over GRU4Rec, respectively. Focusing on the performance improvements obtained by ELECRec over GRU4Rec when training and evaluation are done on MobileRec, ELECRec manages to get a 9.30% and 13.72% improvement in Hit@5 and Hit@10, respectively. Similarly, 1.88% and 10.81% improvement is observed in NDCG@5 and NDCG@10, respectively. This observation also points toward the highly dynamic user-item interaction involved in MobileRec. Both models struggle to achieve high performance, which results in diminished performance gaps between the two approaches as compared to the performance gaps reported in ELECRec. There are over 19 million interactions in MobileRec while only 0.4M user-item interactions

are there in Beauty. This high level of dynamism in user-item interactions pushes the models towards lower performance gains. Considering this analysis, MobileRec should be a valuable addition to the existing volume of recommendation datasets.

ELECRec also shows the comparative gains over SASRec on Beauty. First, we will cover the relative performance gains ELECRec manages to get over SASRec with respect to Hit@5, Hit@10, NDCG@5 and NDCG@10 on Beauty dataset. After that, we will discuss the performance improvements that ELECRec achieves over SASRec when the training and evaluation dataset is MobileRec. Starting off with Hit@5 and Hit@10, ELECRec reports 83.59% and 59.47% improvement over SASRec. Similarly, 103.61% and 84.11% improvement is reported by ELECRec over SASRec with respect to NDCG@5 and NDCG@10, respectively. Now, let us discuss the performance comparison between ELECRec and SASRec when training and evaluation are performed on MobileRec. It can be observed that SASRec performs better than ELECRec on MobileRec in Hit@5, Hit@10, NDCG@5, and NDCG@10 metrics. The quantification of the performance improvement of SASRec over ELECRec in Hit@5, Hit@10, NDCG@5, and NDCG@10 yields 4.25%, 4.02%, 8.92%, and 7.31%, respectively. We believe that ELECRec employs a generator that is trained with an NLP task, while the discriminator is responsible for discerning if the item is the next rightful (real) item in the sequence or a fake next item. The discriminators discerning ability depends upon the quality of samples generated by the generator. Since MobileRec presents as highly dynamic user-item interaction sequences with fleeting interests shown by users, the generator might have difficulty generating high-quality training samples. Since the discriminators' ability to capture the true item correlation depends upon the quality of samples generated by the generator, low-quality samples may lead to diminished performance gains on the discriminator's part.

It is worth recalling Figure 2 that provides an interesting snapshot of the dynamic user-item interaction sequences. For example, it can be noticed from Figure 2(b) that a considerable percentage of users migrate from paid applications to free applications. A similar dynamic migration pattern is evident from Figure 2(d) which depicts the user migration among top-5 categories. We think that with 19 million user-item interactions having such a dynamic temporal interaction pattern may present a challenge for the generator to learn high-quality sample generation. Moreover, ELECRec reported

its comparative performance to BERT4Rec. ELECRec outperformed BERT4Rec in the original paper on all the metrics on Beauty [7]. Specifically, ELECRec had reported a 100.85% and 61.06% performance gain over BERT4Rec in Hit@5 and Hit@10, respectively, and 131.50% and 97% gains were reported in NDCG@5 and NDCG@10 metrics. We observe the same pattern of ELECRec outperforming BERT4Rec in most of the evaluation metrics when MobileRec is employed for training and evaluation except for Hit@1 where BERT4Rec does better than ELECRec. Considering the fact that Hit@1 is a stricter criterion, we trust that the training objective of predicting the masked items in the sequence by joint conditioning on bidirectional context, employed by BERT4Rec might be a better training objective for stricter evaluation metrics like Hit@1. Especially, in the context of fleeting user interests like in MobileRec, masked item prediction objective with joint conditioning on bidirectional context seems to be effective for encoding the user's dynamic behavior.

Graph contextualized self-attention model GCSAN employs both graph neural networks and self-attention networks for the session-based recommendation. In the original paper, several comparative evaluations are reported between GCSAN and competing methods. We are primarily interested in the quantification of the comparative representation learning ability of GRU4Rec and Pop versus GCSAN since we also include Pop and GRU4Rec in our baselines along with GCSAN. In the original paper, using Amazon-Books dataset, GCSAN reports improvement over Pop and GRU4Rec in NDCG@5 and NDCG@10. Similarly, improved results are reported as compared to Pop and GRU4Rec in Hit@5 and Hit@10 on different benchmark datasets. We observe similar pattern of GCSAN outperforming Pop and GRU4Rec in NDCG@5, NDCG@10, Hit@5, and Hit@10 consistently on MobileRec. We notice 9.30% and 6.62% improvement over Pop in NDCG@5 and NDCG@10, respectively. Similarly, improvement is observed considering Hit@5, Hit@10, NDCG@5, and NDCG@10 for GCSAN over GRU4Rec.

LightSANs is a transformer variant for the next-item recommendation task which employs a low-rank decomposed self-attention for projecting the user's historical interests into a few latent interests. LightSANs reported Hit@10 and NDCG@10 results on several competing baselines. We will focus on Pop, GRU4Rec, BERT4Rec, and SASRec for comparing and analyzing the LightSANs performance on Amazon-Books dataset and MobileRec. Before further going into the comparative analysis of the performance of Light-SANs on Amazon-Beauty versus MobileRec, let us first look at the Amazon-Books datasets. Amazon-Books dataset has 19K users, 60K items, and 1.7 million interactions. On Amazon-Books dataset, LightSANs reports a 121.77% and 172.43% improvement in Hit@10 and NDCG@10 over Pop. Similarly, 3.91% and 2.65% of improvement is reported in Hit@10 and NDCG@10, respectively, over SASRec. 8.41% and 5.72% improved Hit@10 and NDCG@10 are reported compared to those of GRU4Rec. Finally, an improvement of 8.76% and 4.03% is reported over BERT4Rec. When dataset is MobileRec, we observe 13.90% and 10.38% improvement over Pop in Hit@10 and NDCG@10. LightSANs manages to get 12.41% and 14.86% improvement over GRU4Rec for the same metrics.

Similar to ELECRec, LightSANs also shows 22.85% and 18.05% better results than BERT4Rec for Hit@10 and NDCG@10 metrics.

We notice that LightSANs maintains the pattern of outperforming competing baselines on MobileRec as it does on the Amazon-Books dataset, with SASRec being an exception that outperforms LightSANs in Hit@10 and NDCG@10 metrics. The consistent performance exhibited by LightSANs against BERT4Rec, Pop, and GRU4Rec can be attributed to better design choices in LightSANs. For example, LightSANs proposes a decoupled position encoding in place of implicit position encoding. We believe that this implicit position encoding helps LightSANs to model the historical user interests and user-item interaction more precisely, which steers the model towards better Hit@10 and NDCG@10 on MobileRec. Nevertheless, LightSANs is outperformed by SASRec. The superior performance of SASRec might be due to the learnable position embeddings introduced by SASRec.

SINE investigates the idea of encoding a user's interests with multiple embedding vectors, building upon their empirical findings that a user's behavior sequence exhibits multiple distinct interests. SINE employs several datasets for benchmarking their results. We consider the performance reported by SINE on the Amazon Product Review dataset using Hit@10 and NDCG@10 metrics. We restrict our comparative analysis to SASRec and GRU4Rec. SINE reports improvement over SASRec in Hit@50, Hit@100, NDCG@50, and NDCG@100, respectively. Better performance is also reported against GRU4Rec in Hit@50, Hit@100, NDCG@50, and NDCG@100. Given MobileRec for training and evaluation, SINE manages to outclass GRU4Rec by 7.27% and 4.90% in Hit@20 and NDCG@20, respectively; but struggles to perform better than SASRec on Hit@20 and NDCG@20. First, it might be because Hit@20 and NDCG@20 are stricter criteria as compared to Hit@50, Hit@100, NDCG@50, and NDCG@100 opted by SINE. Secondly, keeping in view that SINE strives to capture the user's distinct interests from the interaction history of the user, we think that converging to distinct user interests in MobileRec presents a challenge because of fluctuating user interests, as depicted in Figure 2. This inability of learning and embedding the distinct user interests from the interaction sequence may lead SINE to show worse performance than SASRec.

7 CONCLUSION

In this paper, we have introduced MobileRec, a large-scale dataset of sequential user-app interactions. The unique feature of MobileRec is that it captures multiple interactions per user, providing a more comprehensive view of user behavior. With a total of 19.3 million user-app interactions, spanning across more than 10 thousand distinct apps from 48 categories, and involving 0.7 million unique users, each with at least five distinct app interactions, MobileRec offers unprecedented granularity in understanding user engagement across different app categories. Moreover, every user-app interaction in MobileRec contains rich contextual information, such as the user's rating, textual review, and review date. Last but not least, each app in MobileRec carries extensive metadata, such as the app's title, category, long-form textual description, overall average rating, developer information, and content rating, among others. We also show the usefulness of the MobileRec dataset as an experimental testbed for research in app recommendation by conducting a comparative evaluation of various state-of-the-art recommendation techniques. This evaluation also establishes baseline results that

will benefit the research community. We hope that our dataset will inspire further research, enable novel insights, and pave the way for future mobile app recommendation systems.

REFERENCES

[1] Movielens. https://grouplens.org/datasets/movielens/, 2022. Accessed: 2022-11-06.

[2] Selenium webdrive. https://www.selenium.dev/documentation/webdriver/, 2022. Accessed: 2023-18-02.

[3] Top 20 play store app reviews. https://www.kaggle.com/datasets/odins0n/top-20-play-store-app-reviews-daily-update, 2022. Accessed: 2022-12-09.

[4] Yelp open dataset. https://www.yelp.com/dataset, 2022. Accessed: 2023-18-02.

[5] Apple. Apple app store. https://apps.apple.com/, 2022. Accessed: 2022-11-06.

[6] L. Ceci. Number of apps available in leading app store. https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/. Accessed: 2022-11-06.

[7] Yongjun Chen, Jia Li, and Caiming Xiong. Elecrec: Training sequential recommenders as discriminators. arXiv preprint arXiv:2204.02011, 2022.

[8] J. Degenhard. Number of apps available in leading app store. https://www.statista.com/forecasts/1143723/smartphone-users-in-the-world. Accessed: 2022-02-02.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[10] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 193–202, 2014.

[11] Jeffrey L Elman. Finding structure in time. Cognitive science, 14(2):179–211, 1990.

[12] Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pages 1733–1737, 2021.

[13] Umar Farooq, AB Siddique, Fuad Jamour, Zhijia Zhao, and Vagelis Hristidis. App-aware response synthesis for user reviews. In 2020 IEEE International Conference on Big Data (Big Data), pages 699–708. IEEE, 2020.

[14] Moghis Fereidouni, Adib Mosharrof, Umar Farooq, and AB Siddique. Proactive prioritization of app issues via contrastive learning. In 2022 IEEE International Conference on Big Data (Big Data), pages 535–544. IEEE, 2022.

[15] Cuiyun Gao, Jichuan Zeng, Xin Xia, David Lo, Michael R Lyu, and Irwin King. Automating app review response generation. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), pages 163–175. IEEE, 2019.

[16] Google. Google play store. https://play.google.com/store/apps, 2022. Accessed: 2022-11-06.

[17] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis), 5(4):1–19, 2015.

[18] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In proceedings of the 25th international conference on world wide web, pages 507–517, 2016.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.

[20] Claudia Iacob and Rachel Harrison. Retrieving and analyzing mobile apps feature requests from online reviews. In 2013 10th working conference on mining software repositories (MSR), pages 41–44. IEEE, 2013.

[21] Tomoharu Iwata, Shinji Watanabe, and Hiroshi Sawada. Fashion coordinates recommender system using photographs from fashion magazines. In Twenty-Second International Joint Conference on Artificial Intelligence, 2011.

[22] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems (TOIS), 20(4):422–446, 2002.

[23] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In 2018 IEEE international conference on data mining (ICDM), pages 197–206. IEEE, 2018.

[24] Hammad Khalid. On identifying user complaints of ios apps. In 2013 35th international conference on software engineering (ICSE), pages 1474–1476. IEEE, 2013.

[25] Hammad Khalid, Emad Shihab, Meiyappan Nagappan, and Ahmed E Hassan. What do mobile app users complain about? IEEE software, 32(3):70–77, 2014.

[26] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 426–434, 2008.

[27] Dong Li, Ruoming Jin, Jing Gao, and Zhi Liu. On sampling top-k recommendation evaluation. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2114–2124, 2020.

[28] Chen Ma, Peng Kang, and Xue Liu. Hierarchical gating networks for sequential recommendation. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pages 825–833, 2019.

[29] Walid Maalej and Hadeer Nabil. Bug report, feature request, or simply praise? on automatically classifying app reviews. In 2015 IEEE 23rd international requirements engineering conference (RE), pages 116–125. IEEE, 2015.

[30] Julian McAuley. Amazon product data. http://jmcauley.ucsd.edu/data/amazon/, 2022. Accessed: 2022-11-06.

[31] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, pages 43–52, 2015.

[32] Stuart McIlroy, Nasir Ali, Hammad Khalid, and Ahmed E. Hassan. Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews. Empirical Software Engineering, 21:1067–1106, 2016.

[33] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pages 188–197, 2019.

[34] Jianmo Ni, Larry Muhlstein, and Julian McAuley. Modeling heart rate and activity data for personalized fitness recommendation. In WWW, 2019.

[35] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[36] Jérémie Rappaz, Julian McAuley, and Karl Aberer. Recommendation on live-streaming platforms: Dynamic availability and repeat consumption. In Proceedings of the 15th ACM Conference on Recommender Systems, pages 390–399, 2021.

[37] Jérémie Rappaz, Julian McAuley, and Karl Aberer. Recommendation on live-streaming platforms: Dynamic availability and repeat consumption. In RecSys, 2021.

[38] Kamonphop Srisopha, Daniel Link, and Barry Boehm. How should developers respond to app reviews? features predicting the success of developer responses. EASE 2021, page 119–128, New York, NY, USA, 2021. Association for Computing Machinery.

[39] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In Proceedings of the 28th ACM international conference on information and knowledge management, pages 1441–1450, 2019.

[40] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. Sparse-interest network for sequential recommendation. In Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pages 598–606, 2021.

[41] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In Proceedings of the 1st workshop on deep learning for recommender systems, pages 17–22, 2016.

[42] Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian McAuley. Attentive sequential models of latent intent for next item recommendation. In WWW, 2020.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[44] Mengting Wan and Julian McAuley. Item recommendation on monotonic behavior chains. In Proceedings of the 12th ACM conference on recommender systems, pages 86–94, 2018.

[45] Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. Fine-grained spoiler detection from large-scale review corpora. arXiv preprint arXiv:1905.13416, 2019.

[46] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. Mind: A large-scale dataset for news recommendation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3597–3606, 2020.

[47] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In IJCAI, volume 19, pages 3940–3946, 2019.

[48] An Yan, Chaosheng Dong, Yan Gao, Jinmiao Fu, Tong Zhao, Yi Sun, and Julian McAuley. Personalized complementary product recommendation. In WWW, 2022.

[49] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In CIKM, pages 4653–4664. ACM, 2021.

[50] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In Proceedings of the 14th international conference on World Wide Web, pages 22–32, 2005.