A Semi-Personalized System for User Cold Start Recommendation on Music Streaming Apps

Léa Briand
Deezer Research
research@deezer.com

Guillaume Salha-Galvan
Deezer Research
LIX, École Polytechnique

Walid Bendada
Deezer Research
LAMSADE, Univ. Paris Dauphine

Mathieu Morlon
Deezer Research

Viet-Anh Tran
Deezer Research

ABSTRACT

Music streaming services heavily rely on recommender systems to improve their users' experience, by helping them navigate through a large musical catalog and discover new songs, albums or artists. However, recommending relevant and personalized content to new users, with few to no interactions with the catalog, is challenging. This is commonly referred to as the *user cold start* problem. In this applied paper, we present the system recently deployed on the music streaming service Deezer to address this problem. The solution leverages a semi-personalized recommendation strategy, based on a deep neural network architecture and on a clustering of users from heterogeneous sources of information. We extensively show the practical impact of this system and its effectiveness at predicting the future musical preferences of cold start users on Deezer, through both offline and online large-scale experiments. Besides, we publicly release our code as well as anonymized usage data from our experiments. We hope that this release of industrial resources will benefit future research on user cold start recommendation.

CCS CONCEPTS

• **Information systems** → *Recommender systems*; *Personalization*.

KEYWORDS

Recommender Systems; User Cold Start; Music Streaming Service; Semi-Personalization; Heterogeneous Data; A/B Testing.

ACM Reference Format:
Léa Briand, Guillaume Salha-Galvan, Walid Bendada, Mathieu Morlon, and Viet-Anh Tran. 2021. A Semi-Personalized System for User Cold Start Recommendation on Music Streaming Apps. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467110

1 INTRODUCTION

Recommender systems are essential tools for online services providing access to large catalogs, such as e-commerce websites [39, 46], and video [9, 13] or music [2, 10, 19, 36] streaming services. They help users navigate through massive amounts of content, discover new products, movies or songs they might like, and are known to improve the user experience and engagement [6, 31, 36, 51].

A prevalent approach to effectively recommend personalized content on these online services is *collaborative filtering* which, broadly, consists in predicting the preferences of a user within a set of items by leveraging the known preferences of some similar users [9, 20, 36, 41]. In particular, several recent works emphasized the empirical effectiveness of *latent models* for collaborative filtering at addressing industrial-level challenges [9, 13, 19, 39]. In a nutshell, these models aim at directly learning vector space representations, a.k.a. *embeddings*, of users and items where proximity should reflect user preferences, typically via the factorization of a user-item interaction matrix [18, 20, 21] or with neural network architectures processing usage data [9, 30, 45].

However, the performances of these models tend to significantly degrade for new users who only had few interactions with the catalog [7, 26, 36]. They might even become unsuitable for users with no interaction at all, who are absent from user-item matrices in standard algorithms [5, 15, 24]. This is commonly referred to as the *user cold start* problem [5, 24, 26, 36, 44]. Yet, recommending relevant content to these new users is crucial for online services. Indeed, a new user facing low-quality recommendations might have a bad first impression and decide to stop using the service.

In this applied paper, we present a system recently deployed on the global music streaming app Deezer[1], connecting 14 million active users from 180 countries to 73 million music tracks, to practically address this problem. The solution starts from an existing large-scale latent model for collaborative filtering, periodically trained on Deezer's *warm* users (see Section 3). It automatically integrates *cold* users into the existing embedding space, by collecting heterogeneous sources of demographic and interaction information on these users at registration day, processed by a deep neural network, and by leveraging a segmentation of warm users to strengthen the final representations and provide semi-personalized recommendations to cold users *by the end of their registration day*.

The proposed system is suitable for an online production use on a large-scale app such as Deezer. Throughout this paper, we extensively show its practical impact and its empirical effectiveness at predicting the future musical preferences of cold users, through both offline experiments on data extracted from Deezer and an online A/B test on the actual Deezer app. We also emphasize how

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467110

[1] https://www.deezer.com

this system enables us to provide more interpretable music recommendations. Last, along with this paper we publicly release our source code as well as anonymized usage data of Deezer users from our offline experiments. Besides making our results reproducible, we believe that this open-source release of industrial resources will benefit future research on user cold start, by providing to the scientific community a relevant real-world benchmark dataset to evaluate future recommender systems.

This paper is organized as follows. In Section 2, we introduce the user cold start problem more precisely and mention previous research efforts on this topic. In Section 3, we present the semi-personalized recommender system deployed on the Deezer app. We report and discuss our experimental setting, our data, and our results in Section 4, and we conclude in Section 5.

2 PRELIMINARIES

In this section, we provide a precise formulation of the problem we aim at addressing. We also give an overview of the existing related work. Some of the mentioned approaches will constitute relevant baselines to evaluate the effectiveness of our system.

2.1 Problem Formulation

Throughout this paper, we consider a catalog of m music tracks available on the music streaming app Deezer. We assume that the catalog remains fixed over time, which we later discuss. At time t, Deezer gathers n_t *warm* users who, according to some criteria internally fixed by our data scientists, had a sufficiently large number of interactions with the catalog, e.g. enough listening sessions, to be used in the training of our recommender systems. We consider a *latent model for collaborative filtering* [9, 20, 21, 30]. From the observed warm user-track interactions and following the processes described in Section 3, this model learns a vector space representation of both users and tracks. In this *embedding* space, each user i and track j are represented by d-dimensional vectors (with $d \ll m$ and $d \ll n_t$), say $u_i \in \mathbb{R}^d$ and $v_j \in \mathbb{R}^d$, capturing musical preferences. To recommend relevant new tracks to the user i, we leverage user-item similarity measures $f(u_i, v_j)$ in this space, encoding user-item affinities. These measures are typically based on an inner-product or a cosine similarity [21]. The model is updated at regular time intervals to take into account the evolution of preferences.

Everyday, new users, referred to as *cold*, will register to the service. They will only have few to no interactions with the catalog during their registration day. As explained in the introduction, a straightforward inclusion of these cold users in the aforementioned latent model is unsuitable [5, 15, 24, 26]. Waiting for them to become warm users (according to internal criteria) is also undesirable: indeed, recommending relevant content as soon as possible is crucial, as new users facing low-quality recommendations might make up their mind on this first impression and quickly stop using the service. As a consequence, we aim at addressing the following problem: *given an existing latent model for collaborative filtering learning an embedding space from a set of warm users, how can we effectively include new cold users into this same space, by the end of their registration day on Deezer?* In the remainder of this paper, we will evaluate the estimated embedding vectors of cold users by assessing their ability at predicting the future musical preferences of these users on Deezer after their registration day, through the evaluation tasks and metrics presented in Section 4.

2.2 Related Work

The user cold start problem has initiated significant research efforts over the past decade [4, 5, 7, 15, 22–24, 26, 27, 31, 36, 38, 39, 44, 50, 51]. In the following, we provide an overview of the most relevant work w.r.t. our approach. We refer the interested reader to some recent surveys [15, 31, 51] for a more exhaustive review of the existing literature, and to [31, 39, 42, 46] for a presentation of the related *item cold start* problem, which is out of our scope.

A prevalent strategy to address the user cold start problem in the total absence of usage data consists in relying on metadata related to new users, and notably on demographic information (such as the age or country of the user) collected during registration [12, 22, 23, 26, 31, 48]. In particular, various approaches aim at clustering warm users, and subsequently assign cold users to existing clusters by leveraging these metadata [7, 11, 26, 31, 41, 48]. Building upon these works, the model we present in Section 3 will also leverage demographic information and incorporate a clustering component.

Besides, one can enrich such systems by explicitly asking new users to rate items from the catalog through various interview processes, leading to hybrid models based on preferences and side information [15, 31, 38]. On the industry side, Netflix [13] and Spotify [19] are famous examples of services implementing such *onboarding* session for new users; as explained in Section 3, Deezer also adopted this strategy. As the inclusion of an onboarding is not always possible in production, authors of [11] propose to use bandit algorithms to assign cold users to warm user segments, while [27, 37] resort to social media data to connect similar users.

Sometimes, cold users do have a few interactions with the catalog on their registration day. In this case, exploiting such usage signal, in addition to the aforementioned side information, can significantly improve recommendations [5, 17, 36, 41]. In particular, several recent works emphasized the effectiveness of *deep learning* models at dealing with such heterogeneous settings [4, 5, 9, 24, 31, 44, 51]. Notably, authors of [9] explain how a scalable deep neural network, processing various user-item interactions (including watched videos, search queries...) and demographic information to learn embedding vectors, improved the YouTube recommender system. To represent interactions, they average various latent representations of watched/searched items from a same user session, allowing features to have the same dimension for each user. In the method we present in Section 3, we will draw inspiration from their approach to preprocess the user features serving as input to our embedding model. However, a direct comparison to [9] is impossible, as no complete description nor implementation of the YouTube recommender system was made publicly available.

Among the reproducible deep learning approaches, DropoutNet [44] emerged as one of the most powerful latent collaborative filtering models, addressing cold start while preserving performances for warm users. This neural network takes into account usage and content data, and explicitly simulates the cold start situation during training by applying *dropout* [40], alternatively to user and item embedding layers. DropoutNet relies on the assumption that data is *missing at random*, with the risk of introducing biased predictions

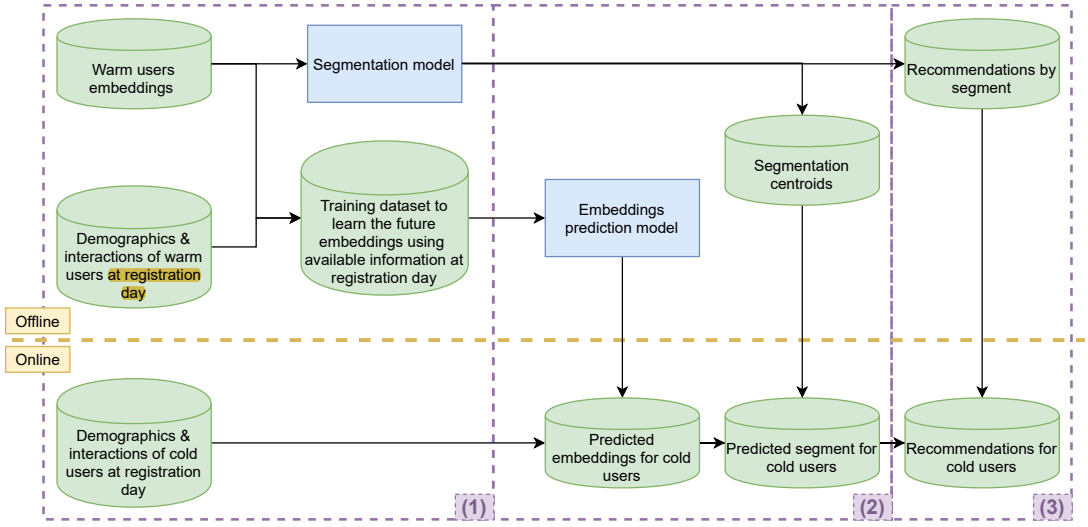


Figure 1: The semi-personalized user cold start recommendation framework available for online requests in our production environment, and described throughout Section 3. (1) Demographics and user-item interactions are concatenated, as described in 3.2.2, to (2) predict a user embedding vector as in 3.2, from warm user embeddings described in 3.1.2 and 3.1.3. From the estimated user preferences, the new user is assigned to a segment of warm users. (3) Combining the online predicted segment with the pre-computed top items by segment, cold users benefit from semi-personalized recommendations.

[28]. Also, it equally considers different types of positive feedback; in Section 3, we will furthermore consider negative feedback. Besides, both warm and cold users' embeddings are learned during training, whereas our system will directly incorporate cold users in an existing and fixed embedding space of warm users.

Recently, *meta-learning* based methods [43] have also been proposed to solve the user cold start problem, with promising performances [4, 24, 51]. Notably, optimization-based algorithms consider each user as a learning task. From a set of global parameters ensuring an initialization of the recommender system, other local parameters are progressively updated while the user interacts with items to capture his/her preferences. In particular, authors of [24] introduce MeLU (for *Meta-Learned User preference estimator*), a neural network architecture following such a meta-learning paradigm and learning preferences from the concatenation of user and item information. When new users interact with some items, then *local* parameters of the neural network are updated to refine predictions for these users. In the absence of usage data, users will still be associated to an embedding vector and receive a recommended list of items, thanks to *global* updates of all layers of MeLU.

Last, we point out that some research efforts also transposed recent advances in *graph representation learning* [16] to recommender systems [3, 47, 49, 50]. Especially, [50] proposed a *Stacked and Reconstructed Graph Convolutional Network (STAR-GCN)* architecture, extending ideas from [3] to tackle the user cold start problem from bipartite user-item graphs. Although such a work deserves a mention, we will however not include it in our baselines, as the model returned memory errors in our experiments (with hundreds of thousands to millions of users) and thus did not scale to Deezer data. Nonetheless, we believe that future works on scalable graph models for user cold start, e.g. by leveraging ideas from [8, 34, 35], could definitely lead to promising new methods, especially considering the recent successes of scalable graph-based recommender systems on large-scale services such as Alibaba [46] or Pinterest [49].

3 A SEMI-PERSONALIZED SYSTEM TO ADDRESS USER COLD START ON DEEZER

In this section, we present the system deployed in 2020 on the global music streaming app Deezer to address the user cold start problem, as formulated in Section 2.1. The architecture of our framework is summarized in Figure 1, and discussed thereafter. The proposed solution is suitable for production use on an online service.

3.1 Two Latent Models to Represent Musical Preferences of Warm Users

We recall that our objective is to effectively incorporate, *by the end of their registration day*, a set of cold users in an existing embedding space trained on warm users. In the following, we introduce two different strategies to learn such a space on Deezer data. Although some technical details on computations are omitted for confidentiality reasons, some embedding vectors from both models will be released with this paper (see Section 4). Moreover, experimental results on both spaces will be reported in the next section.

3.1.1 UT-ALS Embeddings. Latent models for collaborative filtering can approximate a preference matrix between users and items from the product of two low-rank matrices, respectively stacking up latent vector representations, a.k.a. embedding vectors, of users and items [20, 41]. At Deezer, we consider a *user-track (UT) interaction matrix* summarizing interactions between millions of active users and music tracks from the catalog. The *affinity score* between user i and music track j, i.e. the entry (i, j) of the matrix, is computed from

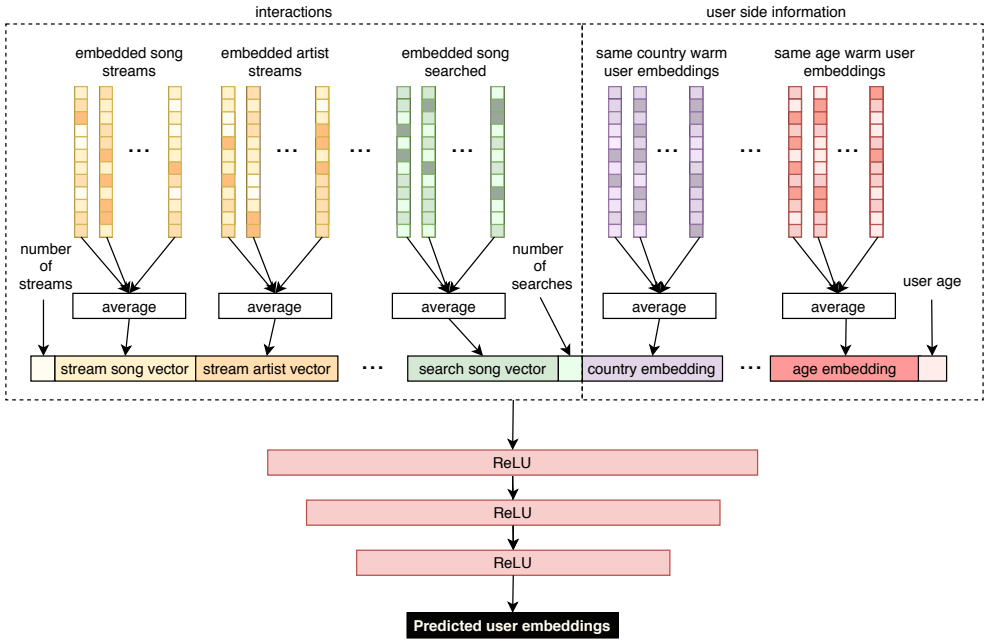


Figure 2: Prediction of user embeddings from heterogeneous data. Top left: embedding vectors of activated items in usage data, including onboarding, are aggregated as described in 3.2.2. Top right: we enrich representations with demographic information. Bottom: after pre-processing the dense input features vector, a deep neural network model, trained as in 3.2.3, predicts user embedding vectors in either the UT-ALS embedding space (from 3.1.1) or the TT-SVD embedding space (from 3.1.2).

various signals, including the number of streams and the potential addition of the music track (or the corresponding album or artist) to a playlist of favorites. The final entry is refined in accordance with internal heuristic rules. Then, we rely on a *weighted matrix factorization*, specifically by using the *alternating least squares* (ALS) method [21], to map both users and music tracks to a joint latent space of dimension $d = 256$. These vector representations will be referred to as *UT-ALS embeddings* in the remainder of this paper.

3.1.2 TT-SVD Embeddings. Models inspired from word2vec [29] rely on the *distributional hypothesis* [33] to map items co-occurring in similar contexts to geometrically close embedding vectors. Authors of [25] show that word2vec with negative sampling implicitly factorizes a shifted *pointwise mutual information* (PMI) matrix using *singular value decomposition* (SVD) [21]. In this paper, we also consider a PMI matrix, based on the co-occurrences of music tracks in diverse music collections on Deezer, such as music playlists. Then, we factorize this *track-track (TT) matrix* using a distributed implementation of SVD[2], leading to embedding vectors of dimension $d = 128$ for each music track[3]. Finally, we derive embedding vectors for warm users, by averaging music track vectors over their listening history on Deezer. These vector representations, different from *UT-ALS embeddings*, will be referred to as *TT-SVD embeddings* in the remainder of this paper.

[2] https://github.com/criteo/Spark-RSVD

[3] Embedding dimensions of UT-ALS and TT-SVD have been optimized independently for recommendation, and are therefore different (256 vs 128). The choice of SVD vs ALS factorization is also driven by internal optimizations on Deezer data. In experiments we will simply exploit these vectors, independently, for user cold start.

3.1.3 Warm User Segmentation. On top of UT-ALS or TT-SVD user embeddings, our system also computes a segmentation of warm users, by running a k-means algorithm, with $k = 1\,000$ clusters/segments, in the embedding space. Each user segment is represented by its *centroid* i.e. by the average of its user embedding vectors. In production, a list of the most popular music items to recommend among each warm user segment is also pre-computed.

3.2 Predicting the Preferences of Cold Users

In the following we present our model, illustrated in Figure 2, to integrate cold users into these embedding spaces, and subsequently predict their future musical preferences on Deezer.

3.2.1 Overall Strategy. Firstly, we gather data from various sources, presented in 3.2.2 and referred to as *input features*. They can be collected for warm users and (at least partially for) cold users. Then, we train a neural network (3.2.3) to map input features of warm users to their (either UT-ALS or TT-SVD) embedding vectors. Last, through a forward pass on this trained neural network, we predict embedding vectors for cold users from their input features. Cold users are therefore integrated into the existing latent space alongside warm users and each track of the catalog. This will permit computing cold user-track similarities, and even similarities between cold and warm users, which we leverage in 3.2.4 for clustering.

3.2.2 Input features. During registration, all users specify their age and country of origin, which we include in input features. This information is enriched with *country embedding* and *age embedding*

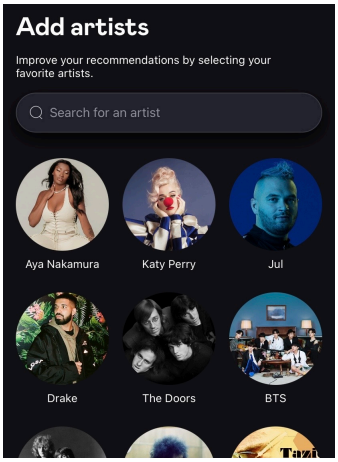


Figure 3: Overview of the *onboarding* process on Deezer.

vectors which, as illustrated in Figure 2, are the average of embedding vectors of warm users from respectively the same country and age class. As hybrid models mentioned in Section 2.2, we complement this side information with data retrieved from user-item interactions, limiting to interactions at registration day (if any). These interactions include *(positive or negative) explicit and implicit signals*, including streaming activity, searches, skips and likes. As granularity of items is crucial in music [36], we also compute such signals at the album, artist and playlist levels, deriving embedding vectors for such music entities by averaging the relevant track embeddings (e.g. by averaging the tracks of an album, or part of the discography of an artist). Then, for each type of interaction and music entity, embedding vectors are averaged, as illustrated in Figure 2. We obtain *fixed-size representations* for both demographics and interactions, i.e. independent of the number of modalities or interactions, which is crucial for scalability in a production environment. For some new users, some (or possibly all) types of interactions at registration day may be missing; the corresponding representations are replaced arbitrarily by null vectors. To avoid such a situation, the Deezer app includes an *onboarding* process for newly registered users, proposing them to add artists from various music genres to their list of favorites, as illustrated in Figure 3.

3.2.3 Model training. Fixed-size representations of demographics and interactions are concatenated to form a unique dense input vector of dimension 5139 (when considering UT-ALS embeddings) or 2579 (TT-SVD embeddings). It constitutes the input layer of a feedforward neural network, with three hidden layers of dimensions 400, 300 and 200 respectively, and an output layer of dimension $d = 256$ (when considering UT-ALS embeddings) or 128 (TT-SVD embedding). We use *rectified linear unit* (ReLU) activations at each layer except the output, followed by *batch normalization* [14]. We train the model on warm users, by iteratively minimizing the *mean squared error* between the predicted user embeddings and their actual value in UT-ALS or TT-SVD spaces, by *stochastic gradient descent* [14] with a learning rate of 0.0001, batch sizes of 512, and 100 (respectively 130) epochs for TT-SVD (resp. UT-ALS).

3.2.4 Semi-Personalization. Our model integrates cold users into the existing embedding space alongside warm users and music tracks. Therefore, one could provide *fully personalized* music recommendation to each of those cold users, by retrieving the most *similar* tracks for each user via an exact or an approximate *nearest neighbors* search and some similarity measure. However, as we will empirically show in our experiments, such a strategy can still lead to noisy results for users with very few to no usage data. As a consequence, our system instead adopts a *semi-personalized* recommendation strategy. On top of our neural network predictions, we include cold users into the pre-computed *warm user segmentation*, described in Section 3.1.3. Specifically, each cold user is assigned to the warm cluster whose centroid is the closest w.r.t. the predicted embedding vector of this cold user. We subsequently recommend the pre-computed most popular tracks among warm users from the cluster. Our framework is summarized in Figure 1.

3.3 Model Deployment

This system is suitable for online production use on large-scale apps. At Deezer, the real-time inference service to predict user embeddings is a Golang web server, deployed in a Kubernetes cluster. The web service wraps the onnxruntime library[4], a fast engine for running ONNX machine learning models. It permits fast predictions of cold users' embeddings via forward passes on already trained neural networks. Models are trained offline using PyTorch, on an NVIDIA GTX 1080 GPU and an Intel Xeon Gold 6134 CPU, and then exported to ONNX format and stored on Hadoop. Embeddings of music tracks (from which we also derive embeddings of artists, albums or playlists) as well as warm segment centroids are exported weekly in tables in a Cassandra cluster, exposed via a JSON REST service. Embeddings and serialized models are weekly updated to take into account changes in the catalog and preferences, and weekly exported as well.

4 EXPERIMENTAL ANALYSIS

In the following, we evaluate the performance and impact of our system on Deezer data, through both offline and online experiments.

4.1 Offline Evaluation

4.1.1 Predicting Future Preferences of Cold Users: Evaluation Task and Metrics. Our system permits recommending musical content to cold users. Through experiments on an offline dataset of Deezer active users, described thereafter, we evaluate to which extent the proposed recommendations at registration day would have matched the actual musical preferences of a set of users on their first month on the service. Specifically, we compute the 50 most relevant music tracks for each user of the dataset, from our model and registration day's input features (described in Section 3.2.2). We compare them to the tracks listened by each user during their next 30 days on Deezer, using three standard recommendation metrics: the *precision*, the *recall*, as well as the *normalized discounted cumulative gain* (NDCG) as a measure of ranking quality [36].

4.1.2 Dataset. For offline experiments, we extracted a dataset of 100 000 fully anonymized Deezer users. Among them, 70 000 are

[4] https://www.onnxruntime.ai/

warm users. They are associated to demographic information (country and self-reported age), as well as their respective UT-ALS and TT-SVD embedding vectors. These vectors correspond to those actually computed by our latent collaborative filtering models on the Deezer production system on November 1st, 2020, from *millions of active warm users*. Our dataset also includes the UT-ALS and TT-SVD embedding representations of the 50 000 most popular anonymized music tracks on Deezer.

The remaining 30 000 users are *cold* users, who registered on Deezer on the first week of November 2020, and subsequently listened to at least 50 music tracks *on their first month on the service (excluding registration day)*. They are split into a validation set and a test set of respectively 20 000 and 10 000 users. For each cold user, we collected demographic information as well as the list of artists potentially selected during the onboarding session, and the lists of available streams, skips, bans, searches, additions to favorites relative to music tracks, artists, albums and playlists *at registration day only*. 77% of cold users from this dataset streamed at least once at registration day, whereas 95% of cold users fulfilled one of the aforementioned interactions. Thus, for the remaining 5%, only demographic information is available at registration day. Last, the dataset includes the tracks listened by cold users during their next 30 days on Deezer, among the 50 000 tracks.

Along with this paper, we publicly release[5] this dataset, as well as the code corresponding to our offline experiments. Besides making our results reproducible, it will publicly provide a relevant real-world benchmark dataset to evaluate and compare future cold start models on actual (albeit anonymized) usage data. We therefore hope that this open-source release of industrial resources will favor and benefit future research and applications on user cold start problems.

4.1.3 Models. We report results from two versions of our system (one trained on the UT-ALS embeddings of the 70 000 warm users, and one trained on their TT-SVD embeddings) on the task presented in 4.1.1. For each embedding space, we simultaneously evaluate:

- the *semi-personalized* recommendations, which are actually used in production at Deezer. In this case, the 50 recommended tracks of each user will correspond to the 50 most popular tracks of his/her *user segment*, as detailed in 3.2.4 ;
- the *fully-personalized* ones, which directly leverage the predicted embedding vectors of each cold user from the neural network. In this case, we recommend, for each cold user, the 50 *nearest neighbors* music tracks w.r.t. his/her vector in the embedding space, according to a cosine similarity.

Moreover, although the main objective of this applied paper is to show the very practical impact of a deployed system and not to chase the state of the art, we also report the performances of several baseline methods as a point of comparison. Foremost, we consider a *popularity-based* baseline. This chart-based method recommends the most popular songs to cold users. Besides, to motivate the need for a careful modeling of user cold start, we consider the *Registration Day Streams* method, which includes cold users in the embedding in a more straightforward way. It estimates a cold user's embedding vector by averaging embedding vectors of music tracks listened at registration day (and relying on popularity in the absence of any stream), then recommends 50 tracks to each cold user via a nearest neighbors search. Moreover, a third ablation baseline, denoted *Input Features Clustering* in the following, will consist in getting rid of our neural network model and directly rely on the *input features* of Section 3.2.2. This method will also cluster all users into segments via a k-means, but from their stacked input features, i.e. the large vector illustrated in Figure 2. Recommended tracks will correspond to the 50 most popular among warm users of each cluster; as users are no longer in the same space as tracks, we do not evaluate any fully-personalized version of this baseline.

Last, we evaluate DropoutNet [44] and MeLU [24], two powerful deep learning models from the recent literature, described in Section 2. We selected these two methods as they are simultaneously *1)* among the most promising approaches, to the best of our knowledge, *2)* scalable to large datasets, and *3)* publicly available online[6]. They process the same input features as our model, with the notable exception that DropoutNet only processes positive user-track interactions (i.e. not skips nor bans, representing approximately 12% of all interactions in our dataset). We carefully tuned each model using the validation set. For each model, we simultaneously evaluate *fully-personalized* recommendations where, as for our system, we recommend to each cold user his/her 50 most similar music tracks, as well as *semi-personalized* recommendations leveraging a warm user segmentation similar to ours. We consider two variants of each model from the last two paragraphs, respectively trained on UT-ALS and TT-SVD embeddings.

4.1.4 Results. Table 1 reports performance scores for all models on the UT-ALS and TT-SVD embeddings, along with standard deviations over ten iterations. As the *Popularity* baseline is independent from embedding vectors, it obtains the same scores for the two embeddings. In both settings, *Popularity* is the worst method, although performances are still fairly good for such a simple strategy.

We first focus on TT-SVD embeddings. The *Registration Day Streams* baseline hardly beats *Popularity* for these embeddings, which emphasizes the limits of a direct use of sparse usage data in cold start settings. On the contrary, our proposed semi-personalized system provides significant improvements (e.g. a +14.86 NDCG points increase w.r.t. *Registration Day Streams*), and even reaches competitive results w.r.t. DropoutNet and MeLU. Overall, semi-personalized methods outperform their fully-personalized variants (e.g. a +13.17 precision points increase for *Deezer Semi-Pers.* vs *Deezer Full-Pers.*). This confirms the empirical relevance of our user segmentation strategy, and that, while the studied methods *could* provide fully personalized recommendations to cold users, this strategy can lead to noisier results on real-world applications such as ours. Last, our system provides better results than a direct use of the *input features* vector (*Input Features Clustering* baseline), which confirms the relevance of our modeling step on top of this vector. We considered replacing our 3-layer neural network by alternative architectures between *Input Features Clustering* and *Deezer Semi-Pers.*, notably by a 1-layer neural network i.e. a simpler linear regression model, but did not reach comparable results.

Regarding the UT-ALS embeddings, most conclusions are consistent w.r.t. TT-SVD. *Deezer Semi-Pers.* reaches quite comparable or

[5] Data and code are available on: https://github.com/deezer/semi_perso_user_cold_start

[6] Public implementations are available on https://github.com/layer6ai-labs/DropoutNet and https://github.com/hoyeoplee/MeLU respectively.

Table 1: Offline prediction of future musical preferences of Deezer cold users.

Methods	TT-SVD Embeddings			UT-ALS Embeddings		
	Precision@50 (in %)	Recall@50 (in %)	NDCG@50 (in %)	Precision@50 (in %)	Recall@50 (in %)	NDCG@50 (in %)
Popularity	8.92 ± 0.21	3.01 ± 0.08	9.72 ± 0.20	8.92 ± 0.21	3.01 ± 0.08	9.72 ± 0.20
Registration Day Streams	9.30 ± 0.22	3.48 ± 0.06	9.73 ± 0.23	16.88 ± 0.46	5.99 ± 0.11	17.72 ± 0.43
Input Features Clustering	8.84 ± 0.22	2.97 ± 0.08	9.75 ± 0.23	8.85 ± 0.22	2.98 ± 0.08	9.75 ± 0.22
DropoutNet Full-Pers.	10.04 ± 0.27	3.75 ± 0.11	10.46 ± 0.29	16.30 ± 0.50	5.77 ± 0.50	17.62 ± 0.54
DropoutNet Semi-Pers.	20.85 ± 0.35	7.55 ± 0.12	22.61 ± 0.36	**19.83 ± 0.32**	**6.93 ± 0.16**	**21.55 ± 0.44**
MeLU Full-Pers.	15.00 ± 0.40	5.12 ± 0.17	16.79 ± 0.45	13.92 ± 0.36	4.71 ± 0.12	15.49 ± 0.39
MeLU Semi-Pers.	19.66 ± 0.36	6.87 ± 0.15	21.63 ± 0.40	**19.35 ± 0.43**	**6.71 ± 0.14**	**21.33 ± 0.45**
Deezer Full-Pers. (ours)	9.58 ± 0.18	3.53 ± 0.03	9.77 ± 0.17	18.50 ± 0.43	6.63 ± 0.10	20.22 ± 0.41
Deezer Semi-Pers. (ours)	**22.75 ± 0.32**	**8.26 ± 0.15**	**24.59 ± 0.30**	**19.00 ± 0.42**	**6.93 ± 0.10**	20.38 ± 0.45

better results w.r.t. alternatives. Among the key differences, we highlight that, while *Deezer, DropoutNet and MeLU Semi-Pers* are overperforming, the fully-personalized *Registration Day Streams*, *DropoutNet Full-Pers* and *Deezer Full-Pers* obtain significantly stronger results than on TT-SVD embeddings (e.g. a 18.50% precision score for *Deezer Full-Pers.* on UT-ALS, vs 9.58% on TT-SVD). UT-ALS embeddings are constructed from a user-item interaction matrix, an approach appearing as better suited for nearest neighbors search.

To conclude on Table 1, we emphasize that, while all scores might seem relatively low, they are actually encouraging considering the intrinsic complexity of the evaluation task (predicting a few listened tracks, among 50 000). Overall, the TT-SVD version of *Deezer Semi-Pers* reaches the best results. We note that the choice of @50 metrics is not restrictive. We reached consistent model rankings for @25 and @100 scores; the number of items to recommend will be a selectable parameter in our public implementation.

To go further, Figure 4 reports the mean precision@50 scores obtained by our semi-personalized system trained on TT-SVD embeddings, depending on available user-item interactions at registration day. We observe that our model returns better results for users who streamed (positive signal) or skipped (negative signal) more tracks at registration day. Liking at least three artists during the onboarding session also improves recommendations, which tends to confirm the relevance of this strategy. On the contrary, our experiments show that users without *any* interaction (5% of them, for which only demographics are available) gets a lower average precision@50 score of 17.74%, 5.01 points below the global average.

Last, Figure 5 reports the popularity distribution of recommended tracks from our system, trained on TT-SVD embeddings and for users from the top 5 countries on Deezer. We aim at assessing whether better performance inevitably means recommending popular tracks, which is referred to as the *popularity bias* [32]. Our semi-personalized system stands out from the popularity baseline, and mainly recommends tracks among the 5 000 most popular in the dataset, out of 50 000. The fully-personalized variant of our system permits recommending even less *mainstream* tracks, but, as we showed, this might come at the price of noisier recommendations.

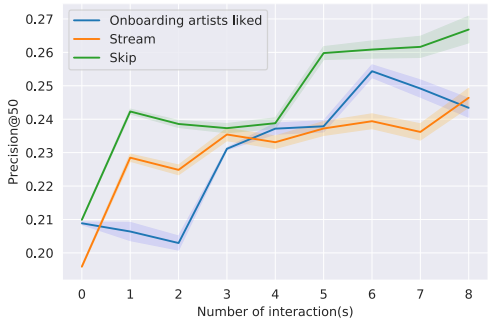


Figure 4: Precision@50 scores of Deezer Semi-Pers. depending on the number of artists liked during the onboarding session, of streams, and of skips *at registration day.*

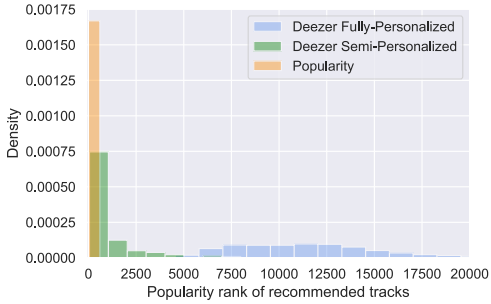


Figure 5: Distribution of music tracks recommended to cold users, by popularity rank on Deezer.

4.2 Online Evaluation

4.2.1 Online Personalization. In addition to these experiments on data extracted from Deezer, online tests were run to check whether our conclusions would hold on the actual Deezer app. On our homepage, we do not directly recommend music tracks, as in our offline experimental setting, but instead *musical collections* such as albums and music playlists. As a consequence, our online tests will rather consist in *recommending playlists to cold users*. The embedding vectors of these playlists are computed from averages of the TT-SVD embeddings of music tracks from each playlist and from internal heuristics. More precisely, Deezer displays 12 recommended

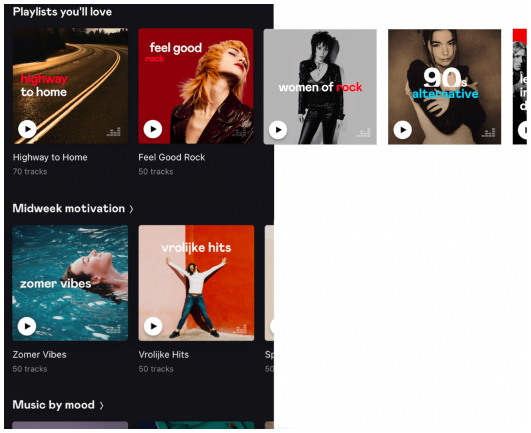


Figure 6: Personalized carousels on the Deezer app.

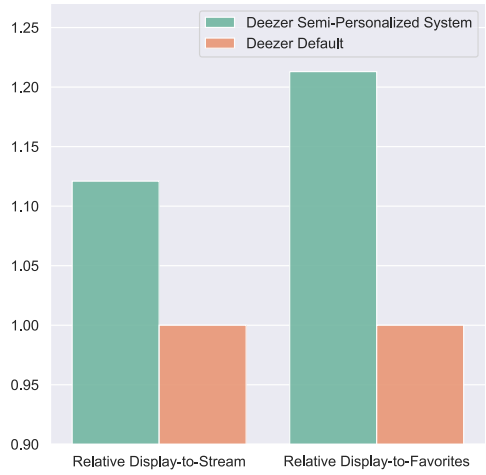


Figure 7: Online A/B test: relative *display-to-stream* and *display-to-favorites* rates w.r.t. internal baseline. Differences are statistically significant at the 1% level (p-value <0.01).

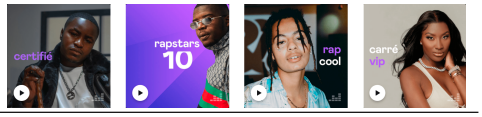
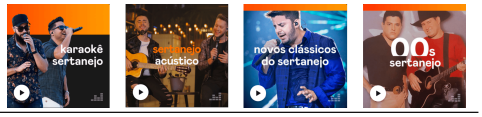
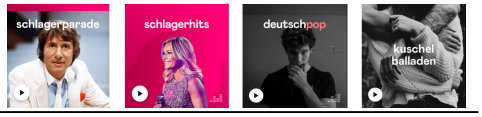
playlists to each user through *carousels* [2], that are ranked and swipeable lists of playlists cards, as illustrated in Figure 6. Carousels are updated on a daily basis on the app[7]. All playlists were created by professional curators from Deezer, with the purpose of complying with a specific music genre, cultural area or mood.

4.2.2 Online A/B Test Results. A large-scale A/B test has been run for a month on Deezer in 2020, on new cold users registering during this period. Due to industrial constraints, testing all model variants in production was impossible. Also, for confidentiality reasons, we do not report the exact number of users involved in each cohort, and results are expressed in *relative* terms w.r.t. the performance of *Deezer Default*, a previous production system estimating cold user embeddings by countries and from internal heuristics. We observe in Figure 7 that our new (TT-SVD-based) semi-personalized system leads to significant improvements of the relative *display-to-stream* and *display-to-favorite* rates, i.e. it permits selecting playlists on which cold users are more likely to click on and then to *stream* the underlying content or *add it to their list of favorite content*. Such results validate the relevance of our proposed system, and emphasize its practical impact on industrial-scale applications in a global music streaming app.

4.2.3 Towards more interpretable recommendations. We underline that our semi-personalized system also enables us to provide *interpretable* recommendations on Deezer. Indeed, carousel personalization relies on user embedding vectors that, for cold users, are linked to centroids of warm user segments. Therefore, recommended playlists can be described via the characteristics of these segments, such as the most common country or age class of warm users from each segment, or the most common music genres among their favorite artists. Table 2 reports a few concrete examples of user segment descriptions, along with music playlists that were actually recommended to cold users from these segments on the Deezer app. Providing interpretable recommendations is often desirable for industrial applications, both for data scientists dealing with opaque model predictions, and for users as a way to improve their satisfaction and trust in the system [1].

[7]We provide a more detailed description to some algorithms behind dynamic carousel personalization for warm Deezer users in [2].

Table 2: Examples of user segments, described by most common country, age class and music genres among favorite tracks, together with online recommended playlists.

Segment Description	Recommended Playlists on Deezer
France 18-24 y.o. Rap, Hip-Hop	
Brazil 25-34 y.o. Sertanejo	
Germany 35-49 y.o. Schlager, Pop	

5 CONCLUSION

In this paper, we presented the semi-personalized system recently deployed in production at Deezer to address the challenging user cold start problem. We demonstrated the tangible impact of this system, through both offline and online experiments on music recommendation tasks. Moreover, although our main focus was on practical impact and not on chasing the state of the art, we also showed that our approach is competitive w.r.t. some powerful user cold start models from the recent scientific literature. Last, along with this paper we publicly release our code, as well as the dataset used in our offline experiments, providing information on 100 000

anonymized users and their interactions with the Deezer catalog. We hope that this open-source release of industrial resources will enable future research on user cold start recommendation. For instance, in this paper we assumed that, while the number of users increases over time, the musical catalog remains fixed, which is a limit, currently addressed at Deezer via internal heuristics. Future studies on a more effective incorporation of new releases in our semi-personalized system would decidedly improve its empirical effectiveness. Besides, future studies could also consider different embedding sizes for users and items [17], or the inclusion of temporal and contextual information in our neural network. Last, as we rely on user segments, we also believe that future work on a more effective user clustering, e.g. by leveraging graph-based methods, could strengthen our system and permit providing even more refined recommendations to new users.

REFERENCES

[1] Darius Afchar and Romain Hennequin. 2020. Making Neural Networks Interpretable with Attribution: Application to Implicit Signals Prediction. In *Fourteenth ACM Conference on Recommender Systems*. 220–229.

[2] Walid Bendada, Guillaume Salha, and Théo Bontempelli. 2020. Carousel Personalization in Music Streaming Apps with Contextual Bandits. In *Fourteenth ACM Conference on Recommender Systems*. 420–425.

[3] Rianne Van Den Berg, Thomas Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *KDD Deep Learning Day* (2017).

[4] Homanga Bharadhwaj. 2019. Meta-Learning for User Cold-Start Recommendation. In *2019 International Joint Conference on Neural Networks*. IEEE, 1–8.

[5] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Jesús Bernal. 2012. A Collaborative Filtering Approach to Mitigate the New User Cold Start Problem. *Knowledge-Based Systems* 26 (2012), 225–238.

[6] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender Systems Survey. *Knowledge-Based Sys.* 46 (2013), 109–132.

[7] Ke Yin Cao, Yu Liu, and Hua Xin Zhang. 2020. Improving the Cold Start Problem in Music Recommender Systems. In *J. of Physics: Conf. Series*, Vol. 1651. 012–067.

[8] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *ICLR*.

[9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for Youtube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 191–198.

[10] Elena V. Epure, Guillaume Salha, and Romain Hennequin. 2020. Multilingual Music Genre Embeddings for Effective Cross-Lingual Music Item Annotation. In *31st International Society for Music Information Retrieval Conference*.

[11] Crícia Z Felício, Klérisson VR Paixão, Celia AZ Barcelos, and Philippe Preux. 2017. A Multi-Armed Bandit Model Selection for Cold-Start User Recommendation. In *25th Conference on User Modeling, Adaptation and Personalization*. 32–40.

[12] Ignacio Fernández-Tobías, Matthias Braunhofer, Mehdi Elahi, Francesco Ricci, and Iván Cantador. 2016. Alleviating the New User Problem in Collaborative Filtering by Exploiting Personality Information. *User Modeling and User-Adapted Interaction* 26, 2 (2016), 221–255.

[13] Carlos A Gomez-Uribe and Neil Hunt. 2015. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems* 6, 4 (2015), 1–19.

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT.

[15] Jyotirmoy Gope and Sanjay Kumar Jain. 2017. A Survey on Solving Cold Start Problem in Recommender Systems. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 133–138.

[16] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *IEEE Data Engineering Bulletin* (2017).

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.

[18] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *39th Int. ACM SIGIR conf. on Research and Development in Information Retrieval*.

[19] Kurt Jacobson, Vidhya Murali, Edward Newett, Brian Whitman, and Romain Yon. 2016. Music Personalization at Spotify. In *RecSys*.

[20] Yehuda Koren and Robert Bell. 2015. Advances in Collaborative Filtering. *Recommender Systems Handbook* (2015), 77–118.

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[22] Maciej Kula. 2015. Metadata Embeddings for User and Item Cold-Start Recommendations. *arXiv:1507.08439* (2015).

[23] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing Cold-Start Problem in Recommendation Systems. In *International Conference on Ubiquitous Information Management and Communication*. 208–211.

[24] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *25th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*.

[25] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. *Advances in Neural Information Processing Systems* 27 (2014).

[26] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. 2014. Facing the Cold Start Problem in Recommender Systems. *Expert Systems with Applications* 41, 4 (2014), 2065–2073.

[27] Jovian Lin, Kazunari Sugiyama, Min-Yen Kan, and Tat-Seng Chua. 2013. Addressing Cold-Start in App Recommendation: Latent User Models Constructed from Twitter Followers. In *Proceedings of the 36th international ACM SIGIR Conference on Research and Development in Information Retrieval*. 283–292.

[28] Roderick JA Little and Donald B Rubin. 2019. *Statistical Analysis with Missing Data*. Vol. 793. John Wiley & Sons.

[29] Tomáš Mikolov, Édouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *International Conference on Language Resources and Evaluation (LREC 2018)*.

[30] A. Mongia, N. Jhamb, E. Chouzenoux, and A. Majumdar. 2020. Deep Latent Factor Model for Collaborative Filtering. *Signal Processing* 169 (2020), 107366.

[31] Ruihui Mu. 2018. A Survey of Recommender Systems based on Deep Learning. *IEEE Access* 6 (2018), 69009–69022.

[32] Yoon-Joo Park and Alexander Tuzhilin. 2008. The Long Tail of Recommender Systems and How To Leverage It. In *RecSys*. 11–18.

[33] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1532–1543.

[34] Guillaume Salha, Romain Hennequin, Jean-Baptiste Remy, Manuel Moussallam, and Michalis Vazirgiannis. 2021. FastGAE: Scalable Graph Autoencoders with Stochastic Subgraph Secoding. *Neural Networks* 142 (2021), 1–19.

[35] Guillaume Salha, Romain Hennequin, Viet Anh Tran, and Michalis Vazirgiannis. 2019. A Degeneracy Framework for Scalable Graph Autoencoders. In *28th International Joint Conference on Artificial Intelligence*.

[36] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. 2018. Current Challenges and Visions in Music Recommender Systems Research. *Int. Journal of Multimedia Information Retrieval* 7, 2 (2018), 95–116.

[37] Bracha Shapira, Lior Rokach, and Shirley Freilikhman. 2013. Facebook single and cross domain data for recommendation systems. *User Modeling and User-Adapted Interaction* 23, 2-3 (2013), 211–247.

[38] Lei Shi, Wayne Xin Zhao, and Yi-Dong Shen. 2017. Local Representative-Based Matrix Factorization for Cold-Start Recommendation. *ACM Transactions on Information Systems (TOIS)* 36, 2 (2017), 1–28.

[39] Brent Smith and Greg Linden. 2017. Two Decades of Recommender Systems at Amazon.com. *IEEE Internet Computing* 21, 3 (2017), 12–18.

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[41] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence* 2009 (2009).

[42] A. Van Den Oord, S. Dieleman, and B. Schrauwen. 2013. Deep Content-Based Music Recommendation. In *Adv. in Neural Information Processing Systems*, Vol. 26.

[43] Joaquin Vanschoren. 2018. Meta-Learning: A Survey. *arXiv:1810.03548* (2018).

[44] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. 2017. Dropoutnet: Addressing Cold Start in Recommender Systems. In *NeurIPS*. 4957–4966.

[45] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1235–1244.

[46] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-Scale Commodity Embedding for E-Commerce Recommendation in Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 839–848.

[47] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z Sheng, Mehmet Orgun, Longbing Cao, Francesco Ricci, and Philip S Yu. 2021. Graph Learning Based Recommender Systems: A Review. In *IJCAI*.

[48] Ling Yanxiang, Guo Deke, Cai Fei, and Chen Honghui. 2013. User-Based Clustering with Top-n Recommendation on Cold-Start Problem. In *Third International Conference on Intelligent System Design and Engineering Applications*. 1585–1589.

[49] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.

[50] J. Zhang, X. Shi, Sh. Zhao, and I. King. 2019. STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems. In *IJCAI*.

[51] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.