EDA4SUM: Guided Exploration of Data Summaries

Aurélien Personnaz
aurelien.personnaz@cnrs.fr
CNRS, Univ. Grenoble Alpes

Brit Youngmann
brity@mit.edu
MIT CSAIL

Sihem Amer-Yahia
sihem.amer-yahia@cnrs.fr
CNRS, Univ. Grenoble Alpes

ABSTRACT

We demonstrate EDA4Sum, a framework dedicated to generating guided multi-step data summarization pipelines for very large datasets. Data summarization is the process of producing interpretable and representative subsets of an input dataset. It is usually performed following a one-shot process with the purpose of finding the best summary. EDA4Sum leverages Exploratory Data Analysis (EDA) to produce connected summaries in multiple steps, with the goal of maximizing their cumulative utility. A useful summary contains k *individually uniform* sets that are *collectively diverse* to be representative of the input data. EDA4Sum accommodates datasets with different characteristics by providing the ability to tune the weights of uniformity, diversity and novelty when generating multi-step summaries. We demonstrate the superiority of multi-step EDA summarization over single-step summarization for summarizing very large data, and the need to provide guidance to domain experts, by interacting with the VLDB'22 participants who will act as data analysts. The application is avilable at https://bit.ly/eda4sum_application.

1 INTRODUCTION

The goal of data summarization is to produce small and representative subsets [10] of an input dataset. Produced subsets must be interpretable and representative of the input. Intuitively, a useful summary contains k *individually uniform* sets that are *collectively diverse* [13]. Uniformity addresses interpretability, and diversity addresses representativity by seeking to cover data variety. This is particularly important for very large datasets. However, a single one-shot summary of a large dataset will not be representative. To this end, we present EDA4Sum a dedicated system that generates connected summaries in a multi-step fashion with the goal of covering diversity in very large datasets.

Example. *Consider the Sloan Digital Sky Survey (SDSS), a database of galaxies belonging to* 169 *classes [11]* [1]*. In SDSS, each galaxy has* 7 *attributes describing magnitude in each color filter (the attributes u, g, r, i, and z), size (the attribute* petroRad_r*), and how far a galaxy is from the Earth (the attribute* redshift*). A summary is a set of k* diverse itemsets *each of which is* uniform*, i.e., contains items that are similar to each other. Figure 1 shows examples of uniform and non-uniform galaxy itemsets. One can see that uniform itemsets are easier to interpret by humans. To build a summary for SDSS, a one-shot approach that leverages diversity algorithms and finds the k most uniform and diverse sets appears as a natural solution. Figure 2 shows a snapshot of a summary returned by SWAP, a common diversity algorithm [13] that finds the most diverse itemsets subject to*

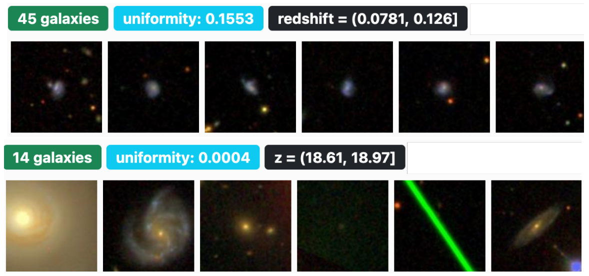


Figure 1: Examples of uniform and non-uniform itemsets.

a uniformity threshold.[2] *While the shown summary satisfies uniformity and diversity, it does not cover the variety of galaxy types in the database. Hence. multi-step summaries are needed for large datasets.*

The first challenge of summarizing large datasets is how to generate high-quality connected summarization pipelines that are interpretable and representative. Connected summaries preserve the train of thought of the user [8] (challenge **C1**). This motivates Exploratory Data Analysis (EDA) for data summarization. However, as in modern EDA [2, 7], the system should guide users by recommending the next operation to discover more uniform and diverse itemsets (challenge **C2**). Last, the system should have interactive running times (challenge **C3**).

To address **C1**, we formalize the Eda4Sum Problem that seeks a summarization pipeline whose *cumulated utility* (which combines uniformity and diversity) is maximized. A summarization pipeline has a fixed length and reduces to a one-shot summarization when the length is equal to 1. A summarization pipeline in EDA4Sum starts by running the SWAP algorithm that greedily finds the most diverse itemsets subject to a threshold on uniformity. When producing summaries sequentially, we must ensure that the generated summaries at each step are both new and related to previous summaries. Therefore, our utility measure also considers the *novelty of the current summary w.r.t. previous steps.* Each summary is obtained by applying an EDA operation on the last seen summary, thereby *connecting summaries.* This exploits semantic relationships between data regions to preserve the stream of consciousness of the user [8].

To address **C2**, EDA4Sum offers three summarization modes: *Manual*, *Partial Guidance* and *Full Guidance*. In the first mode, the system displays a summary at each step, and the user chooses an itemset in the summary and an operation to apply to that itemset; With Partial and Full Guidance, the system displays a t-size summarization pipeline with the difference that in Partial Guidance, users may intervene and modify a step by choosing a different input itemset or a different operator.

To address **C3**, EDA4Sum relies on one of our two algorithms. At each step, each algorithm picks one of the itemsets returned by the previous step and chooses the next operator, resulting in a new summary. Our first algorithm, Top1Sum, applies a local optimization to

[1]https://www.sdss.org/

[2]The actual summary has 10 itemsets that do not cover the 169 galaxy types.

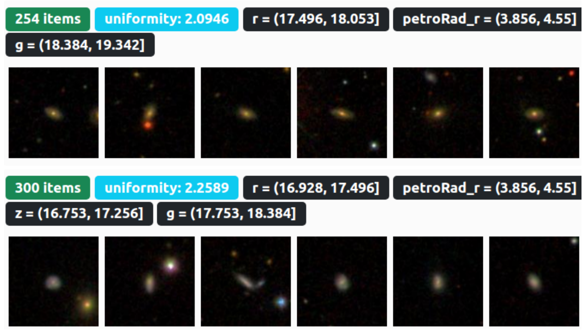


Figure 2: A summary obtained with a diversity algorithm.

find the operation that produces the best utility summary.Our second algorithm, RLSum, is faster as it relies on a pre-trained policy, i.e., summarization pipeline. RLSum leverages Deep Reinforcement Learning to simulate an agent that learns an end-to-end summarization policy exploiting semantic relationships between different data regions. The logic of RLSum is based on the system presented in [5] that guides users in finding items of interest in large datasets. However, while the offline phase of the two systems is identical, our novel contribution is the online phase of EDA4Sum.

While the execution time of Top1Sum depends on data properties and on supported EDA operators, RLSum is agnostic to these parameters and can perform interactive summarization. Our experiments show that while Top1Sum produces higher utility summaries, RLSum is at least one order of magnitude faster than Top1Sum and performs better than Top1Sum in finding ground-truth itemsets.

Related Work. We provide the theoretical foundations and algorithms underlying this framework in [12]. A variety of approaches have been proposed for summarizing data [3]. Prominent examples include those that identify extreme aggregates [10], those that summarize all aggregates [3], and those that produce k diverse clusters [6]. Unlike our work, all these methods consider data summarization as a one-shot task. Approaches that summarize all data typically trade-off summary size against information loss [3]. In cases where the data size is massive, finding the most uniform and diverse parts is more useful. EDA is known as a difficult task, requiring profound analytical skills, experience [1]. Recent work suggested fully automating this process using Reinforcement Learning [2, 5]. As opposed to this line of work whose goal is to extract general insights, we aim to summarize massive datasets by finding highly uniform and diverse itemsets.

2 TECHNICAL BACKGROUND

2.1 Data Model and Problem Formulation

We consider a set of items D described with a set of ordinal attributes A. Numerical attribute values are binned into a fixed number of bins. We use the notion of *itemset* defined as a set of items that share the same values for a set of attributes. Those attributes define the *itemset description* that has the benefit of conveying the content of the itemset at a glance. To illustrate, Figures 1 and 2 illustrate galaxy itemsets. $\mathcal{D}$ denotes the set of all itemsets created from D. We represent each itemset s with a vector v_s of aggregated values (e.g., mean or mode) of items in s *for each attribute in* A.

A summary $S \subseteq \mathcal{D}$ is a set of (at most) k itemsets. Intuitively, a useful summary contains pairwise different itemsets (diversity), each of which consisting of similar items (uniformity). Since we generate multi-step summaries, an important question is to what extent itemsets in the current step differ to those in previous steps (novelty). To that end, we define the *utility of a summary* as a combination of its uniformity, diversity and novelty:

Uniformity. The uniformity of a summary measures how similar items are to each other in each of its itemsets. Itemset uniformity is computed as the inverse of the mean of its attribute variances. The uniformity of a summary is the smallest among the uniformity scores of its itemsets. Other aggregations can be supported.

Diversity. The diversity of a summary is defined as the smallest vector distance among its itemsets. We use the Manhattan distance metric. Other metrics could be used. The intuition behind choosing minimum is that *a summary is diverse if its two most similar itemsets are different.* Here again, other aggregation functions could be used.

Novelty. The novelty of a summary is simply the proportion of how many new itemsets the user is currently seeing.

The utility of a summary is defined as a linear combination of uniformity, diversity and novelty, where $\alpha, \beta, \gamma \in [0, 1]$ are the coefficients of the uniformity, diversity and novelty, resp.

A *summarization pipeline* is a sequence of summaries, where we move to the next summary using some exploration operator. Generally, an operator takes an itemset s and a number k, and returns a summary formed by (at most) k itemsets that are related to items in s. In EDA4Sum, in addition to the traditional drill-down (by-facet) and roll-up (by-superset) operations, we support the by-distrib and by-neighborsoperators, that were introduced in [5]. The cumulated utility of a pipeline is the sum of the utilities of the summaries generated at each step.

The Eda4Sum Problem: Given a bound t on the number of summaries to be displayed, EDA4Sum solves the Eda4Sum Problem that seeks to find for a given dataset D, the highest utility summarization pipeline of length t.

EDA4Sum accommodates datasets with different characteristics by providing the ability to tune the weights of uniformity, diversity and novelty when generating multi-step summaries.

2.2 Algorithms

We implemented two algorithms to solve the Eda4Sum Problem. The initial summary is always obtained using the SWAP algorithm [13]. In the following steps, the user sees a summary S' that is obtained by applying one of our two algorithms, Top1Sum or RLSum to an itemset s chosen from a summary S that was shown in the previous step. This provides an easy to follow sequence of summaries that preserves the stream of consciousness of the user [8] by connecting the summaries the user sees at each step.

Top1Sum: The Top1Sum algorithm applies local optimization to find the operation that produces the highest utility summary at each step of the summarization pipeline. Intuitively, at each step, Top1Sum examines every possible next step, i.e., every (itemset, *explore*(), attributes) combination, and executes the step that yields the summary with the highest utility. Top1Sum has no theoretical guarantees for the Eda4Sum Problem. While Top1Sum works

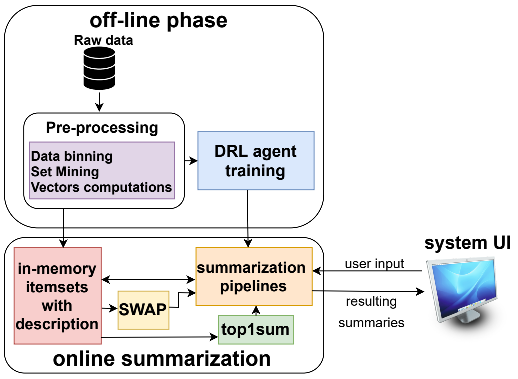


Figure 3: Overall Architecture of EDA4Sum. The off-line phase is based on the system presented in [5]. Our novel part is the online summarization generation component.

well in practice, and is able to generate high utility summarization pipelines, its running time may be high.

RLSum: To address the high running time of Top1Sum, we adopt Deep Reinforcement Learning to train a high-utility pipeline [5].

We model the Eda4Sum Problem as a Markov Decision Process and define a *summarization policy* as a mapping from a summarization state to an action, and look for the policy maximizing expected reward. The reward of an action on a state is the utility of its output summary (with a discount factor affected by the policy length).

RLSum leverages the DRL algorithm presented in [5] for training. This algorithm, guidance consists of an iterative process driven by data familiarity and curiosity, whereas in RLSum the process is driven by uniformity, diversity, and novelty.

We adapt model-free RL [5] and use A3C [4], a state-of-the-art critic-based DRL framework. The appeal of A3C comes from its parallelized and asynchronous architecture: multiple actor-learners are dispatched to separate instantiations of the environment; they interact with the environment and collect experience and asynchronously push their gradient updates to a central target network. We instantiate the environment interface for each worker with the utility weights defined for the training. For every operator execution step, the reward is computed. The value network learns a baseline state value to which the current reward estimate is compared to obtain the “advantage”. The policy network adjusts the log probabilities of the actions based on the advantage via the RL algorithm. We then train the policy with the newly computed advantage values and train the value function with the obtained reward. This process is completed in parallel by each worker.

3 SYSTEM AND DEMONSTRATION

3.1 EDA4SUM

The architecture of EDA4Sum is given in Figure 3. The offline phase is based on the system presented in [5]. Equi-depth binning is applied to each attribute and we use [9] to generate itemsets. Different RL models with different utility weights are trained. Our novel contribution lies in the online phase, where we generate summarization pipelines following one of three modes: *Manual*, *Partial Guidance* and *Full Guidance*. Pipeline execution starts with the SWAP algorithm [13] that finds the k most uniform and diverse itemsets. The next steps are picked by either Top1Sum or RLSum.

System UI. The UI of EDA4Sum is depicted in Figure 4 (with the SPOTIFY dataset). The current 7-step pipeline is shown in the A zone and its results in the B zone. The user investigates a dataset by specifying the summarization mode, underlying algorithm, and possibly customizing the weights of uniformity, diversity and novelty (C zone). In case RLSum is selected, the weights are used to identify the pre-trained model which will be used for guidance. In *Manual* and *Partial* modes, the user can override the next operator and its parameters (D zone). The user may want to store the current pipeline or upload a previously stored one to execute it (E zone).

RLSum implementation We use a Tensorflow-based implementation of A3C.[3] The agents were trained on two servers with Intel Xeon processors, one with 370GB and the other with 126GB of RAM. Training took 100 hours for 4000 episodes on SDSS and 3000 on SPOTIFY, with 50 steps per episode. Each agent used 6 workers in parallel; the update interval was set to 20 steps, and we concatenated three successive states for the LSTM layers.

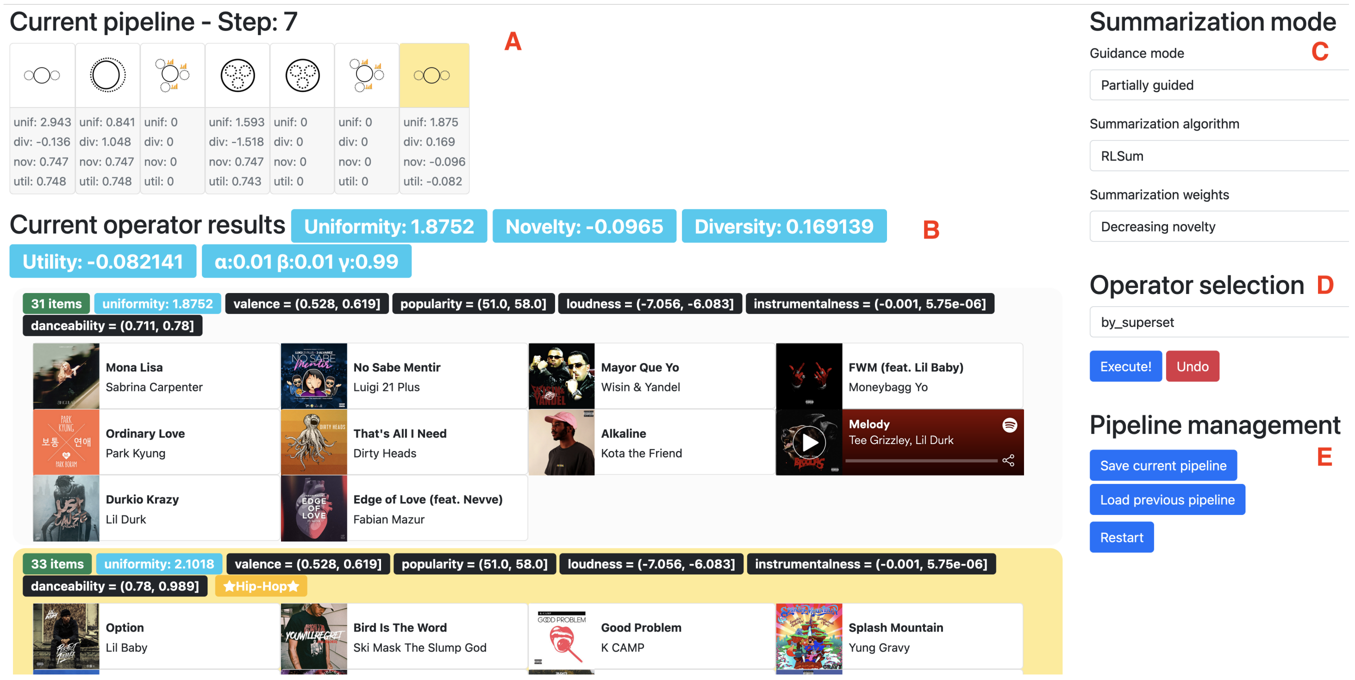
3.2 Demonstration Overview

We demonstrate EDA4Sum over two large datasets, SDSS (includes 2.6M galaxies with 7 attributes) and SPOTIFY (includes $232K$ music tracks with 11 attributes). For both datasets, we define a set of "ground-truth" uniform itemsets to be discovered. For SDSS, they correspond to 169 well-known galaxy types extracted from [11]. For SPOTIFY, they correspond to the partition of all music records by the attribute genre (with 27 values). Attendees will be able to compare Top1Sum and RLSum using different weighting schemes (fixed or evolving weights) for the parameters α, β and γ (affecting the uniformity, diversity, and novelty resp.). For example, increasing novelty weight (with decreasing uniformity and diversity), or fixed balanced weights (for uniformity, diversity, and novelty). The audience will engage with EDA4Sum via the following scenarios:

Summarization Guidance: We simulate a common real-life scenario where a data analyst tries to identify highly uniform and diverse itemsets in a large dataset. To examine the benefit of guidance during summarization, we will randomly assign each participant to one of the *Partial Guidance* or *Manual* modes. We will then ask them to find the ground-truth itemsets and highlight them (see the second ground-truth itemset in Figure 4 marked with a yellow label indicating its genre.) The participants would then compare their obtained results with the ones achieved by *Full Guidance*. This scenario illustrates that data summarization is a difficult task, requiring profound analytical skills, experience, and domain knowledge, and highlights the benefit of EDA.

Utility and Relevance: This scenario leverages *Full Guidance*. The audience can choose an algorithm and a weighting scheme and examine the proposed summarization pipeline in terms of cumulated utility and the number of discovered ground-truth itemsets. The participants would then be able to use the system with different variants to compare the results.

[3]https://github.com/marload/DeepRL-TensorFlow2/


Current pipeline - Step: 7

A

unif: 2.943	unif: 0.841	unif: 0	unif: 1.593	unif: 0	unif: 0	unif: 1.875
div: -0.136	div: 1.048	div: 0	div: -1.518	div: 0	div: 0	div: 0.169
nov: 0.747	nov: 0.747	nov: 0	nov: 0.747	nov: 0	nov: 0	nov: -0.096
util: 0.748	util: 0.748	util: 0	util: 0.743	util: 0	util: 0	util: -0.082

Current operator results Uniformity: 1.8752 Novelty: -0.0965 Diversity: 0.169139 B

Utility: -0.082141 α:0.01 β:0.01 γ:0.99

31 items uniformity: 1.8752 valence = (0.528, 0.619] popularity = (51.0, 58.0] loudness = (-7.056, -6.083] instrumentalness = (-0.001, 5.75e-06] danceability = (0.711, 0.78]

Mona Lisa Sabrina Carpenter	No Sabe Mentir Luigi 21 Plus	Mayor Que Yo Wisin & Yandel	FWM (feat. Lil Baby) Moneybagg Yo
Ordinary Love Park Kyung	That's All I Need Dirty Heads	Alkaline Kota the Friend	Melody Tee Grizzley, Lil Durk
Durkio Krazy Lil Durk	Edge of Love (feat. Nevve) Fabian Mazur		

33 items uniformity: 2.1018 valence = (0.528, 0.619] popularity = (51.0, 58.0] loudness = (-7.056, -6.083] instrumentalness = (-0.001, 5.75e-06] danceability = (0.78, 0.989] ⭐Hip-Hop⭐

Option Lil Baby	Bird Is The Word Ski Mask The Slump God	Good Problem K CAMP	Splash Mountain Yung Gravy

Summarization mode

Guidance mode C

Partially guided

Summarization algorithm

RLSum

Summarization weights

Decreasing novelty

Operator selection D

by_superset

Execute! Undo

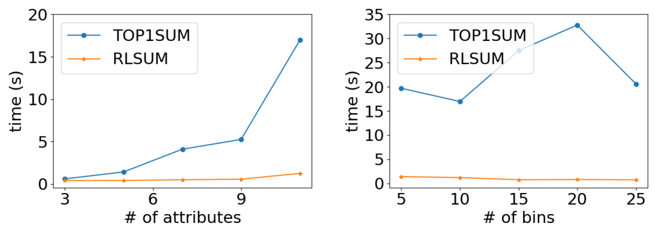
Pipeline management

Save current pipeline E

Load previous pipeline

Restart

Figure 4: UI of EDA4Sum with SPOTIFY.



(a) # attributes **(b) # bins**

Figure 5: Average execution times (SDSS).

Looking Under the Hood: The audience will be allowed to examine the efficiency of our algorithms. We will show that while Top1Sum returns higher utility summaries, RLSum is at least one order of magnitude faster than Top1Sum. As shown in our preliminary results in Figure 5, the difference in running times between the algorithms increases as the number of attributes increases. Observe that the performance of both algorithms improves with a higher number of bins since increasing the number of bins reduces the number of mined itemsets, resulting in reduced execution times. However, in all cases RLSum is at least one order of magnitude faster than Top1Sum.

ACKNOWLEDGMENTS

This work is funded by the European Union's Horizon 2020 research and innovation program (project name: INODE) under grant agreement No 863410.

REFERENCES

[1] Sihem Amer-Yahia, Tova Milo, and Brit Youngmann. 2021. Exploring Ratings in Subjective Databases. In *SIGMOD*.

[2] Ori Bar El, Tova Milo, and Amit Somech. 2020. Automatically generating data exploration sessions using deep reinforcement learning. In *SIGMOD*. 1527–1537.

[3] Alexandra Kim, Laks VS Lakshmanan, and Divesh Srivastava. 2020. Summarizing Hierarchical Multidimensional Data. In *ICDE*. IEEE.

[4] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. In *ICML*.

[5] Aurélien Personnaz, Sihem Amer-Yahia, Laure Berti-Équille, Maximilian Fabricius, and Srividya Subramanian. 2021. DORA THE EXPLORER: Exploring Very Large Data With Interactive Deep Reinforcement Learning. In *CIKM*. ACM.

[6] Senjuti Basu Roy, Sihem Amer-Yahia, Ashish Chawla, Gautam Das, and Cong Yu. 2010. Constructing and exploring composite items. In *SIGMOD*. ACM.

[7] Mariia Seleznova, Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, and Eric Simon. 2020. Guided Exploration of User Groups. *pVLDB Endow.* 13, 9 (2020), 1469–1482.

[8] Dafna Shahaf and Carlos Guestrin. 2011. Connecting the Dots between News Articles. In *IJCAI*. IJCAI/AAAI.

[9] Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. 2004. LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI*, Vol. 126.

[10] Yuhao Wen, Xiaodan Zhu, Sudeepa Roy, and Jun Yang. 2018. Interactive summarization and exploration of top aggregate query answers. In *PVLDB*. NIH.

[11] Kyle W. Willett, Chris J. Lintott, Steven P. Bamford, Karen L. Masters, Brooke D. Simmons, Kevin R. V. Casteels, Edward M. Edmondson, Lucy F. Fortson, Sugata Kaviraj, William C. Keel, and et al. 2013. Galaxy Zoo 2: detailed morphological classifications for 304 122 galaxies from the Sloan Digital Sky Survey. *Royal Astronomical Society* (2013).

[12] Brit Youngmann, Sihem Amer-Yahia, and Aurelien Personnaz. 2022. Guided Exploration of Data Summaries. In *pVLDB Endow.* NIH.

[13] Cong Yu, Laks Lakshmanan, and Sihem Amer-Yahia. 2009. It takes variety to make a world: diversification in recommender systems. In *EDBT*.