SubDEx: Exploring Ratings in Subjective Databases
(Authors’ Copy)
Sihem Amer-Yahia, Tova Milo, Brit Youngmann

To cite this version:
Sihem Amer-Yahia, Tova Milo, Brit Youngmann. SubDEx: Exploring Ratings in Subjective Databases (Authors’ Copy). 2021 IEEE 37th International Conference on Data Engineering (ICDE), Apr 2021, Chania, France. pp.2653-2656, 10.1109/ICDE51399.2021.00296. hal-03379575

HAL Id: hal-03379575
https://hal.archives-ouvertes.fr/hal-03379575
Submitted on 15 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
SubDEx: Exploring Ratings in Subjective Databases
(Authors’ Copy)

Sihem Amer-Yahia
CNRS, Univ. Grenoble Alpes
sihem.amer-yahia@univ-grenoble-alpes.fr

Tova Milo
Tel Aviv University
milo@post.tau.ac.il

Brit Youngmann
Tel Aviv University
brity@mail.tau.ac.il

Mary’s example illustrates two key needs that characterize SDE: the need to select (simultaneously) subsets and supersets of items and reviewers whose aggregated ratings demonstrate useful and diverse facets of reviewers’ opinions, and the need to explore different rating dimensions, e.g., food vs. service for restaurants. In this work we present SubDEx, a dedicated SDE framework. There are three key challenges in building such a system. First, the system should cater to the above-mentioned needs (challenge C1). Namely, it should display to users aggregated ratings that demonstrate useful and diverse facets of the data, while aggregating reviewers and items by different rating dimensions. Second, as in modern EDA tools [2], [3], the system should provide to users some guidance on the next operation to perform, to discover interesting trends in the data (challenge C2). Last, the system should enable interactive running times (challenge C3).

To address C1, SubDEx provides the ability to apply, at each step, a filtering or generalization operation on the items and reviewers of interest. It then displays, alongside the resulting rating records, a set of k rating maps [4] (see Figure 3(a)). Rating maps are histograms that provide a bird’s-eye view of ratings by some reviewers for some items. The rating maps displayed at each step are chosen to be useful and diverse. Our notion of utility generalizes previous interestingness measures [5], whereas our notion of diversity ensures that different facets of the data are revealed. To ensure that the selected rating maps depict different rating dimensions, we use weighted utility scores where the weights reflect the number of times a rating dimension has been previously shown.

To address C2, SubDEx offers two exploration modes: Recommendation-Powered, and Fully-Automated. In the first mode, the system presents the current k most useful and diverse rating maps at each step, and recommends o next-step operations based on the utility and diversity of the rating maps they generate (see Figure 3). The user can choose one recommendation or perform an operation of her own. This was the case for Mary. The second Fully-Automated mode relieves the user from choosing an operation, and generates a fixed-size exploration path, by applying the top-1 operation at each step.

To address C3, SubDEx applies pruning optimizations that estimate the weighted utility score for each rating map based on sampling techniques and prune low utility ones. To enable that, we adapted highly efficient sharing and pruning techniques [6] for identifying high-utility rating maps and reduce computational costs.

Due to space limitations, we provide here only a brief
Overview of our solution. Full details can be found in [7].

Demonstration Overview: We demonstrate the operation of SunDex over multiple real-world subjective datasets. Our demonstration illustrates real-life scenarios where a data analyst attempts to identify special data characteristics. The audience will play the role of data analysts, using one (or more) of SunDex exploration modes. Then, the audience will explore statistics describing the results of other participants, enabling to observe the effect of guidance in SDE. Last, the audience will be allowed to look “under the hood”, examining the efficiency of our solution.

Related Work: Subjective data analysis is an emerging research field [1]. Such data is widely used in web applications, online rating systems, and social sciences [5]. SDE can be used for large-scale population studies whose purpose is to extract trends and insights on the users/items, or for extracting recommendations [4]. To the best of our knowledge, SunDex is the first system dedicated to SDE.

Exploratory Data Analysis (EDA) is an essential task for data scientists, with the goal of extracting insights from datasets. Guiding users in performing EDA is a well-studied task [2], [3]. While general-purpose EDA tools could also be used for SDE, an SDE tool must cater to additional needs that require tailored solutions. Auto-generation of interesting views for a dataset has been studied extensively [8]. A common approach that we follow, is to use heuristic measures of interestingness [5], searching the space of all views, and returning the most interesting ones [6]. Multiple techniques to enable scalable visualization have been proposed [9]. Here we leverage pruning optimizations to identify low-utility rating maps, based on an adaptation of techniques presented in [6].

II. The SunDex Framework

We begin by providing a short technical background, then present SunDex’s architecture and workflow.

A. Technical Background

Data Model: We consider a special type of database, called a subjective database [1], which includes both objective and subjective attributes. We model our database as a triple $(I, U, R)$, representing the sets of items, reviewers, and rating records, resp. Items and reviewers are associated with objective attributes, such as a restaurant address, and a reviewer occupation. An attribute value may be an atomic value or of complex type. For example, the value for the attribute cuisine of a restaurant may be multi-valued. The rating records have subjective attributes, reflecting the rating scores assigned by reviewers to items. For instance, a reviewer may rate a restaurant on several dimensions: food, service, and ambience. Each rating record, $r \in R$ is itself a tuple $⟨i, u, s_1, \ldots, s_j⟩$, where $i \in I, u \in U$, and $s_j$ is the rating score that reviewer $u$ assigned to item $i$ for the $j$-th rating dimension. The rating scores are application-dependent and do not affect our model.

Reviewer, Item and Rating Groups: A reviewer group $g_U$ (resp., item group $g_I$) is a set of reviewers (resp., items) that share the same values for a set of objective attributes which define its description. For example, consider the groups depicted in Figure 3(a). Here $g_U = \{\text{age} \_ \text{group}, \text{young} \_ \text{adult}\}$ contains all young adult reviewers, and $g_I = \{\text{state} \_ \text{NY}, \text{city} \_ \text{NYC}\}$ contains all restaurants in New York city. Given reviewer and item groups $g_U$ and $g_I$, a rating group $g_R$ for $g_U$ and $g_I$ is defined as the group of all rating records $r = (u, i, s_1, \ldots, s_j) \ s.t. \ u \in g_U, \ i \in g_I$. A rating group is captured by a set of attribute value pairs shared among reviewers and items, and can thus be interpreted as a predicate on the rating table.

Rating Maps: To provide a bird’s eye view of the ratings in a group $g_R$, we use rating maps [4] - histograms that aggregate ratings in $g_R$ using some item/reviewer attributes. Previous work has shown that such histograms are an adequate means of understanding rated datasets [10]. A rating map $rm$ of a rating group $g_R$ for a rating dimension $r_i$ partitions the records in $g_R$ into disjoint subgroups, and assigns to each subgroup $g_{ij} \subseteq g_R$ an aggregated score. We assume that a rating map $rm$ partitions $g_R$ using solely one reviewer or item attribute. Thus, a rating map can be seen as the result of a GroupBy operation over $g_R$, followed by an aggregation function (average in this work) to assign a single rating score to each subgroup. For example, consider the upper rating map in Figure 1 step I, obtained by partitioning $g_R$ on age group. It associates to each subgroup its average overall score.

To identify rating maps presenting useful and interesting trends in the data, we next define the utility score of a rating map. We then introduce the refined notion of dimension-weighted (DW) utility score of a rating map, which will help SunDex in presenting different rating dimensions.

To define the utility of a rating map, we generalize common interestingness measures for data exploration [5].

Conciseness. The conciseness score of a rating map $rm$ is a function of the number of subgroups in $rm$. It favors rating maps containing a small number of subgroups that summarizes a large number of records in $g_R$. Here we use the compaction gain measure [11]. Agreement. The agreement score of a rating map conveys that each subgroup in $g_R$ contains reviewers
who agree among themselves. To measure agreement within a subgroup, here we use Standard Deviation, which measures the amount of dispersion of a set of values w.r.t. the mean. The final agreement score is the average score of all subgroups.

**Peculiarity** This measure ranks a rating map w.r.t. itself, i.e., examining the peculiarity of each subgroup within it. The second measures the peculiarity of a rating map w.r.t. previously displayed rating maps (global). It captures the ability of a rating map to show a new facet of the data. To measure the peculiarity, here we use the total variation distance, a common deviation measure.

Other measures can be used for each of the above utility criterion, without impacting our solution. The utility score of a rating map is defined as the maximal score, among the four scores mentioned above.

As mentioned, we refine the utility scores to ensure rating maps of different rating dimensions are presented. Intuitively, the Dimension Weighted (DW) utility of a rating map aggregated by dimension $r_i$ is a combination of its utility and a weight reflecting how important it is to promote $r_i$. Rating dimensions that have been rarely selected would be promoted at the expense of those that have been frequently selected.

**B. System Architecture And Workflow**

An SDE process starts when a user loads a dataset to an analysis UI. She then executes a series of filtering/generalization operations. In each exploration step, the user examines a rating group $g_R$, defined by a reviewer group $g_U$ and an item group $g_I$, and a set of rating maps relevant for $g_R$. To move to the next step, the user performs an operation on $g_U$, on $g_I$, or on both, where each operation can be seen as a selection query over $g_U$ and $g_I$.

The architecture of SunDEx is depicted in Figure 2. Given a user selection query (that is either recommended by SunDEx or manually specified by the user), the SDE engine first extracts from the database the corresponding reviewer, item and rating groups. It then sends those groups to the RM-Set generator which returns a k-size set of diverse rating maps describing the most interesting trends in the current rating group. Each rating map $rm$, is then passed to the Recommendation Builder which returns the top-o most interesting next-step operations associated with $rm$. The SDE Engine then selects the overall top-o operations with the highest utility (among all generated k×o operations), and displays the selected rating maps and recommendations to the user. To speed-up computation, the SDE Engine calls the Recommendation Builder several times in parallel, each time with a different rating map.

**System UI:** The user interacts with the system using a dedicated UI (see Figure 3). The user investigates a rating group, by specifying the attribute-value pairs of interest defining the reviewer and item groups. The selection is done using a simple drop-down menu, or, for advanced users, by providing SQL predicates using the advanced screen (see Figure 3(a)). When the user investigates a rating group she can decide whether she wants to perform a recommended operation, or to provide an operation of her own. By clicking on “Apply Selection”, the corresponding rating group is displayed alongside a set of rating maps. By clicking on “Get Recommendation”, a pop-up window depicting next-step recommendations appears (Figure 3(b)). To select one recommended operation, the user may click on “Apply Selection” associated with it.

We next briefly describe the operation of the **RM-Set Generator** and the **Recommendation Builder** modules. Full details can be found in [7].

**RM-Set Generator:** The RM-Set Generator is composed of two modules: (1) **RM-Generator** that outputs, w.h.p., the top $lxk$ rating maps with the highest DW utilities; (2) **RM-Selector** that selects the most diverse k-size set of rating maps. We next briefly describe these modules.

**RM-Generator.** This module prunes low-utility rating maps, generating only the top $lxk$ maps with the highest utilities, where $l$ is a constant $\geq 1$. To this end, we adapted the sharing and pruning techniques of [6] for identifying high-utility rating maps and reduce computational costs. A main difference is that in our setting, (and unlike in [6] where the utility of a rating map is defined by a single score), the utility of a rating map is the maximum of 4 criteria. Thus, the key challenge here is to adapt these optimizations to our context.

**RM-Selector.** Our goal is to select the most diverse k-size set of rating maps, among the rating maps returned by the RM-Generator. We define the diversity of a set of rating maps to be the minimum distance between two selected maps. Here we use the Earth Mover’s Distance (EMD) to measure the distance between rating maps, a measure that was shown to be well-adapted for comparing rating maps [4], [6]. EMD ensures that rating maps having different shapes are selected. Our experimental results over real-life data show that this also increases the probability of choosing rating maps aggregated by different attributes, thereby exposing different data facets. This module employs the simple and efficient GMM algorithm [12] to identify a diverse k-size set of rating maps, which achieves a 2-approximation factor.

**Recommendation Builder:** Recall that an operation $q$ is a selection criteria defined over the underlying reviewer and item groups (i.e., $g_U$ and $g_I$) of $g_R$. Namely, $q$ is a set of attribute-value pairs, defined as the union of $g_U$ and $g_I$. Let $q'$ denote the current selection operation over a rating group $g_R$, and let $q$ denote a next-step operation. Although the space of possible choices for $q$ is very large, it is natural to expect that a user would be interested in a small adjustment to the current selection query [13]. Thus, to ensure that operation recommendations are understandable to users and preserve their train of thought, we limit $q$ to be different from $q'$ in at most 2 attribute-value pairs. Namely, $q$ may add a new attribute-value pair to $q'$, and may remove or change one of the existing attribute-value pairs in $q'$.

For each candidate operation, the essence of the resulting
An example of exploration step screen with a rating group is presented to the user in the form or a set of rating maps. Correspondingly, we define the utility of an operation $q$ to reflect the utility scores of the resulting rating maps. Namely, the utility of $q$ is defined as the sum of the DW utilities of rating maps selected after applying $q$. To compute the utility of an operation $q$, the Recommendation Builder uses the RM-set Builder, to find the $k$-size set of rating maps to be displayed in the next step. We can compute the utility scores of $x$ operations simultaneously, where $x$ is the number of available cores. Finally, given a rating map $rm$, the Recommendation Builder returns the top-$o$ operations associated with $rm$ with the highest utility scores.

III. DEMONSTRATION

We demonstrate the operation of SubDEx over three real-world subjective datasets: 12) Movielens1, which contains reviewers’ ratings on movies; (2) Yelp2, which contains people’s reviews of various businesses, including restaurants; (3) Hotel Review3, which consists of reviewers’ reviews on hotels. For the last two datasets, following [1], we extracted from the reviews text the rating scores for multiple rating dimensions (e.g., food, service, and ambiance for restaurants). We evaluate two aspects of SubDEx (i) learnability and usability, showing the ability of users to use the functionalities of SubDEx for different information needs, and (ii) scalability, examining how different parameters affect the performance.

Learnability and usability: We demonstrate the learnability and usability of the system via two scenarios.

Identifying special data characteristics. We simulate a scenario where a data analyst seeks “irregular” groups. An irregular group is described by two or three attributes–values shared by the reviewers (resp., items), whose rating scores for the same rating dimension have all been set to 1. This scenario simulates a common real-life event where the goal is to identify special data characteristics. To examine the benefit of guidance during exploration, we will randomly assign each participant with one of the optional exploration modes, and will ask her to load one of the datasets. The participants would then use the system to find the irregular groups.

Insight extraction. In the second scenario, we will use SubDEx for the task of insight extraction - a common goal of data exploration. For all examined datasets, the Kaggle platform4 contains several EDA notebooks, manually created by fellow data scientists to demonstrate their EDA process in obtaining insights. From these notebooks, we gathered three lists containing between 5 to 10 insights on each dataset. An example of insight on MovieLens is that the average rating score young adult reviewers gave to thriller movies is significantly higher than that of adult reviewers. Here again, each participant can choose a dataset, and will be randomly assigned with one of the exploration modes.

In both scenarios, the audience can examine statistics describing the aggregated results of other participants. These statistics include the average number of exploration steps, and the average precision and recall. These statistics are obtained by aggregating the results by different exploration modes and by different datasets.

Scalability: Last, the audience will be allowed to look “under the hood”, examining the efficiency of our algorithms. For this part of the demonstration, we will use growing fragments of the underlying database, showing the effect of different data and system parameters on performance.

Acknowledgment: This work has been partially funded by the European Union’s Horizon 2020 research and innovation program (grant agreement No 863410), the Israel Science Foundation, the Binational US-Israel Science Foundation, Tel Aviv University Data Science center, and eBay Israel.

REFERENCES
