Scaling Up Collaborative Filtering Data Sets through Randomized Fractal Expansions

Francois Belletti
Google AI
belletti@google.com

Karthik Lakshmanan
Google AI
lakshmanan@google.com

Walid Krichene
Google AI
walidk@google.com

Nicolas Mayoraz
Google AI
nmayoraz@google.com

Yi-Fan Chen
Google AI
yifanchen@google.com

John Anderson
Google AI
janders@google.com

Taylor Robie
Google AI
taylorrobie@google.com

Tayo Oguntebi
Google AI
tayo@google.com

Dan Shirron
Intel Nervana
dan.shirron@intel.com

Amit Bleiwess
Intel Nervana
amit.bleiweiss@intel.com

Abstract—Recommender system research suffers from a disconnect between the size of academic data sets and the scale of industrial production systems. In order to bridge that gap, we propose to generate large-scale user/item interaction data sets by expanding pre-existing public data sets. Our key contribution is a technique that expands user/item incidence matrices matrices to large numbers of rows (users), columns (items), and non-zero values (interactions). The proposed method adapts Kronecker Graph Theory to preserve key higher order statistical properties such as the fat-tailed distribution of user engagements, item popularity, and singular value spectra of user/item interaction matrices. Preserving such properties is key to building large realistic synthetic data sets which in turn can be employed reliably to benchmark recommender systems and the systems employed to train them. We further apply our stochastic expansion algorithm to the binarized MovieLens 20M data set, which comprises 20M interactions between 27K movies and 138K users. The resulting expanded data set has 1.2B ratings, 2.2M users, and 855K items, which can be scaled up or down.

Index Terms—Machine Learning, Deep Learning, Recommender Systems, Graph Theory, Simulation

I. INTRODUCTION

Machine Learning (ML) benchmarks compare the capabilities of models, distributed training systems and linear algebra accelerators on realistic problems at scale. For these benchmarks to be effective, results need to be reproducible by many different groups which implies that publicly shared data sets need to be available.

Unfortunately, while recommendation systems constitute a key industrial application of ML at scale, large public data sets recording user/item interactions on online platforms are not yet available. For instance, although the Netflix data set [6] and the MovieLens data set [19] are publicly available, they are orders of magnitude smaller than proprietary data [3], [11], [56].

Proprietary data sets and privacy: While releasing large anonymized proprietary recommendation data sets may seem an acceptable solution from a technical standpoint, it is a nontrivial problem to preserve user privacy while still maintaining useful characteristics of the dataset. For instance, [37] shows a privacy breach of the Netflix prize dataset. More importantly, publishing anonymized industrial data sets runs counter to user

TABLE I: Size of MovieLens 20M [19] vs industrial dataset in [56].

	MovieLens 20M	Industrial
#users	138K	Hundreds of Millions
#items	27K	2M
#topics	19	600K
#observations	20M	Hundreds of Billions

expectations that their data may only be used in a restricted manner to improve the quality of their experience on the platform.

Therefore, we decide not to make user data more broadly available to preserve the privacy of users. We instead choose to produce synthetic yet realistic data sets whose scale is commensurate with that of our production problems while only consuming already publicly available data.

Producing a realistic binary MovieLens 10 billion+ dataset: In this work, we focus on the MovieLens dataset which only entails movie ratings posted publicly by users of the MovieLens platform. The MovieLens data set has now become a standard benchmark for academic research in recommender systems. Many recent research articles rely on MovieLens [1], [7], [20], [22], [26], [32], [34], [38], [44], [49], [50], [55], [57], [59]. The latest version of MovieLens [19] has accrued more than 800 citations according to Google Scholar. A binarized version of this dataset is obtained when all the ratings are substituted by 1.0 (proposed in Neural Collaborative Filtering [20]). While in previous work we have considered the original MovieLens data set comprising ratings on a discrete scale [4], we now focus on its binarized version. Although the binarized version is representative of industrial collaborative filtering aiming at predicting which item a given user is most likely to view [11], the data set still only entails few observed interactions and more importantly a very small catalogue of users/items, compared to industrial proprietary recommendation data.

Industrial recommender systems typically have to nominate items from catalogues comprising several million distinct

elements. The large number of observations collected by online platforms about user/item interactions also enables performance gains by increasing the dimension of the embeddings employed to represent items. In most modern ML recommendations, the model learns a vector valued representer in $\mathbb{R}^d$ for each of the I users/items of the catalog. Typically each user and item is represented with $d \sim 1e^2 \ldots 1e^3$ scalars and $I \sim 1e^6 \ldots 1e^9$ elements are present in user set and in the item set. Storing, accessing and training such vast embedding tables presents unique challenges as large tables will no longer easily fit in the memory of a single machine: distributed embedding tables are often necessary to store the learned embedding tables; hierarchical embedding access strategies such as hierarchical softmax [56] or differentiated softmax [17] provide better data structures and learning paradigms; an appropriate negative sampling strategy [11] or regularization [26] is needed to solve the extreme classification problem selecting one item from the catalog constitutents. By scaling up the public MovieLens data set, we want to move the problem into a regime where such issues are critical so that the corresponding benchmark is helpful for industrial applications.

In order to provide a new data set — more aligned with the needs of production scale recommender systems — we therefore aim at expanding publicly available data by creating a realistic surrogate. The following constraints help create a production-size synthetic recommendation problem similar and at least as hard an ML problem as the original one for matrix factorization approaches to recommendations [20], [25]:

- orders of magnitude more users and items are present in the synthetic dataset;
- the synthetic dataset is realistic in that its first and second order statistics match those of the original dataset presented in Figure 1.

Key first and second order statistics of interest we aim to preserve are summarized in Figure 1 — the details of their computation are given in Section IV.

Adapting Kronecker Graph expansions to binarized user/item interactions: We employ the Kronecker Graph Theory introduced in [28] to achieve a suitable fractal expansion of recommendation data to benchmark linear and non-linear user/item factorization approaches for recommendations [20], [25]. Consider a recommendation problem comprising m users and n items. Let $(R_{i,j})_{i=1\ldots m, j=1\ldots n}$ be the sparse matrix of binarized recorded interactions (i.e. 1 if user i has consumed item j and 0 otherwise). The key insight we develop in the present paper is that a carefully crafted fractal expansion of R can preserve high level statistics of the original data set while scaling its size up by multiple orders of magnitudes.

Many different transforms can be applied to the matrix R which can be considered a standard sparse binary 2 dimensional image. A recent approach to creating synthetic recommendation data sets consists in making parametric assumptions on user behavior by instantiating a user model interacting with an online platform [10], [46]. Unfortunately, such methods (even calibrated to reproduce empirical facts
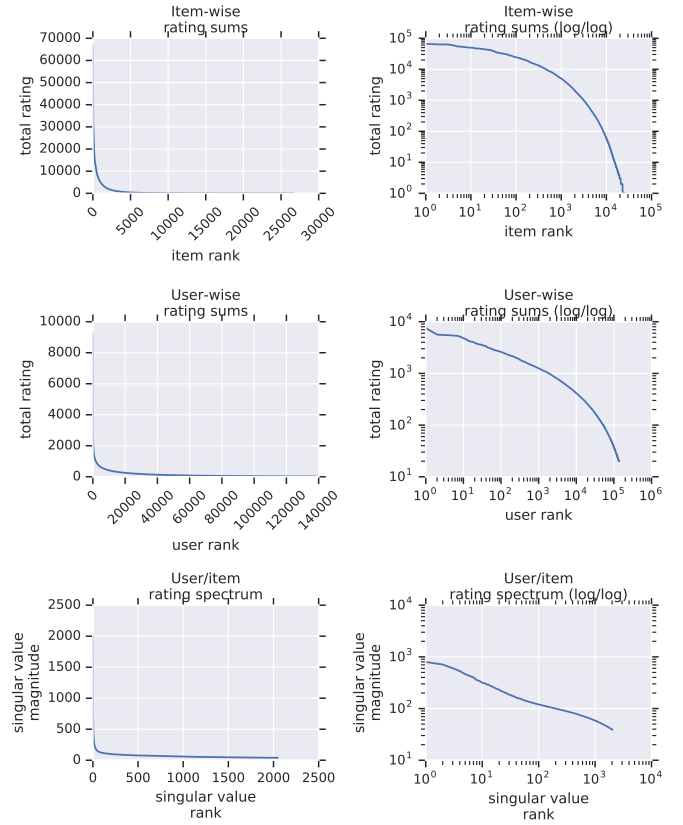

Fig. 1: Key first and second order properties of the binarized MovieLens 20m user/item rating matrix we aim to preserve while synthetically expanding the data set. **Top:** item popularity distribution (total ratings of each item). **Middle:** user engagement distribution (total ratings of each user). **Bottom:** dominant singular values of the rating matrix (core to the difficulty of matrix factorization tasks).

in actual data sets) do not provide strong guarantees that the resulting interaction data is similar to the original. A challenging problem in this domain is to build user models that can provide such guarantees, which can be validated using online experiments. In this work, instead of simulating recommendations in a parametric user-centric way as in [10], [46], we choose a non-parametric approach operating directly in the space of user/item affinity. In order to synthesize a large realistic dataset in a principled manner, we adapt the Kronecker expansions which have previously been employed to produce large realistic graphs in [28]. We employ a non-parametric randomized simulation of the evolution of the user/item bi-partite graph to create a large synthetic data set. It is noteworthy that as opposed to our original approach in [4] — where we put emphasis on analytic tractability — we now employ a method that loses some analytic tractability but still preserves key statistics of the data set. Furthermore, we show how randomized operations help address limitations of the previous method which yielded an interaction matrix with a

discernible block-wise repetitive structure.

While Kronecker Graphs Theory is developed in [28], [29] on square adjacency matrices, the Kronecker product operator is well defined on rectangular matrices and therefore we can apply a similar technique to user/item interaction data sets — which was already noted in [29] but not developed extensively. As in [29] we will use a stochastic version of the Kronecker extension for a binary original matrix. The Kronecker Graph generation paradigm has to be changed with the present data set in other aspects. However, we need to decrease the expansion rate to generate data sets with the scale we desire, not orders of magnitude too large. We need to do so while maintaining key conservation properties of the original algorithm [29]. Furthermore, we introduce a new block-wise shuffling to randomize the Kronecker operator and yield a data set more helpful to train ML models for recommendations.

In order to reliably employ Kronecker based fractal expansions on recommender system data we devise the following contributions:

- we develop a new technique based on linear algebra to adapt fractal Kronecker expansions to recommendation problems;
- we introduce a randomly shuffled extension of the original Kronecker product to prevent block-wise structural repetitions and take steps to prevent test data from leaking into the data set employed to train collaborative filtering models;
- we also show that the resulting algorithm we develop is scalable and easily parallelizable as we employ it on the actual MovieLens 20 million dataset;
- we produce a synthetic yet realistic MovieLens 1.2 billion dataset to help recommender system research scale up in computational benchmark for model training;
- we demonstrate that key recommendation system specific properties of the original dataset are preserved by the deterministic version of our technique;
- we make the corresponding open source code available so that other researchers may reproduce our findings and tailor the generated synthetic data to their needs.

The present article is organized as follows: First, we describe prior research on ML for recommendations and large synthetic dataset generation. Next, we develop a randomized adaptation of Kronecker Graphs to user/item interaction matrices and prove key theoretical properties. Finally, we employ the resulting algorithm experimentally to MovieLens 20m data to validate its statistical properties.

II. RELATED WORK

Recommender systems constitute the workhorse of many e-commerce, social networking and entertainment platforms. In the present paper we focus on the classical setting where the key role of a recommender system is to suggest relevant items to a given user. Although other approaches are very popular such as content based recommendations [43] or social recommendations [9], collaborative filtering remains a prevalent approach to the recommendation problem [33], [42], [45].

Collaborative filtering: The key insight behind collaborative filtering is to learn affinities between users and items based on previously collected user/item interaction data. Collaborative filtering exists in different flavors. Neighborhood methods group users by inter-user similarity and will recommend items to a given user that have been consumed by neighbors [43]. Latent methods try to decompose user/item affinity as the result of the interaction of a few underlying representative factors characterizing the user and the item (e.g. Principal Component Analysis [23], Latent Dirichlet Allocation [8]). Matrix factorization [25] is a Latent Factor Method that relies on solving the matrix completion problem to recommend items for users.

The matrix factorization approach represents the affinity $a_{i,j}$ between a user i and an item j with an inner product $x_i^T y_j$ where x_i and y_j are two vectors in $\mathbb{R}^d$ representing the user and the item respectively. Given a sparse matrix of user/item interactions $R = (r_{i,j})_{i=1...m,j=1...n}$, user and item factors can therefore be learned by approximating R with a low rank matrix XY^T where $X \in \mathbb{R}^{m,k}$ entails the user factors and $Y \in \mathbb{R}^{n,k}$ contains the item factors. The data set R represents ratings as in the MovieLens dataset [19] or item consumption ($r_{i,j} = 1$ if and only if the user i has consumed item j [6]) — the latter being considered here. The matrix factorization approach is an example of a solution to the rating matrix completion problem which aims at predicting the rating of an item j by a user i which has not been observed yet and corresponds to a value of 0 in the sparse original rating matrix. Such a factorization method learns an approximation of the data that preserves a few higher order properties of the rating matrix R. In particular, the low rank approximation tries to mimic the singular value spectrum of the original data set. We draw inspiration from matrix factorization to tackle synthetic data generation. The present paper will adopt a similar approach to extend collaborative filtering data-sets. Besides trying to preserve the spectral properties of the original data, we operate under the constraint of conserving its first and second order statistical properties.

Deep Learning for recommender systems: Collaborative filtering has known many recent developments which motivate our objective of expanding public data sets in a realistic manner. Deep Neural Networks (DNNs) are now becoming common in both non-linear matrix factorization tasks [11], [20], [52] and sequential recommendations [18], [47], [58]. The mapping between user/item pairs and ratings is generally learned by training the neural model to predict user behavior on a large data set of previously observed user/item interactions.

DNNs consume large quantities of data and are computationally expensive to train, therefore they give rise to commonly shared benchmarks aimed at speeding up training. For training, a Stochastic Gradient Descent method is employed [27] which requires forward model computation and back-propagation to be run on many mini-batches of (user, item, score) examples. The matrix completion task still consists in predicting a rating for the interaction of user i and

item j although (i, j) has not been observed in the original data-set. The model is typically run on billions of examples as the training procedure iterates over the training data set.

Freshness in recommender systems: Model freshness is generally critical to industrial recommendations [11] which implies that only limited time is available to re-train the model on newly available data. The throughput of the trainer is therefore crucial to providing more engaging recommendation experiences and presenting more novel items. Unfortunately, public recommendation data sets are too small to provide training-time-to-accuracy benchmarks that can be realistically employed for industrial applications. Too few different examples are available in MovieLens 20m for instance and the number of different available items is orders of magnitude too small. In many industrial settings, millions of items (e.g. products, videos, songs) have to be taken into account by recommendation models. The recommendation model learns an embedding matrices of size (N, d) where $d \sim 10 - 10^3$ and $N \sim 10^6 - 10^9$ are typical values. As a consequence, the memory footprint of this matrix may dominate that of the rest of the model by several orders of magnitude. During training, the latency and bandwidth of the access to such embedding matrices have a prominent influence on the final throughput in examples/second. Such computational difficulties associated with learning large embedding matrices are worthwhile solving in benchmarks. A higher throughput enables training models with more examples which enables better statistical regularization and architectural expressiveness. The multi-billion interaction size of the data set used for training is also a major factor that affects modeling choices and infrastructure development in the industry.

III. FRACTAL EXPANSIONS OF USER/ITEM INTERACTION DATA SETS

The present section delineates the insights orienting our design decisions when expanding public recommendation data sets.

1) Self-similarity in user/item interactions: Interactions between users and items follow a natural hierarchy in data sets where items can be organized in topics, genres, and categories [56]. There is for instance an item-level fractal structure in MovieLens 20m with a tree-like structure of genres, sub-genres, and directors. If users were clustered according to their demographics and tastes, another hierarchy would be formed [43]. The corresponding structured user/item interaction matrix is illustrated in Figure 2. The hierarchical nature of user/item interactions (topical and demographic) makes the recommendation data set structurally self-similar (i.e. patterns that occur at more granular scales resemble those affecting coarser scales [36]).

One can therefore build a user-group/item-category incidence matrix with user-groups as rows and item-categories as columns — a coarse interaction matrix. As each user group consists of many individuals and each item category comprises multiple movies, the original individual level user/item interaction matrix may be considered as an expanded version of the
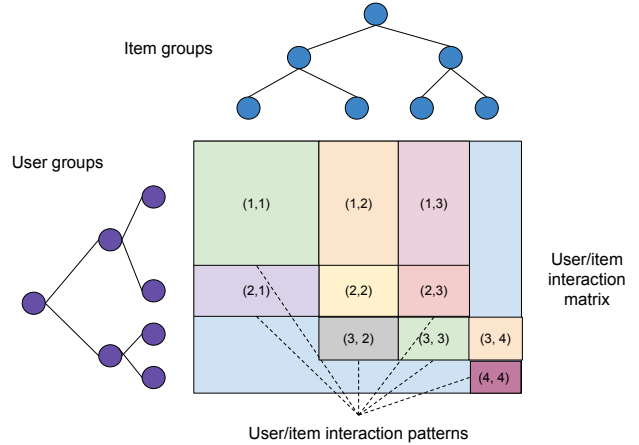

Fig. 2: Typical user/item interaction patterns in recommendation data sets. Self-similarity appears as a natural key feature of the hierarchical organization of users and items into groups of various granularity.

coarse interaction matrix. We choose to expand the user/item interaction matrix by extrapolating this self-similar structure and simulating its growth to yet another level of granularity: original items and users are considered fictional topic and user groups in the expanded data set.

A key advantage of this fractal procedure is that it may be entirely non-parametric and designed to preserve high level properties of the original dataset. In particular, a fractal expansion re-introduces the patterns originally observed in the entire real dataset within each block of local interactions of the synthetic user/item matrix. By carefully designing the way such blocks are produced and laid out, we can therefore hope to produce a realistic yet much larger rating matrix. In the following, we show how the Kronecker operator enables such a construction.

2) Fractal expansion through Kronecker products: The Kronecker product — denoted $\otimes$ — is a non-standard matrix operator with an intrinsic self-similar structure:

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} \quad (1)$$

where $A \in \mathbb{R}^{m,n}$, $B \in \mathbb{R}^{p,q}$ and $A \otimes B \in \mathbb{R}^{mp,nq}$.

In the original presentation of Kronecker Graph Theory [28] as well as the stochastic extension [35] and the extended theory [29], the Kronecker product is the core operator enabling the synthesis of graphs with exponentially growing adjacency matrices. As in the present work, the insight underlying the use of Kronecker Graph Theory in [29] is to produce large synthetic yet realistic graphs. The fractal nature of the Kronecker operator as it is applied multiple times (see Figure 2 in [29] for an illustration) fits the self-similar statistical

properties of real world graphs such as the internet, the web or social networks [28].

If A is the adjacency matrix of the original graph, fractal expansions are created in [28] by chaining Kronecker products as follows:

$$A \otimes A \ldots A.$$

As adjacency matrices are square, Kronecker Graphs are not employed on rectangular matrices in pre-existing work although the operation is well defined. We show that these differences do not prevent Kronecker products from preserving core properties of binarized rating matrices. A more important challenge is the size of the original matrix we deal with: $R \in \mathbb{R}^{(138 \times 10^3, 27 \times 10^3)}$. A naive Kronecker expansion would therefore synthesize a rating matrix with 19 billion users which is too large.

Thus, although Kronecker products seem like an ideal candidate for the mechanism at the core of the self-similar synthesis of a larger recommendation dataset, some modifications are needed to the algorithms developed in [29].

3) Reduced Kronecker expansions: We choose to synthesize a user/item rating matrix

$$\widetilde{R} = \widehat{R} \otimes R$$

where $\widehat{R}$ is a matrix derived from R but much smaller (for instance $\widehat{R} \in \mathbb{R}^{128,256}$). For reasons that will become apparent as we explore some theoretical properties of Kronecker fractal expansions, we want to construct a smaller derived matrix $\widehat{R}$ that shares similarities with R. In particular, we seek $\widehat{R}$ with a similar row-wise sum distribution (user engagement distribution), column-wise distribution (item engagement distribution) and singular value spectrum (signal to noise ratio distribution in the matrix factorization).

4) Implementation at scale and algorithmic extensions: Computing a Kronecker product between two matrices A and B is an inherently parallel operation. It is sufficient to broadcast B to each element (i, j) of A and then multiply B by $a_{i,j}$. Such a property implies that scaling can be achieved. Another advantage of the operator is that even a single machine can produce a large output data-set by sequentially iterating on the values of A. Only storage space commensurable with the size of the original matrix is needed to compute each block of the Kronecker product. It is noteworthy that generalized fractal expansions can be defined by altering the standard Kronecker product. We consider such extensions here as candidates to engineer more challenging synthetic data sets. One drawback though is that these extensions may not preserve analytic tractability.

A first generalization defines a binary operator $\otimes_F$ with $F : \mathbb{R} \times \mathbb{R}^{m,n} \times \mathbb{N} \to \mathbb{R}^{p,q}$ as follows:

$$A \otimes_F B = \begin{bmatrix} F(a_{11}, B, \omega_{11}) & \ldots & F(a_{1n}, B, \omega_{1n}) \\ \vdots & \ddots & \vdots \\ F(a_{m1}, B, \omega_{m1}) & \ldots & F(a_{mn}, B, \omega_{mn}) \end{bmatrix} \quad (2)$$

where $\omega_{11}, \ldots, \omega_{mn}$ is a sequence of pseudo-random numbers. Including randomization and non-linearity in F appears

as a simple way to synthesize data sets entailing more varied patterns. The algorithm we employ to compute Kronecker products is presented in Algorithm 1. The implementation we employ is trivially parallelizable. We only create a list of Kronecker blocks to dump entire rows (users) of the output matrix to file. This is not necessary and can be removed to enable as many processes to run simultaneously and independently as there are elements in $\widehat{R}$ (provided pseudo random numbers are generated in parallel in an appropriate manner).

Algorithm 1 Kronecker fractal expansion $\odot_F$
---
for $i = 1$ to m **do**
   kBlocks $\leftarrow$ empty list
   for $j = 1$ to n **do**
      $\omega \leftarrow$ next pseudo random number
      kBlock $\leftarrow F(\widehat{R}(i, j), R, w)$
      kBlocks append kBlock
   end for
   outputToFile(kBlocks)
end for

The only reason why we need a reduced version $\widehat{R}$ of R is to control the size of the expansion. Also, $A \otimes B$ and $B \otimes A$ are equal after a row-wise and a column-wise permutation. Therefore, another family of appropriate extensions may be obtained by considering

$$B \otimes_G A = \begin{bmatrix} b_{11}G(A, \omega_{11}) & \ldots & b_{1n}G(A, \omega_{1n}) \\ \vdots & \ddots & \vdots \\ b_{m1}G(A, \omega_{m1}) & \ldots & b_{mn}G(A, \omega_{mn}) \end{bmatrix} \quad (3)$$

where $G : \mathbb{R}^{m,n} \times \mathbb{N} \to \mathbb{R}^{m',n'}$ is a randomized sketching operation on the matrix A which reduces its size by several orders of magnitude. A trivial scheme consists in sampling a small number of rows and columns from A at random. Other random projections [2], [14], [31] may of course be used. The randomized procedures above produce a user/item interaction matrix where there is no longer a block-wise repetitive structure. Less obvious statistical patterns can give rise to more challenging synthetic large-scale collaborative filtering problems.

5) Stochastic Kronecker product and dropout for binary rating matrices: As opposed to our original approach [4] which focused on item ratings, we now consider an original binary rating R. In such a setting, a standard Kronecker product is not suitable as the ratings all take the same value of 1 and therefore multiplications by elements of $\widehat{R}$ do not produce ratings that are all still binary. We instead use the stochastic Kronecker graph approach from [29] and employ the reduced matrix's elements $\widehat{R}$ as dropout rates over the matrix R. When computing the block i, j of the expanded rating matrix, instead of using $\widehat{R}_{i,j}R$, we instead consider dropout$(R, rate = \widehat{R}_{i,j})$ after having re-scaled $\widehat{R}$ so that all its elements are in $[0, 1]$. For each element k, l of R, the dropout function for a rate $\widehat{R}_{i,j}$ samples independently from a Bernoulli distribution with parameter $\widehat{R}_{i,j}$. If the sampled

number is 1 $R_{k,l}$, is kept unchanged, otherwise it is dropped and set to 0. Such a *block* dropout operator enjoys statistical properties that are similar to the Kronecker product [29] while being readily employable on binary data-sets. The stochastic Kronecker product we devise can therefore be written as follows in matrix notation:

$$A \otimes_{\text{rand}} B =$$
$$\begin{bmatrix} \text{Sh}\left(\text{drop}(1 - a_{11}, B)\right) & \dots & \text{Sh}\left(\text{drop}(1 - a_{1n}, B)\right) \\ \vdots & \ddots & \vdots \\ \text{Sh}\left(\text{drop}(1 - a_{m1}, B)\right) & \dots & \text{Sh}\left(\text{drop}(1 - a_{mn}, B)\right) \end{bmatrix}$$

where "Sh" denotes the random row-wise and column-wise shuffling operator and "drop" denotes the dropout operator whose first argument is the dropout rate and whose second argument is the matrix from which to zero out elements at random. Algorithm 2 exposes the implementation of the randomized Kronecker product $\odot_{\text{rand}}$.

6) Randomized shuffling and Kronecker SVD: Another limitation of the synthetic data set initially presented in [4] is the block-wise repetitive structure of Kronecker products. Although the synthetic data set is still hard to factorize as the product of two low rank matrices because its singular values are still distributed similarly to the original data set, it is now easy to factorize with a Kronecker SVD [24] which takes advantage of the block-wise repetitions in Eq (1). Randomized fractal expansions which presented in Eq (2) and Eq (3) address this issue. The approach we adopt consists in shuffling rows and columns of each block in Eq (1) independently at random. The shuffles will break the block-wise repetitive structure and prevent Kronecker SVD from producing a trivial solution to the factorization problem.

As a result, the expansion technique we present appears as a reliable first candidate to train linear matrix factorization models [43] and non-linear user/item similarity scoring models [20].

Algorithm 2 Kronecker fractal expansion for binary data sets with random shuffling and dropout $\odot_{\text{rand}}$

---

  for $i = 1$ to m **do**
    kBlocks $\leftarrow$ empty list
    for $j = 1$ to n **do**
      kBlock $\leftarrow$ dropout$(R, rate = \widehat{R}(i,j))$
      kBlock $\leftarrow$ shuffleColumnsAndRows(kBlock)
      kBlocks append kBlock
    end for
    outputToFile(kBlocks)
  end for

---

7) Preventing leaks from the test set into the training set: For matrix factorization tasks, the usual procedure to build disjoint training and test data sets for Movie Lens consists in selecting some ratings and removing them from the training set while adding them to the test set. A naive adaptation of the test data generation procedure to our extended data set would select test items directly on the larger matrix $\widetilde{R}$. Unfortunately,

as $\widetilde{R} = \widehat{R} \otimes R$, such a procedure would implicitly share data between interactions of the training and test sets through $\widehat{R}$ which incorporates information from the entire original data set R. In order to generate training and test data without leaking test data into the training set, we proceed as follows. We consider two separate training and test sets selected from the original data set R: R_{train} and R_{test}. With $R \in \mathbb{R}^{m,n}$, we have $R_{\text{train}} \in \mathbb{R}^{m,n}$ and $R_{\text{test}} \in \mathbb{R}^{m,n}$. For the MovieLens data set, where each rating of a given item by a specific user is timestamped, a typical approach to defining training and testing sets removes the last rating of each user from the train set and adds it to the test set. Such a procedure outputs a R_{test} matrix with much fewer non zero elements than R_{train}.

The smaller matrix $\widehat{R_{\text{train}}}$ is now derived from R_{train} exclusively, without incorporating any data from R_{test}. We create the extended versions of the train and test data sets separately as follows:

$$\widetilde{R_{\text{train}}} = \widehat{R_{\text{train}}} \odot R_{\text{train}} \text{ and } \widetilde{R_{\text{test}}} = \widehat{R_{\text{train}}} \odot R_{\text{test}}.$$

By construction, the procedure prevents test data from leaking into the train data and implicitly informing the model of patterns that will be present in the test set during training.

8) Consistent randomized operations across training and testing sets: With a stochastic Kronecker $\odot_{\text{rand}}$ featuring dropout and block-wise shuffling, additional precautions need to be taken. In order to guarantee that the randomized shuffles of rows and columns are consistent between the training and testing data, we flip the sign of the test elements in the rating matrix to keep track of their belonging to the test set. We apply all randomized operations to the resulting matrix comprising elements in $\{-1, 0, 1\}$:

$$\widetilde{R_{\text{temp}}} = \widehat{R_{\text{train}}} \odot_{\text{rand}} (R_{\text{train}} - R_{\text{test}})$$

where $R_{\text{train}} \in \mathbb{R}^{m,n}$ and $R_{\text{test}} \in \mathbb{R}^{m,n}$. The positive elements of R_{temp} are attributed to $\widetilde{R_{\text{train}}}$ and the negative elements are attributed to $\widetilde{R_{\text{test}}}$ after having flipped their sign:

$$\forall i \in \{1, \dots, mp\}, j \in \{1, \dots, nq\},$$
$$\widetilde{R_{\text{train}}}_{i,j} = \widetilde{R_{\text{temp}}}_{i,j} \text{ if } \widetilde{R_{\text{temp}}}_{i,j} = 1 \text{ else } 0$$
$$\widetilde{R_{\text{test}}}_{i,j} = -\widetilde{R_{\text{temp}}}_{i,j} \text{ if } \widetilde{R_{\text{temp}}}_{i,j} = -1 \text{ else } 0$$

Such an operation is simple and guarantees the consistency of randomized shuffles of the sub-blocks in the extended matrices $\widetilde{R_{\text{train}}}$ and $\widetilde{R_{\text{test}}}$.

IV. STATISTICAL PROPERTIES OF KRONECKER FRACTAL EXPANSIONS

After having introduced Kronecker products to self-similarly expand a recommendation dataset into a much larger one, we now develop theoretical insights about how the transform preserves crucial common properties with the original.

1) Salient empirical facts in MovieLens data: First, we introduce the critical properties we want to preserve. As a user/item interaction dataset on an online platform, one expects MovieLens to feature common properties of recommendation data sets such as power-law or fat-tailed distributions [56] (a power-law or fat-tailed distribution over positive values behaves like $x \to \alpha x^{-\beta}$ for large enough values of x with $\alpha > 0$ and $\beta > 0$).

First important statistical properties for recommendations concern the distribution of interactions across users and across items. It is generally observed that such distributions exhibit a power-law behavior [1], [13], [15], [30], [39], [40], [53], [56]. To characterize such a behavior in the MovieLens data set, we take a look at the distribution of the total ratings along the item axis and the user axis. In other words, we compute row-wise and column-wise sums for the rating matrix R and observe their distributions. The corresponding ranked distributions are exposed in Figure 1 and do exhibit a clear power-law behavior for rather popular items. However, we observe that tail items have a higher popularity decay rate. Similarly, the engagement decay rate increases for the group of less engaged users.

The other approximate power-law we find in Figure 1 lies in the singular value spectrum of the MovieLens dataset. We compute the top k singular values [21] of the MovieLens rating matrix R by power iteration, which can scale to its large $(138K, 27K)$ dimension. The method yields the dominant singular values of R and the corresponding singular vectors so that one can classically approximate R by $R \simeq U\Sigma V$ where Σ is diagonal of dimension (k, k), U is column-orthogonal of dimension (m, k) and V is row-orthogonal of dimension (k, n) — which yields the rank k matrix closest to R in Frobenius norm.

Examining the distribution of the 2048 top singular values of R in the MovieLens dataset (which has at most $27K$ non-zero singular values) in Figure 1 highlights a clear power-law behavior in the highest magnitude part of the spectrum of R. We observe in the spectral distribution an inflection for smaller singular values whose magnitude decays at a higher rate than larger singular values. Such a spectral distribution is as a key feature of the original dataset. This property is particularly important for low-rank approximation approaches to the matrix completion problem, which have to choose a sufficiently large rank for approximating the observations. Therefore, we also want the expanded dataset to exhibit a similar behavior in terms of spectral properties.

In all the high level statistics we present, we want to preserve the approximate power-law decay as well as its inflection for smaller values. Our requirements for the expanding transform which we apply to R are therefore threefold: *we want to preserve the distributions of row-wise sums of R, column-wise sums of R and singular value distribution of R.* Additional requirements, beyond first and second order high level statistics will further increase the confidence in the realism of the expanded synthetic dataset.

2) Analytic tractability through standard Kronecker products: Although we use a randomized version of the Kronecker

product which does not offer the same level of analytic tractability, the choice of such a transform is deeply anchored in some of the theoretical properties of the standard Kronecker product. We now expose how — in its standard deterministic version — the fractal transform design we rely on preserves the key statistical properties of the previous section.

Definition 1: Consider $A \in \mathbb{R}^{m,n} = (a_{i,j})_{i=1...m,j=1...n}$, we denote the set $\left\{ \sum_{j=1}^{n} a_{i,j} \right\}$ of row-wise sums of A by $\mathcal{R}(A)$, the set $\{\sum_{i=1}^{m} a_{i,j}\}$ of column-wise sums of A by $\mathcal{C}(A)$, and the set of non-zero singular values of A by $\mathcal{S}(A)$.

Definition 2: Consider an integer i and a non-zero positive integer p, we denote the integer part of $i-1$ in base p $\lfloor i-1 \rfloor_p = \lfloor \frac{i-1}{p} \rfloor$ and the fractional part $\{i-1\}_p = i - \lfloor i-1 \rfloor_p \times p$.

First we focus on conservation properties in terms of row-wise and column-wise sums which correspond respectively to marginalized user engagement and item popularity distributions. In the following, $\times$ denotes the Minkowski product of two sets, i.e. $A \times B = \{a \times b \mid \forall a \in A, \forall b \in B\}$.

Proposition 1: Consider $A \in \mathbb{R}^{m,n}$ and $B \in \mathbb{R}^{p,q}$ and their Kronecker product $K = A \otimes B$. Then

$$\mathcal{R}(K) = \mathcal{R}(A) \times \mathcal{R}(B) \text{ and } \mathcal{C}(K) = \mathcal{C}(A) \times \mathcal{C}(B).$$

Proof 1: Consider the ith row of K, by definition of K the corresponding sum can be rewritten as follows: $\sum_{j=1...np} k_{i,j} = \sum_{j=1...np} a_{\lfloor i-1 \rfloor_p + 1, \lfloor j-1 \rfloor_q + 1} b_{\{i-1\}_p, \{j-1\}_q}$ which in turn equals

$$\sum_{j=1...n} \sum_{j'=1...q} a_{\lfloor i-1 \rfloor_p + 1, j} b_{\{i-1\}_p, j'}.$$

Refactoring the two sums concludes the proof for the row-wise sum properties. The proof for column-wise properties is identical. ∎

Theorem 1: Consider $A \in \mathbb{R}^{m,n}$ and $B \in \mathbb{R}^{p,q}$ and their Kronecker product $K = A \otimes B$. Then

$$\mathcal{S}(K) = \mathcal{S}(A) \times \mathcal{S}(B).$$

Proof 2: One can easily check that $(XY) \otimes (VW) = (X \otimes V)(Y \otimes W)$ for any quadruple of matrices X, Y, V, W for which the notation makes sense and that $(X \otimes Y)^T = X^T \otimes Y^T$. Let $A = U_A \Sigma_A V_A$ be the SVD of B and $B = V_B \Sigma_B V_B$ the SVD of B. Then $(A \otimes B) = (U_A \otimes U_B)(\Sigma_A \otimes \Sigma_B)(V_A \otimes V_B)$. Now, $(U_A \otimes U_B)^T (U_A \otimes U_B) = (U_A^T \otimes U_B^T)(U_A \otimes U_B) = (U_A^T U_A) \otimes (U_B^T U_B^T)$. Writing the same decomposition for $(V_A \otimes V_B)(V_A^T \otimes V_B^T)$ and considering that U_A, U_B are column-orthogonal while V_A, V_B are row-orthogonal concludes the proof. ∎

The properties above imply that knowing the row-wise sums, column-wise sums and singular value spectrum of the reduced rating matrix $\widehat{R}$ and the original rating matrix R is enough to deduce the corresponding properties for the expanded rating matrix $\widetilde{R}$ — analytically. As in [29], the Kronecker product enables analytic tractability while expanding data sets in a fractal manner to orders of magnitude more data.

In practice, we use a randomized version of the Kronecker product whose block-wise shuffles do not have an analytically tractable effect of the high order statistics of the rating matrix. Therefore, we rely in section V-4 on a statistical examination of the properties of the extended synthetic data set we produce with our randomized fractal operator to verify that our original theoretical insights from the deterministic case are still valid. In particular, we demonstrate that original high order statistics of the new data set we produce preserve — as in our first deterministic approach [4] — the original properties of the binary MovieLens 20m data set.

3) Constructing a reduced $\widehat{R}$ matrix with a similar spectrum: Considering that the quasi power-law properties of R imply — as in [29] — that $\mathcal{S}(R) \times \mathcal{S}(R)$ has a similar distribution to $\mathcal{S}(R)$, we seek a small $\widehat{R}$ whose high order statistical properties are similar to those of R. As we want to generate a dataset with several billion user/item interactions, millions of distinct users and millions of distinct items, we are looking for a matrix $\widehat{R}$ with a few hundred or thousand rows and columns. The reduced matrix $\widehat{R}$ we seek is therefore orders of magnitude smaller than R. In order to produce a reduced matrix $\widehat{R}$ of dimensions $(1000, 1700)$ one could use the reduced size older MovieLens 100K dataset [19]. Such a dataset can be interpreted as a sub-sampled reduced version of MovieLens 20m with similar properties. These data sets have been collected seven years apart, wherein the characteristics of the dataset are not comparable. Also, we aim to produce an expansion method where the expansion multipliers can be chosen flexibly by practitioners. In our experiments, it is noteworthy that naive uniform user and item sampling strategies have not yielded smaller matrices $\widehat{R}$ with similar properties to R in our experiments. Different random projections [2], [14], [31] could more generally be employed. However, we rely on a procedure better tailored to our specific statistical requirements.

We now describe the technique we employed to produce a reduced size matrix $\widehat{R}$ with first and second order properties close to R which in turn led to constructing an expansion matrix $\widetilde{R} = \widehat{R} \otimes R$ similar to R. We want the dimensions of $\widehat{R}$ to be (m', n') with $m' << m$ and $n' << n$. Consider again the approximate Singular Value Decomposition (SVD) [21] of R with the $k = min(m', n')$ principal singular values of R:

$$R \simeq U \Sigma V \qquad (4)$$

where $U \in \mathbb{R}^{n,k}$ has orthogonal columns, $V \in \mathbb{R}^{k,m}$ has orthogonal rows, and $\Sigma \in \mathbb{R}^{k,k}$ is diagonal with non-negative terms.

To reduce the number of rows and columns of R while preserving its top k singular values a trivial solution would consist in replacing U and V by a small random orthogonal matrices with few rows and columns respectively. Unfortunately such a method would only seemingly preserve the spectral properties of R as the principal singular vectors would be widely changed. Such properties are important: one of the key advantages of employing Kronecker products in [29] is

the preservation of the network values, i.e. the distributions of singular vector components of a graph's adjacency matrix.

To obtain a matrix $\widetilde{U} \in \mathbb{R}^{n',k}$ with fewer rows than U but column-orthogonal and similar to U in the distribution of its values we use the following procedure. We re-size U down to n' rows with $n' < n$ by down-scaling through local averaging (using skimage.transform.resize in the scikit-image library [51]). Let $\bar{U} \in \mathbb{R}^{n',k}$ be the corresponding resized version of U. We then construct $\widetilde{U}$ as the column orthogonal matrix in $\mathbb{R}^{n',k}$ closest in Frobenius norm to $\bar{U}$. Therefore as in [16] we compute

$$\widetilde{U} = \bar{U} \left(\bar{U}^T \bar{U} \right)^{-1/2}. \qquad (5)$$

We apply a similar procedure to V to reduce its number of columns which yields a row orthogonal matrix $\widetilde{V} \in \mathbb{R}^{k,m'}$ with $m' < m$. The orthogonality of $\widetilde{U}$ (column-wise) and $\widetilde{V}$ (row-wise) guarantees that the singular value spectrum of

$$\widehat{R} = \widetilde{U} \Sigma \widetilde{V} \qquad (6)$$

consists exactly of the $k = min(m', n')$ leading components of the singular value spectrum of R. Like R, $\widehat{R}$ is re-scaled to take values in $[-1, 1]$. The whole procedure to reduce R down to $\widehat{R}$ is summarized in Algorithm 3.

Algorithm 3 Compute reduced matrix $\widehat{R}$

$(U, \Sigma, V) \leftarrow \text{sparseSVD}(R, k)$
$\bar{U} \leftarrow \text{imageResize}(U, n', k)$
$\bar{V} \leftarrow \text{imageResize}(V, k, m')$
$\widetilde{U} \leftarrow \bar{U} \left(\bar{U}^T \bar{U} \right)^{-1/2}$
$\widetilde{V} \leftarrow \left(\bar{V} \bar{V}^T \right)^{-1/2} \bar{V}$
$\widehat{R}_{temp} \leftarrow \widetilde{U} \Sigma \widetilde{V}$
$M \leftarrow max(\widehat{R}_{temp})$
$m \leftarrow min(\widehat{R}_{temp})$
return $\widehat{R}_{temp}/(M - m)$

We verify empirically that the distributions of values of the reduced singular vectors in $\widetilde{U}$ and $\widetilde{V}$ are similar to those of U and V respectively to preserve first order properties of R and value distributions of its singular vectors. Such properties are demonstrated through numerical experiments in the next section.

V. EXPERIMENTATION ON MOVIELENS 20 MILLION DATA

The MovieLens $20M$ data comprises $20M$ ratings given by $138K$ users to $27K$ items. In the present section, we demonstrate how the fractal Kronecker expansion technique we devised and presented helps scale up this dataset to orders of magnitude more users, items and interactions — all in a parallelizable manner.

1) Pre-processing of MovieLens 20m: The first pre-processing step we apply to MovieLens 20m is binarizing all the ratings: all the rating values are set to 1. Such a step is standard for tasks such as Neural Collaborative Filtering [20]. The second pre-processing step filters out users who have

	MovieLens 20m	Synthetic train set	Synthetic test set
Interactions	20 M	1.22 B	12.7 M
Users	138 K	2.20 M	2.20 M
Items	27 K	855 K	855 K

TABLE II: Size of the extended MovieLens20m data set

fewer than 2 ratings with distinct timestamps. The filter enables the splitting of MovieLens 20m into a train set R_{train} consisting of all the ratings of each users except the last one in chronological order and its complement. For each user, the rating with the latest timestamp is put in the test set R_{test}. After removal of users with too few ratings and splitting into training and test sets, we expand MovieLens 20m with the randomized Kronecker product presented in Algorithm 2.

2) Size of expanded data set: In the present experiments we construct a reduced rating matrix $\widehat{R}$ of size $(16, 32)$. The dropout based method in Algorithm 2 yields a new data set whose size is detailed in Table II.

Such a high number of interactions and items enable the training of deep neural collaborative models such as the Neural Collaborative Filtering model [20] with a scale which is now more representative of industrial settings. Moreover, the increased data set size helps construct benchmarks for deep learning software packages and ML accelerators that employ the same orders of magnitude as production settings in terms of user base size, item vocabulary size and number of observations.

3) Empirical properties of reduced $\widehat{R}$ matrix: The objective of the construction technique for $\widehat{R}$ was to produce a matrix sharing the properties of $R \otimes R$ though smaller in size ([29]). To that end, we aimed at constructing a matrix $\widehat{R}$ of dimension $(16, 32)$ with properties close to those of R in terms of column-wise sum, row-wise sum and singular value spectrum distributions.

We now check that the construction procedure we devised does produce a $\widehat{R}$ with the properties we expected. As the impact of the re-sizing step is unclear from an analytic standpoint, we had to resort to numerical experiments to validate our method.

In Figure 3, one can assess that the first and second order properties of R and $\widehat{R}$ match with high enough fidelity. In particular, the higher magnitude column-wise and row-wise sum distributions follow a "power-law" behavior similar to that of the original matrix. Similar observations can be made about the singular value spectra of $\widehat{R}$ and R.

There is therefore now a reasonable likelihood that our adapted Kronecker expansion — although somewhat differing from the method originally presented in [29] — will enjoy the same benefits in terms of enabling data set expansion while preserving high order statistical properties.

4) Empirical properties of the expanded data set $\widetilde{R}$: We now verify empirically that the expanded rating matrix $\widetilde{R} = \widehat{R} \otimes R$ does share common first and second order properties with the original rating matrix R. The new data size is 2 orders of magnitude larger in terms of number of rows and columns and 4 orders of magnitude larger in terms of number of non-
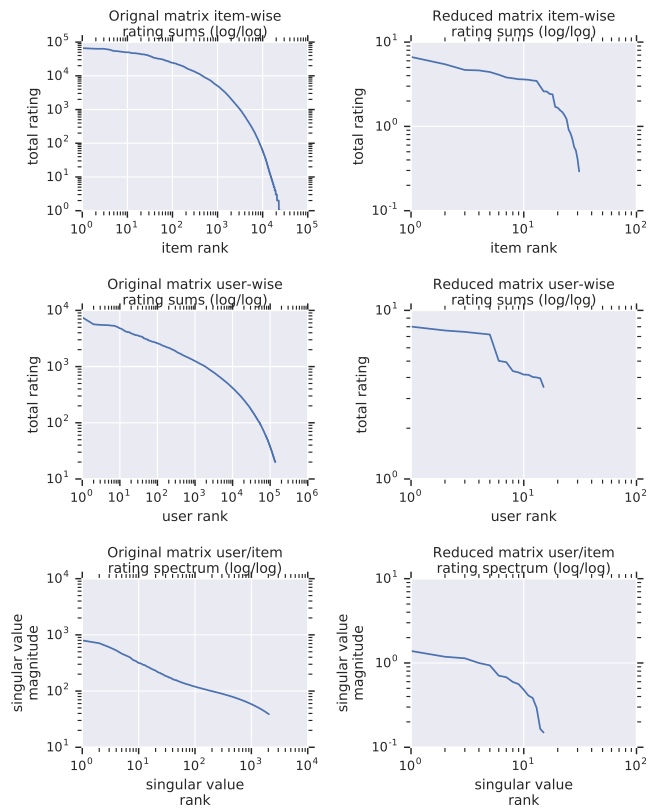

Fig. 3: Properties of the reduced dataset $\widehat{R} \in \mathbb{R}^{16,32}$ built according to steps 4, 5 and 6. We validate the construction method numerically by checking that the distribution of row-wise sums, column-wise sums and singular values are similar between R and $\widehat{R}$. Note here that as R is large we only compute its leading singular values. As we want to preserve statistical "power-laws", we focus on preservation of the relative distribution of values and not their magnitude in absolute.

zero terms. Notice here that because of the dropout, the density of the resulting data set is about 20% that of the original data set.

In Figure 4, one can confirm that the spectral properties of the expanded data set as well as the user engagement (row-wise sums) and item popularity (column-wise sums) are similar to those of the original data set. Such observations demonstrate that the theoretical insights from Proposition 1 and Theorem 1 are indeed informative of the high order statistics of the synthetic data set we generate. Our ex-post empirical study indicates that the resulting data set is representative — in its fat-tailed data distribution and quasi "power-law" singular value spectrum — of problems encountered in ML for collaborative filtering. Furthermore, the expanded data set reproduces some unique properties of the original data, in particular the accelerating decay of values in ranked row-wise and column-wise sums as well as in the singular values spectrum.
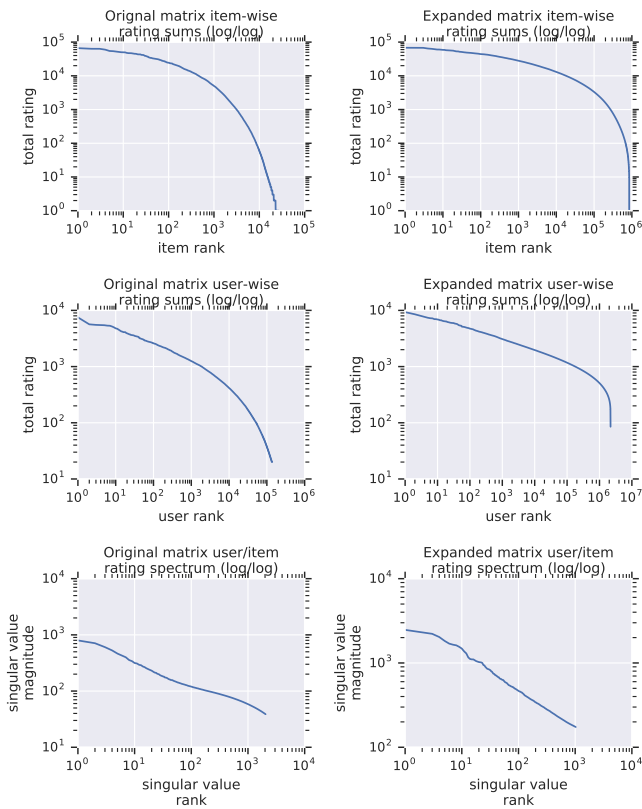
Fig. 4: High order statistical properties of the expanded dataset $\widehat{R} \otimes R$. We validate the construction method numerically by checking that the distributions of row-wise sums, column-wise sums and singular values are similar between R and $\widehat{R} \otimes R$. Here we leverage the tractability of Kronecker products as they impact column-wise and row-wise sum distributions as well as singular value spectra. In all plots we can observe the preservation of the linear log-log correspondence for the higher values in the distributions of interest (row-wise sums, column-wise sums and singular values) as well as the accelerated decay of the smaller values in those distributions.

VI. CONCLUSION

In conclusion, this paper presents an attempt at synthesizing a realistic large-scale recommendation data sets without having to make compromises in terms of user privacy. We use a small size publicly available data set, MovieLens 20m, and expand it to orders of magnitude more users, items and observed ratings. Our expansion model is rooted into the hierarchical structure of user/item interactions which naturally suggests a fractal extrapolation model.

We leverage randomized Kronecker products as self-similar operators on user/item rating matrices that preserve key properties of row-wise and column-wise sums as well as singular value spectra. We modify the original Kronecker Graph generation method to enable a randomized expansion of the original data by orders of magnitude that yields a synthetic data set matching industrial recommendation data sets in scale. Our

numerical experiments demonstrate the data set we create has key first and second order properties similar to those of the original MovieLens 20m binarized rating matrix.

Our next steps consist in making large synthetic data sets publicly available although any researcher can readily use the techniques we presented to scale up any user/item interaction matrix. Another possible direction is to adapt the present method to recommendation data sets featuring metadata (e.g. timestamps, topics, device information). The use of metadata is indeed critical to solve the "cold-start" problem of users and items having no interaction history with the platform. In this work, we did not consider here the temporal structure of the MovieLens data set. We leave the study of sequential user behavior — often found to be Long Range Dependent [5], [12], [41], [48] — and the extension of synthetic data generation to sequential recommendations [3], [49], [54] for further work. We also plan to benchmark the performance of well established baselines on the new large scale realistic synthetic data we produce.

REFERENCES

[1] ABDOLLAHPOURI, H., BURKE, R., AND MOBASHER, B. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (2017), ACM, pp. 42–46.

[2] ACHLIOPTAS, D. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences 66*, 4 (2003), 671–687.

[3] BELLETTI, F., BEUTEL, A., JAIN, S., AND CHI, E. Factorized recurrent neural architectures for longer range dependence. In *International Conference on Artificial Intelligence and Statistics* (2018), pp. 1522–1530.

[4] BELLETTI, F., LAKSHMANAN, K., KRICHENE, W., CHEN, Y.-F., AND ANDERSON, J. Scalable realistic recommendation datasets through fractal expansions. *arXiv preprint arXiv:1901.08910* (2019).

[5] BELLETTI, F., SPARKS, E., BAYEN, A., AND GONZALEZ, J. Random projection design for scalable implicit smoothing of randomly observed stochastic processes. In *Artificial Intelligence and Statistics* (2017), pp. 700–708.

[6] BENNETT, J., LANNING, S., ET AL. The netflix prize. In *Proceedings of KDD cup and workshop* (2007), vol. 2007, New York, NY, USA, p. 35.

[7] BHARGAVA, A., GANTI, R., AND NOWAK, R. Active positive semidefinite matrix completion: Algorithms, theory and applications. In *Artificial Intelligence and Statistics* (2017), pp. 1349–1357.

[8] BLEI, D. M., NG, A. Y., AND JORDAN, M. I. Latent dirichlet allocation. *Journal of machine Learning research 3*, Jan (2003), 993–1022.

[9] CHANEY, A. J., BLEI, D. M., AND ELIASSI-RAD, T. A probabilistic model for using social networks in personalized item recommendation. In *Proceedings of the 9th ACM Conference on Recommender Systems* (2015), ACM, pp. 43–50.

[10] CHANEY, A. J., STEWART, B. M., AND ENGELHARDT, B. E. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. *arXiv preprint arXiv:1710.11214* (2017).

[11] COVINGTON, P., ADAMS, J., AND SARGIN, E. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (2016), ACM, pp. 191–198.

[12] CRANE, R., AND SORNETTE, D. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences 105*, 41 (2008), 15649–15653.

[13] CREMONESI, P., KOREN, Y., AND TURRIN, R. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems* (2010), ACM, pp. 39–46.

[14] FRADKIN, D., AND MADIGAN, D. Experiments with random projections for machine learning. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (2003), ACM, pp. 517–522.

[15] GOEL, S., BRODER, A., GABRILOVICH, E., AND PANG, B. Anatomy of the long tail: ordinary people with extraordinary tastes. In *Proceedings of the third ACM international conference on Web search and data mining* (2010), ACM, pp. 201–210.

[16] GOLUB, G. H., AND VAN LOAN, C. F. *Matrix computations*, vol. 3. JHU Press, 2012.

[17] GRAVE, E., JOULIN, A., CISSÉ, M., JÉGOU, H., ET AL. Efficient softmax approximation for gpus. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 1302–1310.

[18] HARIRI, N., MOBASHER, B., AND BURKE, R. Context-aware music recommendation based on latenttopic sequential patterns. In *Proceedings of the sixth ACM conference on Recommender systems* (2012), ACM, pp. 131–138.

[19] HARPER, F. M., AND KONSTAN, J. A. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis) 5*, 4 (2016), 19.

[20] HE, X., LIAO, L., ZHANG, H., NIE, L., HU, X., AND CHUA, T.-S. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web* (2017), International World Wide Web Conferences Steering Committee, pp. 173–182.

[21] HORN, R. A., HORN, R. A., AND JOHNSON, C. R. *Matrix analysis*. Cambridge university press, 1990.

[22] JAWANPURIA, P., AND MISHRA, B. A unified framework for structured low-rank matrix learning. In *International Conference on Machine Learning* (2018), pp. 2259–2268.

[23] JOLLIFFE, I. Principal component analysis. In *International encyclopedia of statistical science*. Springer, 2011, pp. 1094–1096.

[24] KAMM, J., AND NAGY, J. G. Optimal kronecker product approximation of block toeplitz matrices. *SIAM Journal on Matrix Analysis and Applications 22*, 1 (2000), 155–172.

[25] KOREN, Y., BELL, R., AND VOLINSKY, C. Matrix factorization techniques for recommender systems. *Computer*, 8 (2009), 30–37.

[26] KRICHENE, W., MAYORAZ, N., RENDLE, S., ZHANG, L., YI, X., HONG, L., CHI, E., AND ANDERSON, J. Efficient training on very large corpora via gramian estimation. *arXiv preprint arXiv:1807.07187* (2018).

[27] LECUN, Y., BENGIO, Y., AND HINTON, G. Deep learning. *nature 521*, 7553 (2015), 436.

[28] LESKOVEC, J., CHAKRABARTI, D., KLEINBERG, J., AND FALOUTSOS, C. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *European Conference on Principles of Data Mining and Knowledge Discovery* (2005), Springer, pp. 133–145.

[29] LESKOVEC, J., CHAKRABARTI, D., KLEINBERG, J., FALOUTSOS, C., AND GHAHRAMANI, Z. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research 11*, Feb (2010), 985–1042.

[30] LEVY, M., AND BOSTEELS, K. Music recommendation and the long tail. In *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys, 2010, Barcelona, Spain* (2010), Citeseer.

[31] LI, P., HASTIE, T. J., AND CHURCH, K. W. Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (2006), ACM, pp. 287–296.

[32] LIANG, D., ALTOSAAR, J., CHARLIN, L., AND BLEI, D. M. Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Proceedings of the 10th ACM conference on recommender systems* (2016), ACM, pp. 59–66.

[33] LINDEN, G., SMITH, B., AND YORK, J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 1 (2003), 76–80.

[34] LU, J., LIANG, G., SUN, J., AND BI, J. A sparse interactive model for matrix completion with side information. In *Advances in neural information processing systems* (2016), pp. 4071–4079.

[35] MAHDIAN, M., AND XU, Y. Stochastic kronecker graphs. In *International Workshop on Algorithms and Models for the Web-Graph* (2007), Springer, pp. 179–186.

[36] MANDELBROT, B. B. *The fractal geometry of nature*, vol. 1. WH freeman New York, 1982.

[37] NARAYANAN, A., AND SHMATIKOV, V. How to break anonymity of the netflix prize dataset. *arXiv preprint cs/0610105* (2006).

[38] NIMISHAKAVI, M., JAWANPURIA, P. K., AND MISHRA, B. A dual framework for low-rank tensor completion. In *Advances in Neural Information Processing Systems* (2018), pp. 5489–5500.

[39] OESTREICHER-SINGER, G., AND SUNDARARAJAN, A. Recommendation networks and the long tail of electronic commerce. *Mis quarterly* (2012), 65–83.

[40] PARK, Y.-J., AND TUZHILIN, A. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM conference on Recommender systems* (2008), ACM, pp. 11–18.

[41] PIPIRAS, V., AND TAQQU, M. S. *Long-range dependence and self-similarity*, vol. 45. Cambridge university press, 2017.

[42] RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL, J. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work* (1994), ACM, pp. 175–186.

[43] RICCI, F., ROKACH, L., AND SHAPIRA, B. Recommender systems: introduction and challenges. In *Recommender systems handbook*. Springer, 2015, pp. 1–34.

[44] RUDOLPH, M., RUIZ, F., MANDT, S., AND BLEI, D. Exponential family embeddings. In *Advances in Neural Information Processing Systems* (2016), pp. 478–486.

[45] SARWAR, B., KARYPIS, G., KONSTAN, J., AND RIEDL, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (2001), ACM, pp. 285–295.

[46] SCHMIT, S., AND RIQUELME, C. Human interaction with recommendation systems. *arXiv preprint arXiv:1703.00535* (2017).

[47] SHANI, G., HECKERMAN, D., AND BRAFMAN, R. I. An mdp-based recommender system. *Journal of Machine Learning Research 6*, Sep (2005), 1265–1295.

[48] TANG, J., BELLETTI, F., JAIN, S., CHEN, M., BEUTEL, A., XU, C., AND CHI, E. H. Towards neural mixture recommender for long range dependent user sequences. *arXiv preprint arXiv:1902.08588* (2019).

[49] TANG, J., AND WANG, K. Personalized top-n sequential recommendation via convolutional sequence embedding. In *International Conference on Web Search and Data Mining* (2018), IEEE, pp. 565–573.

[50] TU, K., CUI, P., WANG, X., WANG, F., AND ZHU, W. Structural deep embedding for hyper-networks. In *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).

[51] VAN DER WALT, S., SCHÖNBERGER, J. L., NUNEZ-IGLESIAS, J., BOULOGNE, F., WARNER, J. D., YAGER, N., GOUILLART, E., AND YU, T. scikit-image: image processing in python. *PeerJ 2* (2014), e453.

[52] WANG, H., WANG, N., AND YEUNG, D.-Y. Collaborative deep learning for recommender systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), ACM, pp. 1235–1244.

[53] YIN, H., CUI, B., LI, J., YAO, J., AND CHEN, C. Challenging the long tail recommendation. *Proceedings of the VLDB Endowment 5*, 9 (2012), 896–907.

[54] YU, F., LIU, Q., WU, S., WANG, L., AND TAN, T. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), ACM, pp. 729–732.

[55] ZHANG, J.-D., CHOW, C.-Y., AND XU, J. Enabling kernel-based attribute-aware matrix factorization for rating prediction. *IEEE Transactions on Knowledge and Data Engineering 29*, 4 (2017), 798–812.

[56] ZHAO, Q., CHEN, J., CHEN, M., JAIN, S., BEUTEL, A., BELLETTI, F., AND CHI, E. H. Categorical-attributes-based item classification for recommender systems. In *Proceedings of the 12th ACM Conference on Recommender Systems* (2018), ACM, pp. 320–328.

[57] ZHENG, Y., TANG, B., DING, W., AND ZHOU, H. A neural autoregressive approach to collaborative filtering. *arXiv preprint arXiv:1605.09477* (2016).

[58] ZHOU, B., HUI, S. C., AND CHANG, K. An intelligent recommender system using sequential web access patterns. In *IEEE conference on cybernetics and intelligent systems* (2004), vol. 1, IEEE Singapore, pp. 393–398.

[59] ZHOU, G., ZHU, X., SONG, C., FAN, Y., ZHU, H., MA, X., YAN, Y., JIN, J., LI, H., AND GAI, K. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2018), ACM, pp. 1059–1068.