CHARLES: Change-Aware Recovery of Latent Evolution Semantics in Relational Data

Shiyi He
University of Utah
Salt Lake City, USA
shiyi.he@utah.edu

Alexandra Meliou
University of Massachusetts Amherst
Amherst, USA
ameli@cs.umass.edu

Anna Fariha
University of Utah
Salt Lake City, USA
afariha@cs.utah.edu

ABSTRACT

Data-driven decision-making is at the core of many modern applications, and understanding the data is critical in supporting trust in these decisions. However, data is dynamic and evolving, just like the real-world entities it represents. Thus, an important component of understanding data is analyzing and drawing insights from the changes it undergoes. Existing methods for exploring data change list differences exhaustively, which are not interpretable by humans and lack salient insights regarding change trends. For example, an explanation that *semantically* summarizes changes to highlight gender disparities in performance rewards is more human-consumable than a long list of employee salary changes.

We demonstrate CHARLES, a system that derives *semantic* summaries of changes between two snapshots of an evolving database, in an effective, concise, and interpretable way. Our key observation is that, while datasets often evolve through point and other small-batch updates, rich data features can reveal *latent semantics* that can intuitively summarize the changes. Under the hood, CHARLES compares database versions, infers feasible transformations by fitting multiple regression lines over different data partitions to derive change summaries, and ranks them. CHARLES allows users to customize it to obtain their preferred explanation by navigating the accuracy-interpretability tradeoff, and offers a proof of concept for reasoning about data evolution over real-world datasets.

1 INTRODUCTION

The task of data understanding is of prime importance in today's data-driven world. To make sense of data, existing data summarization systems enable users to interactively understand the content of a static database [4]. However, data is dynamic, evolving over time, and summarization techniques for static databases are ineffective at explaining this data evolution. Datasets often evolve through many tuple-level or small-batch-level updates. However, exhaustively listing all such fine-grained changes overwhelms human analysts. Fortunately, rich features in the data have the potential to concisely summarize such fine-grained changes, offering an "explanation" that captures the salient *semantics* of the data's evolution.

To understand change, we need to understand its mechanism, quantification, and cause. This information is hard to extract from change logs, which are often unavailable or inaccessible to end users. Even if available, their format is often unsuitable for consumption by non-experts, and, thus, interpreting such large change logs to understand data changes poses a significant hurdle for data consumers. Data versioning techniques can trace the locations and quantities of changes, but high-level trends are not typically obvious at that fine granularity. Instead, changes should be summarized at a coarser granularity to reveal the underlying causes and mechanisms.

name	gen	edu	exp	salary	bonus
Anne	F	PhD	2	$230,000	$23,000
Bob	M	PhD	3	$250,000	$25,000
Amber	F	MS	5	$160,000	$16,000
Allen	M	MS	1	$130,000	$13,000
Cathy	F	BS	2	$110,000	$11,000
Tom	M	MS	4	$150,000	$15,000
James	M	BS	3	$120,000	$12,000
Lucy	F	MS	4	$150,000	$15,000
Frank	M	PhD	1	$210,000	$21,000

(a) 2016 snapshot

name	gen	edu	exp	salary	bonus
Anne	F	PhD	3	$230,000	$25,150
Bob	M	PhD	4	$250,000	$27,250
Amber	F	MS	6	$160,000	$17,440
Allen	M	MS	2	$130,000	$13,790
Cathy	F	BS	3	$110,000	$11,000
Tom	M	MS	5	$150,000	$16,400
James	M	BS	4	$120,000	$12,000
Lucy	F	MS	5	$150,000	$16,400
Frank	M	PhD	2	$210,000	$23,050

(b) 2017 snapshot

Figure 1: Employee salaries have evolved over a year, with the *bonus* attribute increasing by 8–10% (highlighted in yellow). Context and trends of these changes are not apparent from the point updates.

EXAMPLE 1. *Figure 1 presents two snapshots of a salary database in (a) 2016 and (b) 2017. In 2016,* `bonus` *was a flat 10% of* `salary` *for all employees. In contrast, we observed no such straightforward trend in 2017. In some cases, the value of* `bonus` *differs from last year's value (highlighted in yellow), while in some cases they are identical (for Cathy and James). Furthermore, the difference ranges from 8% to 10% and is not identical for everyone. Simply knowing that* `bonus` *changed from last year leaves one unsatisfied, as it is not obvious what is the underlying trend behind such non-uniform changes.*

It turns out that the company opted for a policy to reward long-serving employees and promote educational advancement. In 2017, the company decided to depart from a flat-rate bonus to a customized one. The new scheme for bonus calculation is influenced by three principles: (1) no one should receive lower bonus than the previous year, (2) employees with higher level of education should be rewarded more, and (3) long-serving employees should be rewarded more.

This is not immediately apparent by just looking at the data, since `bonus` *for 2017 is no longer directly tied to* `salary`*, as was the case in 2016. Instead, it is calculated based on a combination of last year's* `bonus`*, employee's education (*`edu`*), and years of experience (*`exp`*). Specifically, the following rules accurately explain the change trend:*

- ***R1**: Employees who have a PhD receive a 5% increase on last year's bonus, plus flat $1000.*
- ***R2**: Employees who have an MS and served for at least 3 years receive a 4% increase on last year's bonus, plus flat $800.*
- ***R3**: Employees who have an MS and served for less than 3 years receive a 3% increase on last year's bonus, plus flat $400.*

There are two desirable properties for a *change summary*: (1) it should be *precise*, i.e., be able to explain the changes *accurately*, and (2) it should be *interpretable* and *succinct* for easy human consumption. Note that there is a natural tension between these two desirable properties. Consider the following change summary:

- ***R4**: Everyone receives about 6% increase on last year's bonus.*

R4 is more interpretable (more succinct and human-consumable) than {**R1**, **R2**, **R3**}; however, **R4** does not accurately capture the

change, while {**R1**, **R2**, **R3**} does. In contrast, one can provide a change summary by listing each individual cell that changed. However, such a summary—despite being very precise—would lack interpretability as this level of detail overwhelms the user.

CHARLES. To meet the requirements of accuracy and interpretability, we developed CHARLES (Change-Aware Recovery of Latent Evolution Semantics), a system for producing a *semantic summary of changes* between two snapshots of a relational database, while striking a balance between accuracy and interpretability. Our key observation is that data changes are often driven by some underlying policies and the patterns within data evolution, as manifested by the changes, can potentially recover those policies. In this work, we focus on temporal changes and assume that, given a *source* dataset (earlier version) and a *target* dataset (later version) of identical schema, the latter is obtained via a set of update operations over the former. Furthermore, we assume that no tuples are inserted or deleted; only (numerical) values of various cells are altered.

The core challenge in this problem is to derive a partitioning of the tuples, such that tuples within each partition conform to a uniform "transformation" of reasonable complexity. As proof of concept, CHARLES uses k-means clustering to guide the search for data partitions based on a subset of data attributes (e.g., education and year of experience). Once approximate partitions are discovered, CHARLES applies linear regression to find the most suitable transformations to capture the changes within each partition (e.g., $\text{bonus}_{2017} = 1.05 \times \text{bonus}_{2016} + 1000$). The output is a *linear model tree* [8] (Figure 2), where the path from the root to a leaf defines a partition and the leaf defines the transformation (a linear model).

Furthermore, CHARLES enhances user experience by (1) *customization*—users can specify system parameters such as the maximum number of attributes they want to see in the change summary—and (2) *visualization*—they can interactively inspect different partitions of the data and the corresponding change trends.

Limitations. CHARLES focuses on finding an interpretable summary of data changes based only on the data, without any knowledge of external information. While the change summary produced by CHARLES may not always match the factual explanation (e.g., when change is due to some external factors), it nevertheless helps facilitate the development of hypotheses about the underlying causes of these changes. While CHARLES relies on linear models to capture change trends, this can be extended by augmenting the data with nonlinear features. However, nonlinear models are less interpretable, which justifies our choice of linear models.

Related work. Prior work [1] has studied the problem of exploring the entire history of changes in a database, but is limited to syntactic or raw changes, suitable for historical change exploration involving a particular entity. Database comparator tools, such as PostgresCompare [7] and RDBMS version control system OrpheusDB [3], only look for syntactic changes—values are changed, objects are altered, rows are removed or added—which is not concise enough to provide high-level insights of the changes. Data-diff [11] explores change in distributions of datasets, specialized in the context of data wrangling. Muller et al. [6] describe change in two datasets in terms of "update distance", defined by the minimal number of insert, delete, and modification operations necessary. However, none of these works focus on summarizing changes between two databases.

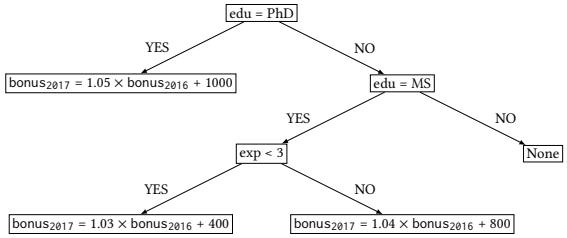

Figure 2: A linear model tree explaining diff in datasets in Figure 1.

Explain-Da-V [10] is closest to CHARLES as it also explains transformations that convert a source dataset to a target dataset. However, it focuses on semantics of schema and data format transformations (e.g., data extraction, row deletion, adding attributes representing length of an attribute). In contrast, we focus on semantic changes where values of attributes are altered based on interactions among other attributes, with the goal of explaining the evolution of real world entities represented by datasets.

Local explanations for ML models [9] is inapplicable in our use case as they focus on classification. In contrast, we focus on the challenging problem of *pattern-based clustering*, where we need to find clusters where elements share the same change patterns. However, a natural cyclic dependency exists in finding shared change patterns and clustering, as shared patterns can only be discovered once clusters are formed, where the clusters must be formed such that elements within the same cluster exhibit identical change patterns.

In our demonstration, participants will witness how CHARLES generates change summaries from two datasets, tailored to user preferences, and enables them to effectively gain insights about data changes. We proceed to describe our solution sketch in Section 2 and then provide the demonstration outline in Section 3.

2 SOLUTION SKETCH

Given two datasets of identical schema—a source dataset $\mathcal{D}_s$ and a target dataset $\mathcal{D}_t$—and a numerical attribute of interest a_i, we aim to produce a *ranked list* of *change summaries* that capture the changes observed between $\mathcal{D}_s(a_i)$ and $\mathcal{D}_t(a_i)$. Each change summary consists of a set of *transformations* over different data partitions. We rank the summaries based on their *scores*, which indicate how well they can strike the balance between accuracy and interpretability. We assume that $\mathcal{D}_s$ and $\mathcal{D}_t$ contain the same real-world entities, i.e., only values of non-primary-key attributes were modified, and there were no insertions or deletions of tuples.

Change summary and conditional transformation. Our unit of explanation within a change summary is a *conditional transformation* (CT), which comprises a *condition* and a *transformation*. A summary $S = \{CT_1, CT_2, \dots\}$ comprises a set of CTs. The condition explains why a change happened, and the transformation describes the change itself. For instance, the following CT explains that employees with a PhD got 5% increase in bonus plus $1000.

$$\underbrace{\text{edu = PhD}}_{\text{Condition}} \longrightarrow \underbrace{\text{new\_bonus = 1.05 } \times \text{ old\_bonus + 1000}}_{\text{Transformation}}$$

Desiderata for change summary. A desirable change summary must ensure that (1) all or most of the data changes are sufficiently covered and (2) the summary itself is interpretable and succinct for human consumption. To this end, we introduce $Score(S) \in [0, 1]$

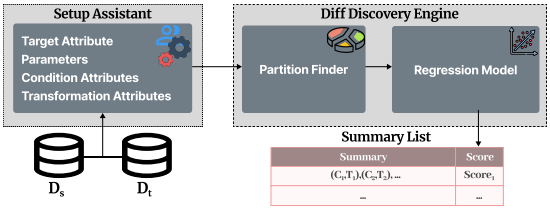

Figure 3: CHARLES overview: The setup assistant helps users choose system parameters such as attributes to consider for conditions and transformations, and the diff discovery engine summarizes the changes based on data partitioning and fitting regression lines.

for a summary S, which indicates how well S can represent the differences between $\mathcal{D}_s(a_i)$ and $\mathcal{D}_t(a_i)$.

$$Score(S) = \alpha \times Accuracy(S) + (1-\alpha) \times Interpretability(S)$$

Here, we model *Accuracy* by the inverse L_1 distance between $\hat{\mathcal{D}}_s(a_i)$ and $\mathcal{D}_t(a_i)$, where $\hat{\mathcal{D}}_s$ is the transformed dataset obtained by applying the CTs in S on $\mathcal{D}_s$, and α is a system parameter that controls the interplay between accuracy and interpretability.

To achieve interpretability, we prioritize the following characteristics of summaries:

- *Smaller summaries.* A summary with fewer CTs is preferable, as it leads to increased conciseness.
- *Simpler conditions and transformations.* A condition consists of a series of descriptors that identify specific segments of the data, so we prefer a simpler condition with fewer descriptors. E.g., the transformation "All Female employees received 5% bonus" is more interpretable than "All Asian, European Females, or Females working in HR received a 5% bonus". Similarly, a transformation involves a linear equation, so a transformation with fewer variables in the equation is preferred.
- *Conditions with higher data coverage.* A condition that yields a small data partition explains little of the change. Thus, we prefer conditions with higher coverage, yielding larger partitions.
- *Higher "normality" for conditions and transformations.* Conditions and transformations may involve numeric constants. We prefer the ones involving more "normal" values. E.g., the condition "Age > 25" is more normal than "Age > 23.796", and 5% for a salary increase is more normal (and interpretable) than 2.479%. We rely on domain expertise to learn such notions of normality.

Enhancing interpretability without significantly compromising accuracy results in a more effective summary, with a higher overall score. The parameter α controls the tradeoff between accuracy and interpretability and has a default value of 0.5. CHARLES is designed to cater to a diverse audience, allowing novices to bypass system parameter tuning and experts to adjust parameters.

CHARLES architecture. Figure 3 shows the CHARLES architecture. It consists of two components: the *setup assistant*, which helps the users (optionally) tune system parameters, and the *diff discovery engine*, which is responsible for generating the change summaries.

Setup assistant. For datasets with many attributes, the search space for possible summaries can explode. A user may help narrow down the set of relevant attributes, which has the dual benefits of reducing the search space, and pivoting the change summaries around attributes of interest. However, users may struggle to provide such input when they are unfamiliar with the schema. CHARLES addresses this challenge by estimating the influence of other attributes on the target attribute using correlation analysis and presents to the user a shortlist of attributes that are most likely to be effective for explaining the changes. By default, CHARLES presents candidate attributes for both condition ($\mathcal{A}_{cond}$) and transformation ($\mathcal{A}_{tran}$) that have a correlation with the target attribute greater than 0.5. Additionally, CHARLES allows the user to narrow down or expand the candidate attributes by providing two parameters, c and t, where c defines the maximum number of attributes $\in \mathcal{A}_{cond}$ to use for partition discovery, and t denotes the maximum number of numerical attributes $\in \mathcal{A}_{tran}$ to use to fit the linear model within each partition.

Diff discovery engine. The diff discovery engine comprises (1) partition discovery, which identifies potentially significant partitions, and (2) transformation discovery, which fits a linear regression model within each partition. The goal to find a list of change summaries $SL = \{(S_1, score_1), (S_2, score_2), \ldots\}$, ordered by decreasing order of their scores. For each summary S_i, CHARLES discovers different data partitions specified by conditions (defined by a subset of attributes in $\mathcal{A}_{cond}$), where each partition conforms to a specific transformation (defined by a subset of attributes in $\mathcal{A}_{tran}$).

Based on $\mathcal{A}_{cond}$, $\mathcal{A}_{tran}$, and the parameters c and t , CHARLES enumerates all possible combinations of attributes to use for partitioning and generating transformations. For example, for the parameters $c = 3$ and $t = 2$, CHARLES will consider all subsets of $\mathcal{A}_{cond}$ with cardinality ≤ 3 as partitioning attributes and all subsets of $\mathcal{A}_{tran}$ with cardinality ≤ 2 as transformation attributes.

Partition discovery. For a specific set of condition attributes $C \subseteq \mathcal{A}_{cond}$ and transformation attributes $\mathcal{T} \subseteq \mathcal{A}_{tran}$, CHARLES first fits a linear regression model for a_i, over the entire data, based on the attributes in $\mathcal{T}$. Then CHARLES performs K-means clustering, based on the distance from the regression line, to discover potentially meaningful partitions in terms of the attributes in C.

Transformation discovery. For each discovered partition, CHARLES again fits a linear regression model based on $\mathcal{T}$ to generate a transformation. All such transformations over different partitions, together, result in a change summary, which can be represented using a linear model tree similar to Figure 2. Once all summaries $\{S_1, S_2, \ldots\}$ are generated within the specified parameters, CHARLES computes the *Score* of each summary, ranks the generated summaries according to the descending order of their scores, and returns the ranked list of summaries $\{(S_1, score_1), (S_2, score_2), \ldots\}$.

This is a proof-of-concept prototype implementation of CHARLES. Other methods of partitioning and transformation discovery are certainly possible, but we defer deeper investigation to future work.

3 DEMONSTRATION

We will demonstrate CHARLES on a real-world dataset [5] representing salary information for all active, permanent employees of Montgomery County, MD for the years 2016 and 2017. The dataset contains information about employee salaries over 8 attributes, including Department, Department Name, Division, Gender, Base Salary, Overtime Pay, Longevity Pay, and Grade. Figure 4 shows CHARLES's graphical user interface. During the demonstration, we will guide the participants through ten steps. We have annotated each step with a circle.

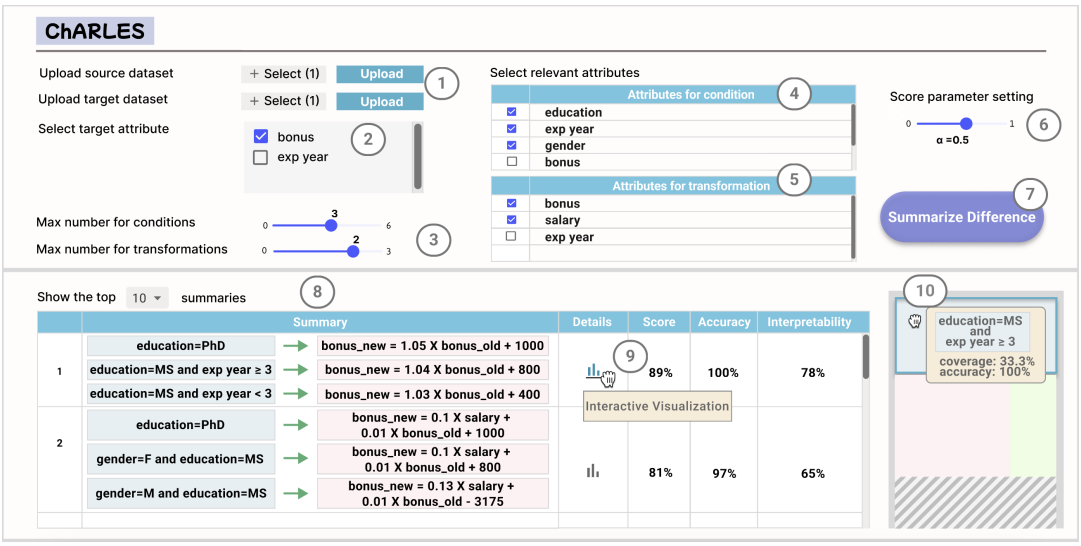

Figure 4: The ChARLES demo: (1) upload datasets, (2) select the target attribute, (3) specify the maximum number of attributes for condition and transformation, (4) ChARLES selects attributes for condition automatically, (5) ChARLES selects attributes for transformation automatically, (6) tune score parameter α, (7) request change summaries, (8) ChARLES presents a list of ranked summaries, with their overall scores, and scores for accuracy and interpretability, (9) click on a summary for more details, (10) detailed visualization of data partitions.

Step (1) (Uploading datasets) The user uploads two dataset versions they want to compare. For ease of exposition, we use the toy datasets of Example 1 in this demo scenario.

Step (2) (Selecting the target attribute) Next, the user chooses the target attribute that manifests changes they wish to investigate. For our scenario, the user chooses "bonus".

Step (3) (Setting parameters) Next, the user chooses the maximum number of condition attributes to use for partitioning (3) and the maximum number of transformation attributes (2).

Steps (4) & (5) (Attribute selection) ChARLES presents a ranked list of attributes that are most promising for condition (4) and transformation (5) attributes. Based on the user specifications, ChARLES selects the top 3 results from the condition attributes list and the top 2 results from the transformation attributes list. Users can accept this default selection or interactively filter out undesired attributes and select other attributes. In our case, the user accepts the default selection: "education", "exp year", and "gender" as potential condition attributes and "bonus" (of the previous year) and "salary" as potential transformation attributes.

Steps (6)–(8) (Change summaries) The value of α represents the weight of accuracy in the *Score* function, which is set to 0.5 by default. Users can modify this parameter according to their requirements (6). For those aiming for a more interpretable summary, adjusting α to a lower value can shift the balance towards interpretability at the expense of accuracy. The user then requests to generate change summaries (7) and ChARLES displays the summaries (8). Each summary comprises a set of conditional transformations—where conditions are in light blue and transformations are in light pink—followed by an option to visualize (9), overall score, accuracy, and interpretability scores. In this scenario, the first summary produced by ChARLES reflects the scenario described in Example 1, which incurs a very high score of 89%. By default, ChARLES presents the 10 top-scoring summaries.

Steps (9) & (10) (Visualization) To better understand a summary, the user requests more details (9). ChARLES offers an interactive visualization (10) comprising several non-overlapping rectangles, each representing a data partition achieved via applying the conditions. The size of each rectangle corresponds to its data coverage. E.g., 33.3% employees fall within the top partition. For each partition, additional details—such as partitioning condition, data coverage, accuracy of the transformation—are revealed when the user hovers over these rectangles. The bottom partition, marked by diagonal patterns, indicates that no change was observed there.

Demonstration engagement. Our target users are data analysts, decision makers, and data enthusiasts who want to understand data change trends. After our guided demonstration, participants will be able to plug their own datasets into ChARLES. We will also make additional datasets [2] available. Through the demonstration, we will showcase how ChARLES can semantically summarize changes between two datasets.

REFERENCES

[1] T. Bleifuß, L. Bornemann, T. Johnson, D. Kalashnikov, F. Naumann, and D. Srivastava. 2018. Exploring Change - A New Dimension of Data Analytics. *PVLDB* (2018).
[2] Forbes World's Billionaires List [n.d.]. `https://www.forbes.com/billionaires/`.
[3] S. Huang, L. Xu, J. Liu, A. Elmore, and A. Parameswaran. 2017. OrpheusDB: Bolt-on Versioning for Relational Databases. *PVLDB* (2017).
[4] M. Joglekar, H. Garcia-Molina, and A. Parameswaran. 2019. Interactive Data Exploration with Smart Drill-Down. *IEEE* (2019).
[5] Montgomery Police Dataset [n.d.]. `https://data.montgomerycountymd.gov/`.
[6] H. Müller, J. C Freytag, and U. Leser. 2006. Describing differences between databases. In *CIKM*.
[7] PostgresCompare [n.d.]. `www.postgrescompare.com/`.
[8] D. Potts. 2004. Incremental learning of linear model trees. In *ICML*, Vol. 69. ACM.
[9] M. Ribeiro, S. Singh, and C. Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *SIGKDD*. 1135–1144.
[10] R. Shraga and R. Miller. 2023. Explaining Dataset Changes for Semantic Data Versioning with Explain-Da-V. *PVLDB* (2023).
[11] C. Sutton, T. Hobson, J. Geddes, and R. Caruana. 2018. Data Diff: Interpretable, Executable Summaries of Changes in Distributions for Data Wrangling. In *SIGKDD*.