Provectories: Embedding-based Analysis of Interaction Provenance Data

Conny Walchshofer, Andreas Hinterreiter, Kai Xu, Holger Stitz, and Marc Streit

Abstract—Understanding user behavior patterns and visual analysis strategies is a long-standing challenge. Existing approaches rely largely on time-consuming manualprocesses such as interviews and the analysis of observational data. While it is technically possible to capture a history of user interactions and application states, it remains difficult to extract and describe analysis strategies based on interaction provenance. In this paper, we propose a novel visual approach to meta-analysis of interaction provenance. We capture single and multiple user sessions as graphs of high-dimensional application states. Our meta-analysis is based on two different types of two-dimensional embeddings of these high-dimensional states: layouts based on (i) topology and (ii) attribute similarity. We applied these visualization approaches to synthetic and real user provenance data. From our visualizations, we were able to extract patterns for data types and analytical reasoning strategies.

Index Terms—Visualization techniques, Information visualization, Visual analytics, Interaction Provenance, Sensemaking

1 Introduction

Understanding the analytical reasoning process of users who work with interactive tools in general, and with also visualization tools in particular, has been an active research topic. One way to gain more insights into how users work with such tools is to record *interaction provenance* data, which describes the lineage of data, system states, visualizations used, and user interactions. It is typically recorded in the form of *protocols*, such as audio/video recordings [3], usage logs [14], and user notebooks [44]. In the human-computer interaction (HCI) community, these protocols are analyzed in an attempt to better understand a user's behavior and intentions [24].

In recent years, the visualization community has also recognized the potential of insights gained from capturing [4], [9], [27], visualizing [5], [40], and interpreting provenance [38] from user interactions with visualization tools. According to the distributed cognition approach by Hollan et al. [18], a close relationship exists between users' activities and their thought processes. Pohl et al. [33] argued that visualizations of interaction provenance data can be used to make sense of users' reasoning processes. However, there are few approaches that support effectively the *meta-analysis* of analytic provenance as defined by Ragan et al. [35].

The primary contribution of our work is *Provectories*, an approach that helps visualization researchers, designers, and developers to better understand the behavioral patterns and analytic strategies of users. As shown in Figure 1, we transform application states of the interaction provenance into feature vectors and visualize them using two different types of embeddings: (1) a *topology-driven* layout that aims to show patterns of states based on their connectivity and (2) an *attribute-driven* layout that visualizes states based on their similarity. These embeddings give rise to visual patterns that can be related to specific user actions. *Provectories* can be applied to a broad spectrum of use cases and tools, ranging from single interactive visualizations to feature-rich tools such as Tableau and Microsoft Power BI.

As secondary contributions, we describe the visual patterns that we extracted from *Provectories* visualizations of synthetic user interactions. We then show how these patterns can be identified in visualizations of provenance data from real-world user interactions. We describe insights gained from single sessions and patterns resulting from the combination of multiple sessions. Finally, we discuss the relative strengths and weaknesses of two different types of layouts used in the *Provectories* workflow.

We structured the paper as follows. In Section 2 we discuss existing approaches to interaction provenance representation and analysis. In Section 3 we present application scenarios and introduce an illustrative example. In Section 4 we describe the *Provectories* workflow conceptually; implementation details are given in Section 5. In Section 6 we present the results of applying *Provectories* to synthetic and real-world interaction provenance data and discuss the advantages of two different layouts. We then summarize limitations of our new visual analysis approach in Section 7. Section 8 concludes the paper.

- *Conny Walchshofer, Andreas Hinterreiter, and Marc Streit are with Johannes Kepler University Linz, Austria. E-mail: {conny.walchshofer, andreas.hinterreiter, marc.streit}@jku.at.*
- *Kai Xu is with Middlesex University London, UK. E-mail: k.xu@mdx.ac.uk.*
- *Holger Stitz is with datavisyn GmbH, Austria. E-mail: holger.stitz@datavisyn.io.*

Manuscript received December XX, 2020

2 Related Work

In this section we describe how interaction provenance has been defined in the literature and discuss why visualization researchers might study interaction provenance. We then discuss previous approaches to meta-analysis, in particular those based on visualizations of provenance data.

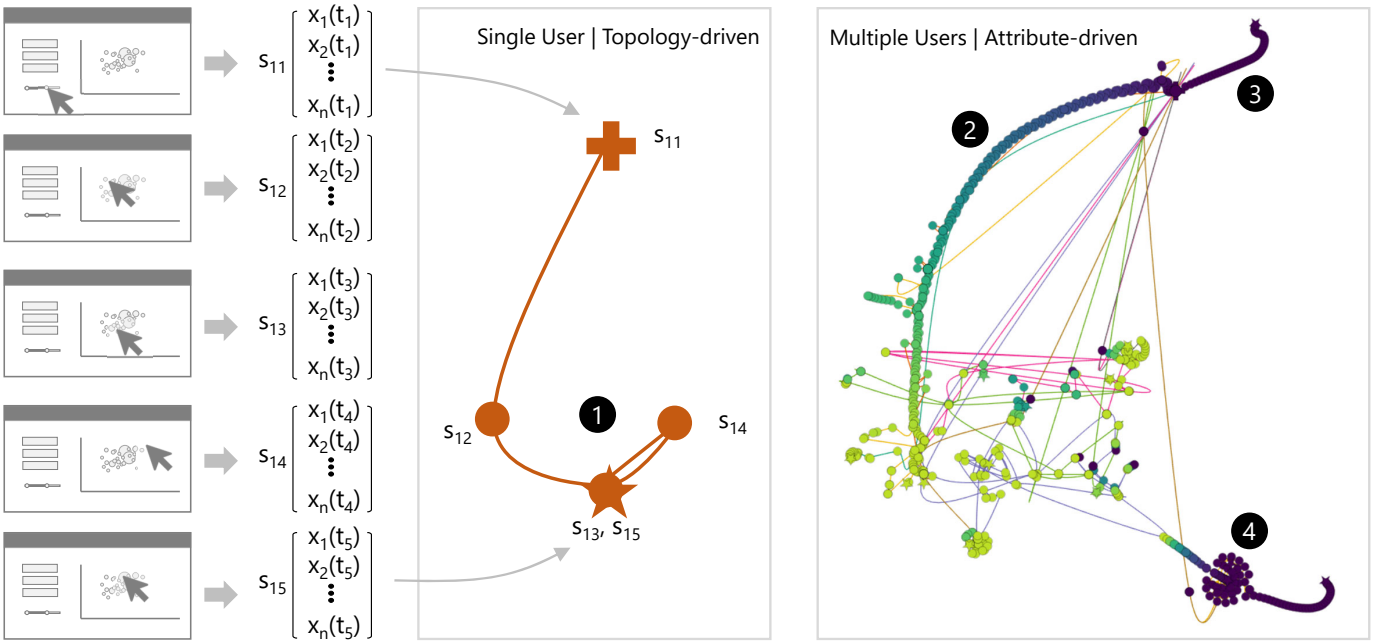


Fig. 1: Identifying meta-analysis patterns for interaction provenance with a topology-driven layout (left: force-directed) for a single user session and with an attribute-driven layout (right: t-SNE projection) for multiple analysis sessions. The circular annotations highlight ❶ a loop back within the analysis process, ❷ a chain of numerical value changes, ❸ a selection of multiple countries, and ❹ a cluster of single selected countries. The colors of the states indicate numerical values from **1800** in violet to **2015** in light green.

2.1 Interaction Provenance

Ragan et al. introduced an organizational framework for different types of provenance in visualization and data analysis [35]. They defined *interaction provenance* as focusing on "the history of user actions and commands with a system" [35, p.35]. There are various motivations for logging explicit and observable user interactions, such as selections, clicks, keystrokes, and mouse movement. Interaction logs can be used for the purposes of collaboration, reproducibility, storytelling and retrieval [15], [39]. More closely related to our work, interaction provenance can be analyzed to understand how users interact with a visualization system [14] or to measure the effectiveness of a tool [8]. The process of making sense of such logs is referred to as provenance meta-analysis.

2.2 Provenance Meta-Analysis

Ragan et al. [35] described meta-analysis as one of six purposes for interaction provenance tracking. Xu et al. [45] provided a spectrum of possible reasons for conducting meta-analyses on provenance data. Reviewing an analysis process to understand analytic strategies of users has been identified as an important task [9], [11], [35], which can be implemented in various ways.

Wei et al. [43], for instance, employed clickstream data to analyze purchase patterns. The data is labeled with predefined actions (e.g., selection of a category, setting a price) and analyzed based on the ordered sequence. Heer et al. [16] described how users interact with a visual analytics tool by evaluating aggregated collections of history sessions. Pohl et al. [25] qualitatively analyzed interaction provenance on the basis of thinking-aloud protocols. They identified various strategies that users applied to interpret and understand visualizations: comparing, laddering, explaining (storytelling), summarizing, eliminating, and verifying.

Provectories aims to identify such user strategies as visual patterns. Thus, *Provectories* is a visualization-based approach to the meta-analysis of interaction provenance.

2.3 Provenance Visualization

According to a recent survey, interaction provenance is most commonly encoded as a temporally ordered sequence [45]. Visualizing interaction provenance in this way allows stepwise retracing of the individual interactions [4], [9], [11] and can thus convey the users' thought processes [23], [46]. However, sequential visualizations are less suitable for discovering patterns and relationships. They neither preserve interesting topological structures, such as loops or branches in a user's interaction path, nor convey a potential similarity between application states visited. These issues are addressed by the *topology-based* and *attribute-driven* visualization techniques in Provectories.

2.3.1 Topology-driven Layouts

Provenance data can be treated as a graph, with nodes representing states of a data item or application and edges representing actions of users that lead to transitions between the states. Graph-based provenance visualization can reveal patterns such as branching, cycles, or commonly revisited states (i.e., nodes with high connectivity).

VisTrails [6] is a graph-based visualization of workflow provenance. *GraphTrails* [10] is an exploration tool for network analysis that incorporates interaction-provenance on the fly. *VizCept* [7] is a collaborative analysis system for textual data that allows users to keep track of each other's findings and relationships in a shared topological concept map. The *Knowledge-Transfer Graph* by Zhao et al. [47] shows

a node-link visualization that aims to help researchers to externalize their thought processes in collaborative analyses.

Similarily, we use a force-directed graph layout for visually representing interaction provenance. In addition to this topology-driven layout, we also investigate and employ layouts in which the similarity between states determines the positions of the nodes.

2.3.2 Attribute-driven Layouts

The application states in a provenance log can be viewed as a high-dimensional time series rather than a graph. Bach et al. [2] proposed *TimeCurves* as a visualization technique for revealing similarity in high-dimensional time series. Time curves are trajectories through a two-dimensional embedding of the data points, which give rise to visual patterns such as clusters, cycles, U-turns and oscillations. Time curves are based on multidimensional scaling (MDS) for the embedding; similar visualizations can be constructed by means of other dimensionality-reduction techniques, such as PCA, *t*-SNE [42], and UMAP [26]. Time-curve-like visualizations have been used to visualize high-dimensional time series in a wide variety of application domains (e.g., dynamic graphs by van den Elzen et al. [41] and neural networks by Rauber et al. [36]).

In previous work, we used collections of time curves to visualize decision-making processes in games and puzzles, and described general patterns emerging in such visualizations [17]. In this work, we use the same approach for visualizing interaction provenance in an attribute-driven layout. *ModelSpace* by Brown et al. [5] is one of only a few approaches that make use of embedding-based techniques to visualize provenance data. However, the ModelSpace approach is limited to interactions with an underlying machine learning model. In contrast, *Provectories* is general enough to be applied to a wide variety of visual analytics systems. In addition, *Provectories* shows not only embedding-based (i. e., attribute-driven) layouts, but also topology-driven ones based on the connectivity of application states visited.

3 Requirements and Usage Scenario

We designed *Provectories* for the purpose of understanding user behavior patterns and analysis strategies from interaction provenance. Gleicher [13] enumerated three ways of comparing sessions: comparison between two items, between "a few" items, and between many items at the same time. With *Provectories*, we aim to cover all three aspects, performing meta-analysis to understand (i) a single user's analytical process and (ii) similar approaches by multiple users. Thus, *Provectories* uses two layouts to enable comparison between unique states from a single session, between unique states from multiple user sessions, and between contiguous states from multiple user sessions.

Single-session investigation focuses on understanding behavioral patterns and the overall analysis strategy of a single user. This type of investigation aims to answer questions such as whether a user encountered difficulties during the analysis and whether the user had a systematic search strategy or performed a somewhat untargeted exploratory analysis.

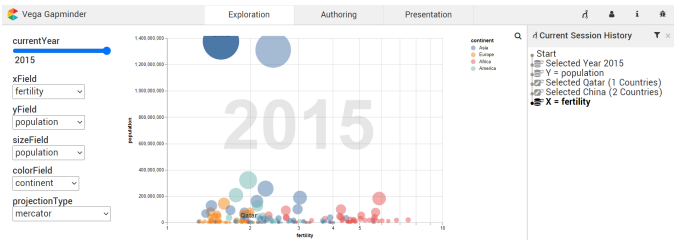


Fig. 2: Interface of the *Gapminder* visual analytics tool [39] with the history graph.

Multi-session investigation builds on single-session investigation, but focuses on the comparison of the interaction provenance from multiple users working with the same tool. Here, the goal is to understand the similarities and differences in analysis behavior between the users. This type of investigation aims to answer questions such as whether many users encounter the same difficulties, or how effective different analysis strategies are. Multi-session investigation can be divided into comparing sessions in which users perform (i) the same or similar tasks or (ii) different tasks.

3.1 Requirements

We derive the following requirements for single and multiples sessions from the existing literature [21], [29] and our prior research experience. To support single-session investigation, *Provectories* is designed to:

S1 Show the entire analysis sequence from beginning to end, following the temporal order;

S2 include the user interaction and/or system state information, such as the changes between two consecutive steps in the analysis sequence;

S3 facilitate the discovery of behavioral patterns, such as the occurrence of revisited or similar states within an analysis; and

S4 facilitate the discovery of any systematic analysis strategy, such as comparison and verification.

To support multi-session investigation, *Provectories* aims to:

M1 show all sessions simultaneously;

M2 facilitate the discovery of similar parts among the sessions; and

M3 facilitate the discovery of any patterns present.

3.2 Usage Scenario

We hereafter use *Gapminder* [37], [39] as a guiding example to explain how *Provectories* works. The Gapminder tool allows users to explore the development of countries over time. As outlined in Figure 2, it consists of a bubble chart in which each country is represented by a colored mark. Users can interactively map attributes, such as GDP, life expectancy, and child mortality, to either one of the axes or the size of the country marks, and change the year between 1800 and 2015 with a time slider. At any time, the application state can be fully described by the following information: the timestamp of the interaction; the data attributes mapped to x-axis, y-axis, mark size, and mark color; the year selected (between 1800 and 2015); and any countries selected.

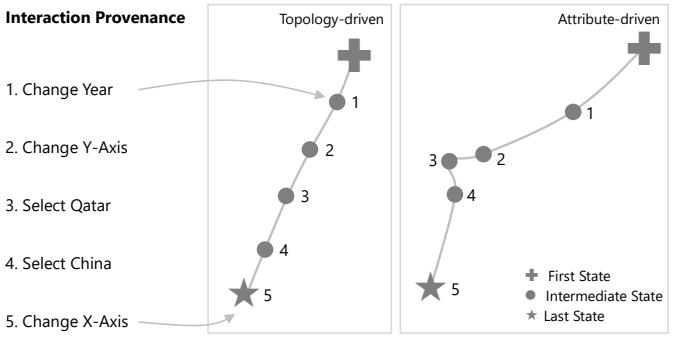


Fig. 3: Schematic illustration of the Gapminder usage scenario explaining how application states are mapped in the two layout variants.

In a simple analysis of the relationship between population and fertility among countries in 2015, the user can perform the following steps: (1) change the year to 2015; (2) change the data attribute for the y-axis to *population*; (3) select the country *Qatar*; (4) add *China* to the country selection; and (5) change the data attribute for the x-axis to *fertility*. This analysis results in the five applications states listed in Table 1.

For the purpose of meta-analysis, we use the sequence of application states visited by a user and display the interaction provenance in two layouts, see Figure 3. In both representations, ✚ and ★ indicate the beginning and the end of a session, respectively. When applying a force-directed layout (topology-driven), a chain of five successive states is visible. In contrast, the attribute-driven layout (calculated using t-SNE) places the states corresponding to selections of the countries (states 3 and 4) closer to each other than the other selected states. This is the result of an underlying "conceptual" or "analytical distance", which was defined to be smaller between states 2 to 4 than between the other ones.

4 Provectories

The fundamental workflow underlying the *Provectories* approach consists of three steps, as illustrated in Figure 4: 1 the application states resulting from one user's or multiple users' interactions with a visual analytics tool are recorded; 2 the application states are transformed to high-dimensional feature vectors; and 3 for the purpose of meta-analysis, the recorded analysis sessions are interactively visualized as trajectories through a two-dimensional embedding space based on various layout techniques.

TABLE 1
Interaction provenance for the *Gapminder* example. Here, x_0, y_0, s_0, and c_0 represent the default data attributes mapped to the axes, size, and color, respectively; Y_0 and C_0 represent the default initial selections for year and countries. **Bold text** indicates changes in the application state resulting from a user interaction.

Time	x-axis	y-axis	Size	Color	Year	Countries
t_0	x_0	y_0	s_0	c_0	Y_0	C_0
t_1	x_0	y_0	s_0	c_0	**2015**	C_0
t_2	x_0	**population**	s_0	c_0	2015	C_0
t_3	x_0	population	s_0	c_0	2015	**Qatar**
t_4	x_0	population	s_0	c_0	2015	Qatar, **China**
t_5	**fertility**	population	s_0	c_0	2015	Qatar, China

4.1 Logging of Application States

As indicated in Figure 4 1, the first step in the *Provectories* workflow consists of creating user interaction logs for a given visual analytics tool. Each user interaction (of a predefined set of interactions) triggers the logging of the updated application state. The complexity of the visual analytics tool and the goals of the subsequent meta-analysis determine the granularity of the application state and which interactions are to be logged. For the subsequent steps in the *Provectories* workflow it is important that each user session is stored as a temporally ordered list (**S1**) of potentially unstructured or heterogeneous data items which can be transformed to feature vectors.

4.2 Vectorization of Application States

In the second step of the *Provectories* workflow, the logged application states are transformed into numerical feature vectors (see Figure 4 2). This transformation serves two purposes: First, it provides a structured way to determine equivalent states for the subsequent topology-driven layout (see Section 4.3.1). Instead of performing the similarity check directly on the complete and potentially complex logged states, this "quantization" introduces an optional abstraction and/or simplification step. Second, it enables the calculation of distances between application states (see Section 4.3.2).

For simple applications, the complete state can be readily transformed into a numerical representation. In the case of the previously outlined *Gapminder* example, the application state is determined fully by one Boolean option (attribute encoded by color), three categorical options (attributes encoded by x-axis, y-axis, and size mark), one numerical selection (year), and one set selection (countries). For example, the state t_5 from the interaction provenance sequence in Table 1 is described by the set of choices (GDP, population, s_0, c_0, 2015, {Qatar, China}). One way to construct a feature vector from this list is to transform each of the choices/attributes individually and chain the results together. The individual encodings depend on the data type.

Cat For categorical attributes, a simple one-hot encoding is the obvious choice. For m possible values of the attribute, the one-hot encoding is a sequence of length m in which only the entry corresponding to the attribute value is 1 and all other entries are 0.

Bool Boolean attributes can be treated as categorical attributes with two options and encoded with a one-hot sequence of length 2. Alternatively, each Boolean attribute can be represented by a single integer that is either 0 or 1.

Num Numerical attributes require no further encoding.

Set Set attributes can be encoded by a sequence of zeros and ones of length l, where l is the cardinality of the complete set. Each entry corresponds to one element of the complete set, and is 0 if the element is not part of the subset (i.e., not selected) or 1 if it is part of the subset (i.e., selected).

These encoding rules lead to *provenance vectors* (**S2**). For

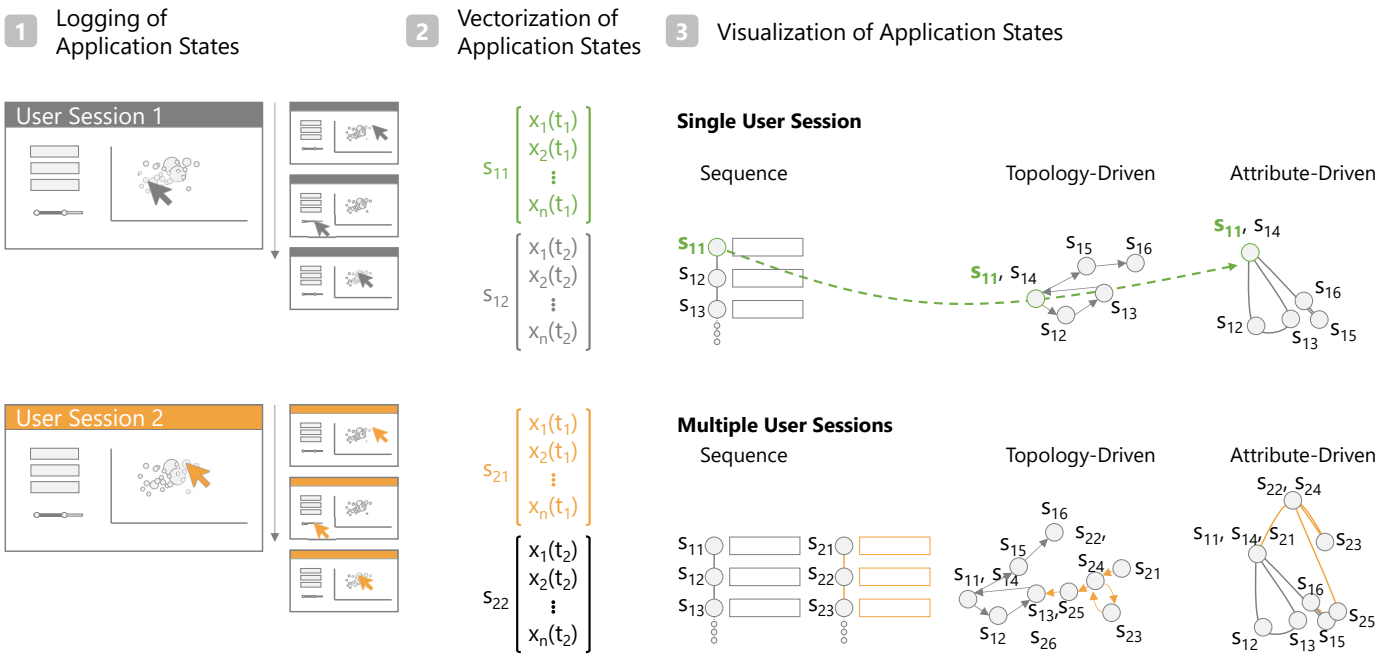


Fig. 4: *Provectories* workflow. (1) Interaction provenance is captured from a visual analytics tool. (2) Each interaction leads to a system state which is encoded as a feature vector such as s_{11}. (3) The sequence of provenance vectors is then visualized with a topology-driven or attribute-driven layout. Such a visualization can help to analyze a single session or to compare multiple sessions (such as Sessions 1 and 2) concurrently.

instance, the provenance vector for the state t_5 in Table 1 is:

$$t_5 = (\underbrace{1,0,0,0,0}_{\text{fertility}}, \underbrace{0,1,0,0,0}_{\text{population}}, \underbrace{0,1,0,0,0}_{s_0},$$
$$\underbrace{1,0}_{c_0}, \underbrace{2015}_{\text{year}}, \underbrace{0,1,0,\ldots,0,1,0,\ldots,0}_{\{\text{China, Qatar}\}}).$$

The recorded application states may include other types of data, such as text or images. In these cases, more elaborate vectorization techniques such as word embeddings or feature extractors may be required. Additionally, more complex applications—in particular ones with an infinite number of possible states—may require more synoptic transformation rules. We briefly discuss possible choices in Section 7. Finally, the choice of encoding in this step strongly influences the construction of the distance metric used in the attribute-driven layouts. The example encodings for categorical, Boolean, numerical, and set attributes introduced above give rise to simple distance metrics, as specified in Section 4.3.2.

4.3 Visualization of Application States

The final step of the *Provectories* workflow is the visualization of the paths that users take through the application-state space (see Figure 4 3). The high-dimensional feature vectors are embedded in two dimensions and visualized as a scatterplot using either a *topology-driven layout*, which emphasizes connectivity between identical states, or an *attribute-driven layout*, which focuses on the similarity between states. In both cases, each point represents an application state. The paths of users are visualized as trajectories going through these points. Data attributes or metadata can be mapped to the visual channels of the line and point marks. We first describe the embedding techniques and the required pre-processing of the provenance vectors. We then discuss the visual encoding choices and how meta-analysts can interact with the *Provectories* visualization.

4.3.1 Topology-driven Layout

For the topology-driven layout, we treat the collection of user sessions as a graph. We first determine a set of unique nodes, where each node represents a unique application state. Uniqueness is based on identity of the high-dimensional feature vectors. We treat two nodes as connected if one succeeds the other in any of the user sessions. We lay out the nodes using a force-directed network spatialization algorithm (ForceAtlas2 [19]; for implementation details, see Section 5). The nodes are then connected by drawing a spline trajectory through them for each session (**M1**; see Section 4.3.3).

In the topology-driven layout, a single user session with no duplicate states always results in a linear "chain" of points. Only when states are revisited or shared across multiple sessions, patterns, such as loops and branches, emerge from this layout (**S3**, **M2**).

4.3.2 Attribute-driven Layout

For the attribute-driven layout, we treat the whole collection of application-state vectors across all user sessions as samples from a high-dimensional manifold. We embed these samples based on their similarity (**S3**, **M2**), using various dimensionality reduction techniques. Specifically, we compare the results for MDS, *t*-SNE, and UMAP.

As these dimensionality reduction techniques aim to place similar points close to each other, it is important to define a meaningful metric for calculating the mutual distances between the high-dimensional feature vectors. We

suggest defining this distance metric based on individual distance functions for each attribute type. To this end, we treat each feature vector a as consisting of sub-vectors α_i, and let $k(i)$ denote whether α_i encodes a categorical, Boolean, numerical, or set attribute:

$$a = (\underbrace{\alpha_1, \ldots, \alpha_C}_{k(i)=\text{cat}}, \underbrace{\alpha_{C+1}, \ldots, \alpha_{C+B}}_{k(i)=\text{bool}}, \underbrace{\alpha_{C+B+1}, \ldots, \alpha_{C+B+N}}_{k(i)=\text{num}}, \underbrace{\alpha_{C+B+N+1}, \ldots, \alpha_{C+B+N+S}}_{k(i)=\text{set}}).$$

We then define distance metrics for each α_i depending on $k(i)$. To improve comparability, each attribute distance can be normalized to the interval between 0 and 1.

Cat As discussed earlier, we use a one-hot encoding to represent the categorical attributes. A reasonable distance metric is 0 if the two states have the same categorical value, and 1 otherwise: $d_{\text{cat}}(\alpha, \beta) = 1 - \alpha \cdot \beta$. Here, $\alpha \cdot \beta$ denotes the inner product of α and β.

Bool The same distance function can be used for Boolean attributes if the one-hot encoding was used: $d_{\text{bool}}(\alpha, \beta) = d_{\text{cat}}(\alpha, \beta)$. If the single-number encoding is used instead, the distance function is an exclusive or: $d_{\text{bool}}(\alpha, \beta) = \alpha \oplus \beta = \alpha + \beta \pmod 2$.

Num For numerical attributes, we define the distance as the absolute difference normalized by the total range of the numerical attribute: $d_{num}(\alpha, \beta) = |\alpha - \beta|/(n_{max} - n_{min})$.

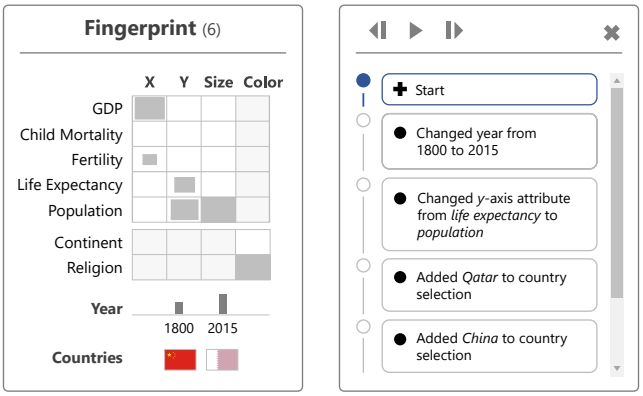
Set For set attributes encoded as described in Section 4.2, we propose the family of distance functions $d_{\text{set}}(\alpha, \beta) = \|\alpha - \beta\|_p / s^{1/p}$, where s is the cardinality of the complete set. For $p = 1$ this amounts to counting the number of different entries in the encoded set vectors and dividing by the cardinality. This entails that the average distance between two random subsets is 0.5. However, for the Gapminder data, the sets that represent country selections typically have only a few elements, which leads to generally smaller distances. To counteract this, we chose $p = 2$, which corresponds to a normalized Euclidean distance between set vectors. For $p = 2$ the average distance between two random subsets is close to 0.7 (for typical values of s) and the average distance between typical country selections is close to 0.5. This choice therefore leads to better comparability with the other metrics.

Total Finally, the total distance between two feature vectors can be calculated as:

$$d(a, b) = \sum_{i=1}^{C+B+N+S} w_i \times d_{k(i)}(\alpha_i, \beta_i). \quad (1)$$

Here, w_i is a weight that can be assigned to the i^{th} attribute. The default weights are $w_i = 1$. The higher the weight of a specific attribute, the greater the likelihood that patterns for the associated data type will prevail in the embedding.

We use this distance as an input to the MDS, t-SNE, and UMAP techniques. Unlike in our previous work [17], we remove duplicate high-dimensional vectors prior to the embedding by default. Otherwise, clusters of identical points can be mistaken for specific data-related patterns. This removal of duplicates takes into account whether any of the weights are set to zero by the user. Zero-weighted attributes are treated as duplicates regardless of their value. We set the perplexity hyperparameter of t-SNE to 50 by default [42], and choose the nearest neighbor parameter for UMAP accordingly (since perplexity can be understood as a smooth measure of the number of nearest neighbors). Details of the implementations used for the embedding are given in Section 5.



(a) Fingerprint (b) History Graph

Fig. 5: (a) *Provectories* fingerprint visualization for multiple selected application states, encoding the relative frequency of attribute values. (b) The history graph view for step-wise analysis of a single session.

4.3.3 Visual Encoding and Interactivity

As stated above, each layout technique results in a scatterplot of embedded application states. We visualize the user sessions as spline trajectories through these points (**S1**, **M1**). We chose this design over more traditional graph-drawing techniques (e.g., tree layout) for three reasons: First, drawing an individual trajectory for each session automatically results in an effective multigraph visualization in which parallel edges are visible as such. Second, each user session has its own distinct path, whose visual channels can be used to encode additional data. Third, in cases in which the meta-analyst decides not to remove duplicates in the attribute-driven layout. The same drawing algorithm can be applied.

Meta-analysts can select the visual encoding of the point and line marks. Point marks can be colored categorically depending on categorical or Boolean values, or using a sequential color scale for numerical values. An age attribute is also available which corresponds to the temporal index of each application state within its session. Lines can be colored categorically by meta-attributes such as usernames and predefined task labels. They can further be switched on and off by their categorical labels and filtered by length by using a range slider. These coloring and filtering options address *Explore Dimensions* and *Explore Items in Enriched Layout* tasks and described by Nonato and Aupetit [28].

To let meta-analysts inspect the underlying high-dimensional data for specific points, the *Provectories* visualization features so-called *fingerprint* visualizations. Upon hovering over a point, the fingerprint of the corresponding single application state is shown. When multiple points are selected (e.g., via a lasso selection), the fingerprint is adapted to encode the distribution of values among the application states. The exact visual encoding of the fingerprint depends

on the number and types of attributes that describe the state of a given application.

Here, we describe the fingerprints designed for the Gapminder example (see Figure 5a). A table lists all categorical and Boolean attributes with their values, where the frequency of values within the selection is encoded by the size of the marker in each cell. A histogram shows the distribution of year values. The distribution of set selections is displayed as a list of country flags, with each flag's opacity encoding the number of states for which that country was part of the selection. This list is ordered by frequency, with the most frequently selected countries appearing first.

To facilitate tracing of individual user sessions even in the presence of potential visual clutter, the history graph with a list of user actions and the resulting application states for an individual session can be activated (see Figure 5b). Selecting states in the history graph overlays the embedding space with arrows that indicate the position and direction of the user session. Analysts can follow the session in a stepwise manner or via an automated animation.

5 Implementation

The *Provectories* workflow is implemented as three individual components which closely correspond to the three steps of the workflow described in Figure 4: 1 a system for tracking the interaction provenance, which must be incorporated into the visual analytics tool that meta-analysts want to study; 2 a module that structures, processes, and exports the recorded provenance data; and 3 the interactive visualization of the user sessions.

For provenance tracking, we use the *KnowledgePearls* implementation of Gapminder [39]. The resulting provenance files are processed in Python. We provide a Python module[1] with classes for application states, user sessions, and collections of sessions which can be adapted to interaction data from different visualizations. The processed provenance data can be exported with or without pre-calculated embedding coordinates. We use the openTSNE implementation of t-SNE [34], the official UMAP Python implementation [26], the scikit-learn MDS implementation [30], and the ForceAtlas2 implementation from the datashader module [1].

To visualize the exported interaction data, we use an improved version of the *ProjectionPathExplorer* tool [17], with online embedding functionality based on tsnejs [20], umap-js [31] and Graphology ForceAtlas2 [32]. All sessions described in this paper can be explored online[2].

6 Results

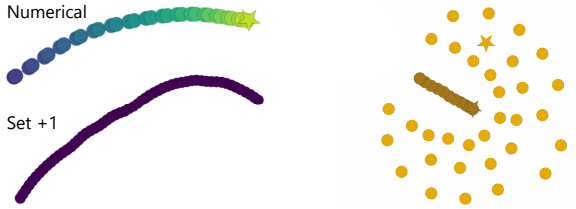
In this section, we describe patterns identified within interaction provenance data from both synthetically generated and real user sessions using the *Gapminder* dataset described earlier. The generated sessions serve to show the visual patterns of simple user actions, whereas the real user sessions demonstrate the utility of *Provectories* in studying actual analysis provenance.

1. Python code: https://github.com/JKU-ICG/sensemakingspace/
2. Prototype: https://sensemaking-paper--projection-path-explorer.netlify.app/

6.1 Meta-Analysis of Synthetic User Sessions

We started by creating synthetic sessions in which only a single data type (e.g., numerical or set) is changed in a strictly defined way. We then continued to construct more complex artificial stories in which two different data types are changed (e.g., selecting several set attributes (countries) before increasing the numerical value (year) by a constant). We also created stories in which some of these value changes were semi-randomized (e.g., increasing the numerical value randomly in every other step).

In total, we created 20 synthetic user sessions. Our rationale was to simulate actual user behavior, which is usually not based on a single data type modification, while still keeping the session simple enough to reveal basic patterns.



(a) t-**SNE**—Top: Numerical value changes; Bottom: Accumulative set selections (b) t-**SNE**—Single set selections

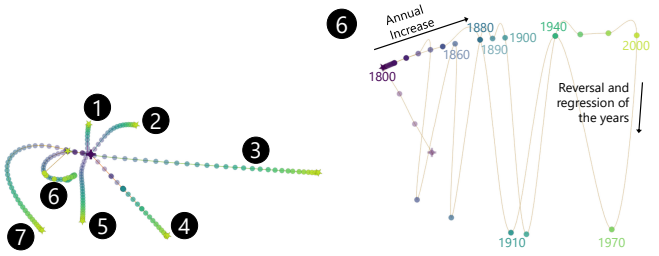
Fig. 6: State patterns identified from synthetically generated single sessions, using t-SNE as an attribute-driven layout. *Gapminder* data showing (a) numerical value changes from **1800** to **2015** (top) and a chain of sets when one item is added at a time to the current set selection (bottom); and (b) **ring layout** based on a single selected and two selected set attributes. The **chain of states** represents a session with numerical changes and no country selected.

From these synthetic sessions, we observe that the numerical data type forms a chain of states with increasing values in the attribute-driven layout (Figure 6a, top). A similar chain can form for the set data type if in each step a single item is added to an existing selection (Figure 6a, bottom).

For the set data type, a second, more frequently observed pattern can arise. If only single set items are selected in each state, all of these states have a mutual distance of $1/\sqrt{n}$, where n represents the total number of countries within the embedding space. This gives rise to circular patterns (see **disconnected dots** in Figure 6b). Here, this set-related pattern formed around a central state, which is the final state of a numerical session (**chain of states** in Figure 6b).

By definition, the topology-driven layout does not distinguish between data types. As long as two states in the dataset are not identical, a linear "chain" of unique points will be drawn. Figure 7a represents seven synthetically generated sessions visualized by a topology-driven layout. The different sessions include ❶ numerical changes only, ❷ categorical changes only, and ❸ alternating set selections. All sessions start at a central point of the embedding (✚) and are directed outwards (★), but they cannot be distinguished from each other.

Figure 7b shows an attribute-driven layout of a numerical session, session ❻ from Figure 7a, with MDS (with increased weight for the numerical data type). It reveals effectively the value changes throughout the session. In particular, the position on one of the axes can be directly related to the years selected in the *Gapminder* tool. Thus,



(a) **Force-Directed** layout (b) **MDS** layout

Fig. 7: (a) Seven synthetic stories using *Gapminder* data visualized by a force-directed layout, where data type sessions are displayed independently in the embedding space. (b) Session ❻ visualized by an MDS layout, where numerical value changes are highlighted with a higher weight ($w_{\text{num}} = 10$).

in this simple case, one of the axes in the MDS plot can be interpreted as a temporal axis (in this case the year). Further, a loop structure reveals that the story, after reaching the year 2000, moves back to 1970 and then to 1940. Since this reversal of direction is based on similar—but not identical—states, it is only visible in the attribute-driven and not in the topology-driven layout.

Overall, we confirmed that no data-type-specific patterns are visible in the topology-driven layout, while we were able to extract patterns for Boolean, categorical, numerical, and set attributes from the attribute-driven layout (see Figure 8).

Bool *Boolean* attributes are distinctly separated within the embedding space. By inclusion of color encoding, two areas for **religion** and **continent** become visible in Figure 8a. These separate areas in the embedding space are driven by the dissimilarity of the feature vector. This holds true for all attribute-driven layouts and for single or multiple sessions in one projection.

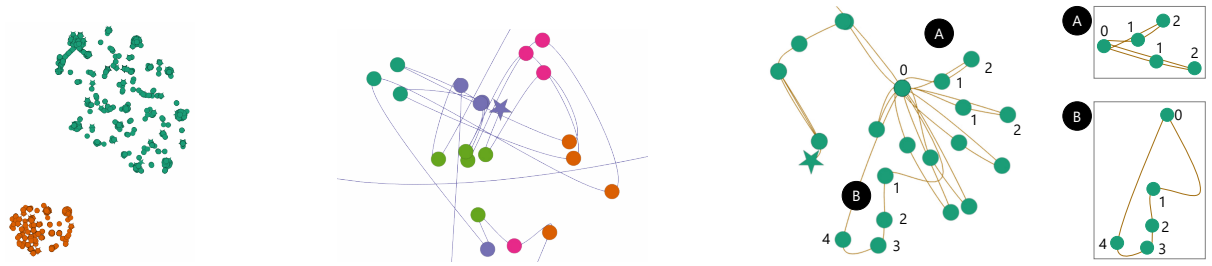
Cat Like Boolean changes, *categorical* attribute changes can cause formation of clusters for each category in the embedding. Furthermore, certain trajectory patterns can reveal categorical changes. In Figure 8b, for instance, a cluster with the same value for the size attribute **population** is shown. Within this cluster, categorical changes in another attribute (here, x-axis) lead to substructures (**fertility rate**, **child mortality**, or **GDP**) that are connected by crossing, zigzagging lines. This phenomenon can be more pronounced, giving rise to hierarchical clustering, as explained below.

Num As shown in the examples in Figure 1❷ and Figure 6a, changes in *numerical* attributes lead to a chain of states in ascending or descending order. The states need not be traversed by the user explicitly in this sequential order. These states will automatically form a chain based on the definition of the numerical distance. This chain pattern is consistent for all three attribute-driven approaches (t-SNE, MDS, and UMAP).

Set t, which attempts to preserve high-dimensional distances, gives circular arrangements for single selected set attributes. Furthermore, a combination of single and multiple selected countries leads to a ring pattern, as outlined in Figure 8c. Here, Ⓐ represents (1) a single selected set attribute as inner ring, and (2) a second, added country as the outer ring, before (0) both countries are deselected again. This ring structure arises from a distance of $\frac{1}{\sqrt{n}}$ between states with different single-country selections and a distance of $\frac{2}{\sqrt{n}}$ between states with two different countries selected. The same reasoning can be applied to the formation of chains when countries are selected only in an accumulative way (as shown in Figure 6a). Such chains can also be visible within the ring arrangements (Ⓑ in Figure 8c)

Weighting. All patterns described so far were identified for equally weighted attributes (i.e., $w_i = 1$ for all i in Eq. 1). If the weight for a particular attribute of type T is increased (e.g., $w_i = 10$ for some i with $k(i) = T$), the patterns related to that data type become more dominant in the attribute-driven layout. For instance, increasing the weight for a numerical attribute forces more states to be placed along a shared axis representing that numerical attribute. In Figure 9, states are colored by their year value, ranging from **1800** to 2015. For the default weighting (Figure 9a), two chains move outwards from the center in different directions, and some "outliers" with low year values are visible in the central region of the embedding. For the increased weight (9b, the two chains are parallel and no outliers are visible.

Hierarchical clustering. The weighting can be adjusted to focus on a subset of data types while reducing or completely removing the effect of the other types. For the attribute-driven layout shown in Figure 10, the weight of numerical and set attributes was reduced to zero ($w_i = 0$ for all i with $k(i) = \text{num}$ or $k(i) = \text{set}$). This gives rise to a hierarchical clustering based on the remaining Boolean



(a) t-**SNE**—Boolean distinction over multiple sessions (b) t-**SNE**—Categorical changes in multiple sessions (c) t-**SNE**—Single and multiple set selections within a single session

Fig. 8: Patterns identified from synthetically generated single and multiple sessions using an attribute-driven layout. *Gapminder* data showing (a) a Boolean distinction between **religion** and **continent** within the embedding space; (b) categorical changes (● ● ● ● ● the colors show clusters for x-axis attributes); and (c) single and multiple set selections, where the number of selected states is indicated (0–4).

(a) *t*-**SNE**—same weight for all data types ($\forall i$: $w_i = 1$).

(b) *t*-**SNE**—increased numerical weight ($w_{num} = 5$).

Fig. 9: Effect of weight changes with respect to the numeric data type using *t*-SNE. (a) Shows synthetic stories with a uniform weight ($w_i = 1$ for all i) on all data types, and (b) the same session with an increased weight on the numerical data type ($w_{num} = 5$). The numerical attribute ranges from **1800** to **2015**.

and categorical attribute values. Figure 10 1 shows a clear separation of the application states based on whether the color attribute represents **continent** or **religion**; 2 within the **religion** cluster, further division is determined by the size attribute; 3 within each cluster of equal-size value, the attribute mapped to the x-axis causes a further sub-clustering. The values of these attributes are shown in the fingerprints in the lower part of Figure 10.

6.2 Meta-Analysis of Real User Sessions

For the analysis of real user sessions, we conducted a user study with 32 participants (m: 17, f: 15). The participants were students of a Data Science master program, as part of which they attended an introductory course on data visualization. We asked participants to find answers to the four tasks listed in Table 2 by using the *Gapminder* tool.

We designed T1 and T2 as directed tasks, where the answers could be identified within a small number of interactions. In contrast, T3 and T4 are exploratory, open-ended tasks, which typically lead to longer sequences. We asked the participants to save each task in a file so the starting point for each session could be identified. We removed the sessions that contained all tasks in one file, which reduced the total number of sessions to 109.

6.2.1 Data Types

By means of the real user sessions, we were able to confirm the patterns observed in the synthetic stories for all four data types. As expected, T1 did not reveal any data-type-specific patterns because accomplishing the task required only the year to be changed once. Since task T2 evoked a Boolean change, sessions related to that task cluster in the embedding space, see Figure 11b. Moreover, categorical changes become apparent for both single and multiple sessions. The open-ended task T3 covered categorical changes, numerical variations, and set alternations, where most participants set the target attribute to be *population* on the x-axis or mapped it to the size. As anticipated, T4 consisted mainly of categorical changes, where participants explored the data point distribution from the Gapminder scatterplot for almost all possible attribute combinations and selected single countries at the end of the sessions. Interestingly, participants often changed the size encoding to detect differences. Overall, the data type observations in all four tasks match the patterns from the synthetically generated stories.

6.2.2 Analytical Strategies

T1 was answered correctly by 78.0% of the participants using an average of 17.52 steps. Based on the large variance (14.01) in the number of steps, it can be seen in both layouts that most participants had already found the answers to both questions after an average of 4 steps, but continued to explore the data and the tool by using the slider for the numerical attribute or the drop-down menu for categorical changes. As these additional steps are not necessary to complete the task, we call this process a *random walk*. This may be traced back to our study design, as we performed no explicit onboarding with the tool. Figure 11b shows multiple superimposed trajectories pointing to the same cluster within the embedding space for this task, which means that most participants chose the same approach to accomplishing the task. States visualized by the attribute-driven layout distinctly show two small clusters (★) within the embedding for both answers. In contrast, in the topology-driven layout, no unique positions for the answers can be identified. This can be attributed to the higher number of nonidentical states (e.g., attribute on x was placed on the y-axis). Thus, answers that are relatively close to the actual answer point to the outer region of the embedding if no other user selected the same application state (see Figure 11a). Moreover, such sessions cannot be distinguished from random-walk analysis strategies.

For T2, half of the participants started by changing the year, whereas the other half began by changing the Boolean attribute first. Particularly noticeable are the variations in categorical attributes. Participants confirmed the country selection several times by changing the assignment of the target attribute to different categorical positions (e.g., x-axis, y-axis, mark size), see zigzag pattern in Figure 11b. About one third of the participants (35.38%) completed the task by identifying both answers (country with the highest child mortality and fertility rate) in the same application

TABLE 2
Overview of the four tasks from the user experiment with average answer correctness and average number of steps taken to accomplish a task.

Task	Question	∅ Answer Correctness	∅ Number of Steps
T1	In 2015, select (a) the country with the highest GDP, and (b) the country with the largest population.	78.0 % ± 0.3 %	17.5 ± 14.0
T2	In 1843, select the Muslim country that has (a) the highest child mortality rate (b) the highest fertility rate.	87.0 % ± 0.3 %	18.7 ± 13.7
T3	Select the European country that had the largest relative drop in population between 1939 and 1945.	40.6 % ± 0.5 %	27.7 ± 23.8
T4	Select any country on the continent that has the highest correlation between any two attributes in 1945.	0.3 % ± 0.1 %	29.8 ± 34.0

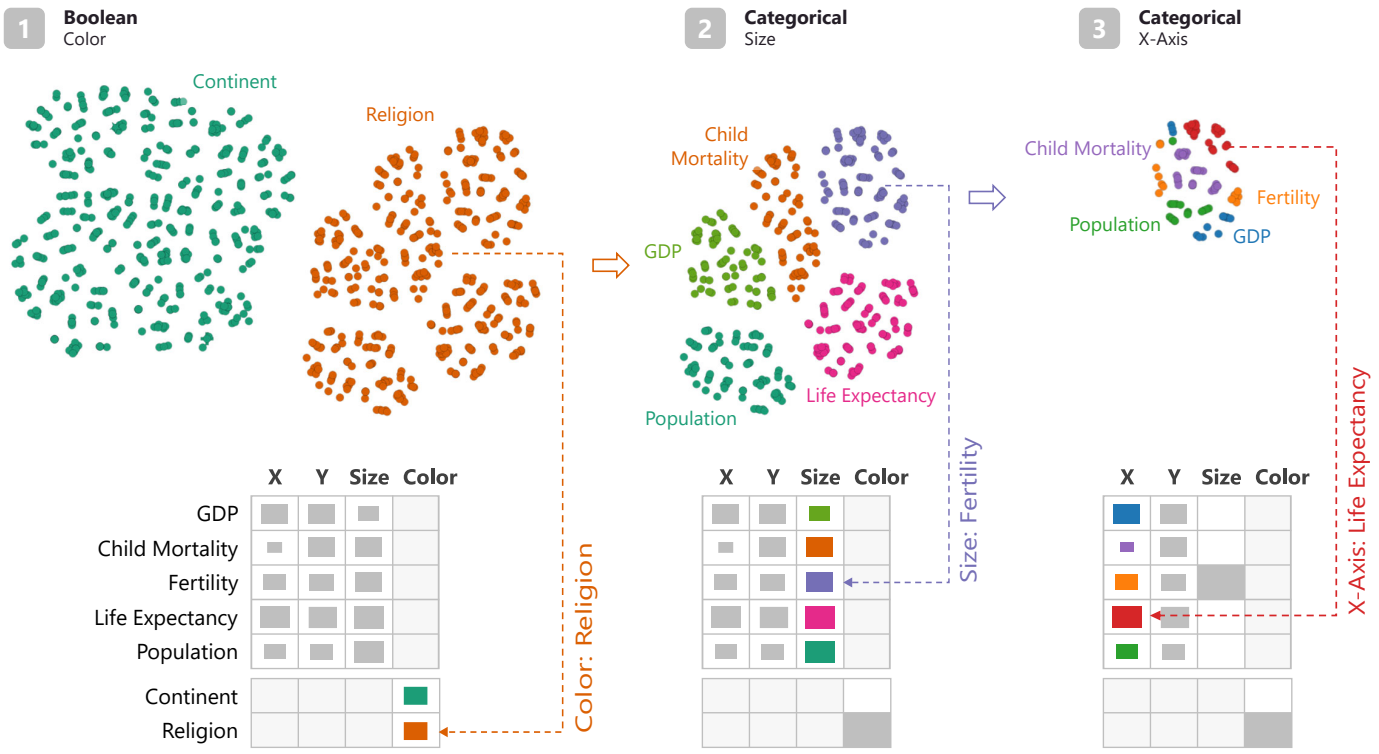


Fig. 10: Hierarchical clustering emerges in the following order: 1 Boolean, 2 categorical, 3 categorical after removing the effect of numerical and set attributes by setting their weights to zero ($w_{num,set} = 5$). The simplified fingerprint shows the relative frequencies of the Boolean and categorical attributes in the corresponding attribute color.

state with both (T2a) *child mortality* and (T2b) *fertility* as categorical axis options.

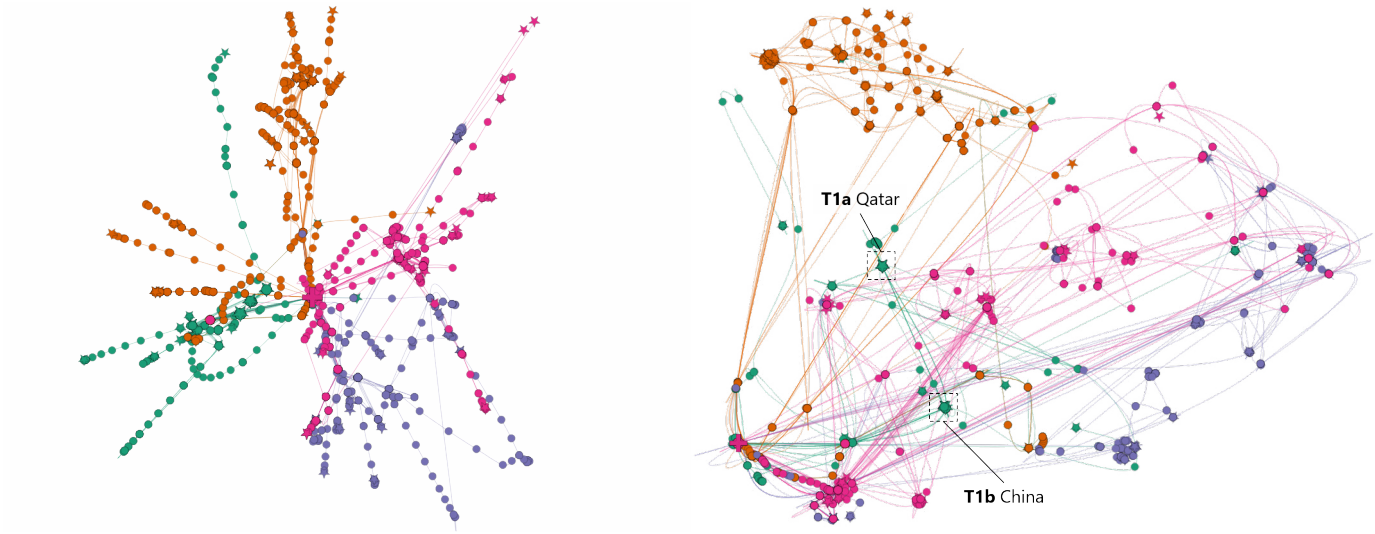
For real-world scenarios, multiple exploration paths lead to the correct answer. For T3, 18 out of 22 sessions first converge on one unique application state before continuing the analysis in various ways. Participants started by changing the year to 1940 before selecting different countries and varying categorical characteristics to determine the largest relative drop in the population between 1939 and 1945. One user, for instance, started (✚) by assessing the correlation between two attributes by ❶ selecting all y-axis attributes (except for child mortality), and then revisited the initial attribute. This sequence emerges as a visual loop in Figure 12. The user continued by changing the year and ❷ alternating between two set attributes. Toggling compares the two alternatives and verifies the final selection. Before terminating the analysis, the participant ❸ looked at four other countries and confirmed the initial selection (★). T3 benefits from visualizing the interaction provenance in the attribute-driven layout because participants selected different countries after changing the year to 1940, and the selected states are positioned close to each other. Hence, the topology-driven layout treats these states as independent and unique (see Figure 12b), and the attribute-driven approach emphasizes the similarity of the application states for different analysis processes (see Figure 12a).

The last exploration task T4 shows identical states and overlapping trajectories within the embedding with T3 because the same year—1945—was selected. In general, both open-ended tasks cover a large area within the embedding space. Furthermore, T4 shows an average response accuracy of 0%, while the number of steps to accomplish the task is higher than for the directed tasks (T1 and T2) (see Table 2). Overall, participants tried to find the highest correlation between any two attributes by varying all attribute combinations for any categorical combinations for T4. Furthermore, the attribute-driven layout for T4 shows a zigzag pattern, which means that x- and size attributes are contained within clusters of y-attributes, which confirms the hierarchical dependency (see Section 6.2).

6.3 Layout Applicability

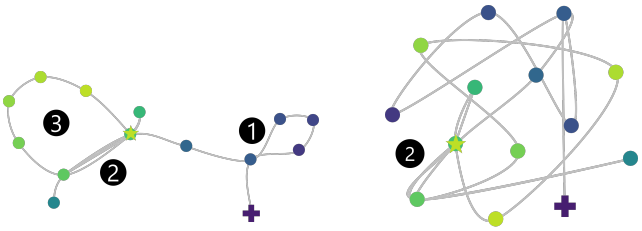
Based on the insights gained from synthetically generated sessions and actual user-interaction provenance, we summarize the identifiable patterns in Table 3. Depending on the layout and the visual pattern, we introduce an indicator that describes the readability and validity of each pattern. The former indicates whether it is possible to identify this pattern within the embedding space, and the latter signifies how reliably the pattern has been identified.

Data-type-specific patterns emerge only in an attribute-driven layout. Each Boolean item occupies its own area within the embedding space, which leads to two distinct areas for all three techniques (t-SNE, MDS, and UMAP); this can be accentuated by putting a higher weight on the data type. When only a single session is embedded by itself, categorical changes are difficult to extract due to the low number of states within the embedding space. In contrast, when multiple sessions are embedded at the same time, additional states provide enough context that clusters for each attribute can emerge. However, based on the high number of trajectories, visual clutter can result. In these



(a) **Force-Directed** layout (b) t**-SNE** layout

Fig. 11: All four tasks of the user experiment within the same embedding space. T1, T2, T3, and T4 displayed in (a) a topology-driven layout based on a force-directed embedding, and (b) an attribute-driven layout based on t-SNE.



(a) **Force-Directed** layout (b) t**-SNE** layout

Fig. 12: T3, where ❶ shows alternation between y-axis items, ❷ toggling between two states and ❸ a verification loop by screening for incorrect states

cases, highlighting and tracing single sessions supports the identification of categorical changes. In contrast, a chain of states emerging from numerical value changes has a high validity within all three algorithms, but also cumulatively selected set items resemble this behavior. In addition, single selected set attributes form a circular state pattern for t-SNE. MDS and UMAP do not yield a clear pattern, since many data points converge to almost a single position in the embedding space. Although t-SNE is known for preserving local structure better, while UMAP is said to preserve global structures better, the attribution of set patterns was the only clear difference we could identify between the two approaches. We also observed—as expected—that with increasing perplexity values the t-SNE scatterplots tended to resemble those constructed with UMAP.

To trace users' analysis steps, we analyzed single user sessions embedded individually or together with other sessions, and obtained similar results. For both topology-driven and attribute-driven layouts, steps revisited in single and multiple sessions can be recognized based on the removal of duplicates (and, similarly, for loops containing intermediate states). Near identical data points can only be identified in an attribute-driven layout, where they are positioned closer to each other. In MDS and UMAP, however, data points of set attributes almost overlap in the embedding space, whereas the chain pattern of numerical values results in a small distance between similar data points. For UMAP, this may be improved by choosing a different setting of the "mindist" parameter.

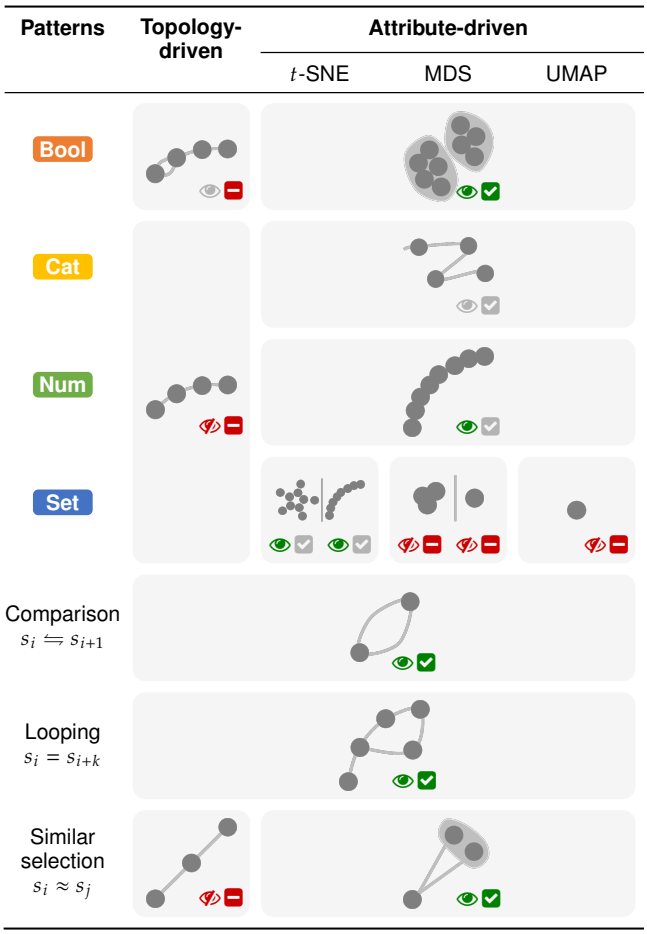
Single sessions serve to understand a user's analytical behavior. Confirmation and verification tasks are displayed as loops for one session, whereas overlapping trajectories and nearby data points for topology-driven layouts signify application states that were also visited by other users.

Due to overlapping and intersecting of trajectories, identification of an analytical reasoning process for a single session becomes more difficult with an increasing number of sessions. We address this shortcoming with the history graph (see Figure 5b), which allows meta-analysis to detect and understand patterns of single user sessions in multiple simultaneously displayed sessions by highlighting the session of interest. As a result, patterns recognized in single sessions cannot always be applied to multiple sessions. To identify whether multiple users tackled a task in a similar manner, we refer to an attribute-driven layout that is characterized by the similarity of the application states. Trajectories pointing in the same direction within the embedding space and from one cluster to another indicate similar approaches by multiple users. In accordance with attribute-driven layouts, individual analysis steps or steps of a random walk represent unique data points. Particularly for MDS and UMAP, Boolean, and categorical changes evoke very visually distant data points. Based on the entropy of the embedding space, individual data points or even sessions become distinctive.

7 Discussion

Complexity and Length of Provenance Vectors: We have demonstrated our approach using Gapminder, which offers a limited set of functionalities, and can be fully described with a feature vector of the length $5+5+5+1+2+n =$

TABLE 3
Visual and data patterns extracted from various techniques, with indicators for the levels of readability (low, medium, high) and validity (low, medium, high).



$18 + n$, where n is the total number of countries included in the projected sessions (see detailed composition in Section 4.2). For more feature-rich applications, such as Tableau or Microsoft Power BI, a feature vector that describes the application state uniquely would be extremely long. If users can create unlimited visual components themselves (e.g., new views in dashboards), such a feature vector may even become impossible to construct. Choosing a suitable subset of descriptors to describe such application states can be challenging, as the selection directly impacts the meta-analysis. Selecting too many features can be detrimental to the distance metric to such an extent that no meaningful patterns appear. One possible approach is to introduce an abstraction layer such that the application states are described only approximately in a simplified configuration space. For tools with an arbitrary number of views, a representation similar to Fock states in quantum mechanics [12] could be used to describe the elements in this configuration space. In such a representation, instead of listing all views with their attributes, the possible attribute combinations are listed along with associated "counts" of views that share those attributes. As part of future work, we plan to apply the *Provectories* approach to more complex visual analysis tools, starting with an analysis of tools using different visualization types (e.g., an interactive parallel coordinates plot) before moving on to more complex tools that support a rich set of interactions and use multiple-coordinated views.

Dual-View Meta-Analysis: As described in Section 6, the combination of either a topology or an attribute-driven layout with the sequential history graph helped us to understand analytical reasoning processes of a single user without losing the context of relative position of application states arising from other user sessions. In order to increase the comprehensiveness of pattern recognition through the respective techniques, we plan to show both layouts simultaneously in a multiple-coordinated view.

Motifs: With our novel visual analysis approach, we extracted patterns based on the connectivity and similarity of application states. To increase the knowledge about analysis sequences and to reduce visual complexity, we suggest using a motif-based aggregation for both layout approaches [40]. Detection of motifs allows us to aggregate the provenance graph or parts thereof while preserving the high-level structure. This adds the potential of chunking interaction sequences for a more compact display. Furthermore, identified patterns could be rendered as a sequence of actions and compared across multiple sessions.

Hybrid Layout Approach: In order to combine the advantages of both the topology *and* the attribute-driven layout, we plan to develop a hybrid layout approach. In the purely attribute-driven layout used in our work, the distance matrix for t-SNE or UMAP is calculated directly from the attribute values (see Section 4.2), while the topology-driven layout is based on the connectivity of states. A hybrid approach could build on *tsNET* [22], which creates a topology-driven layout by transforming the adjacency matrix of a graph to a distance matrix which is then used for t-SNE. Such a topology-driven distance matrix could be combined with an attribute-driven one and subsequently used for a hybrid embedding. It may thus be possible to preserve local similarity and topological patterns in a single *Provectories* visualization.

8 Conclusion

In this paper, we have presented a novel visual analysis approach to extracting patterns from interaction provenance data. Our *Provectories* approach consists of three steps: (1) the acquisition of interaction provenance data in the form of logged application states, (2) the construction of feature vectors representing these states, and (3) the visualization of provenance using topology- and attribute-driven layouts. By interactively exploring such visualizations for synthetic stories and actual user sessions, patterns based on data types and analytical reasoning processes can be revealed. However, interaction provenance from other applications and variation of the logged application states remain to be explored. We strongly believe that *Provectories* can fill a key in the field of provenance and sense-making to improve understanding of similarities between analysis processes and user-specific behaviors.

REFERENCES

[1] Anaconda Inc. Datashader. Accurately render even the largest data, 2016. https://datashader.org/.

[2] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):559–568, 2016.

[3] T. Blascheck, F. Beck, S. Baltes, T. Ertl, and D. Weiskopf. Visual analysis and coding of data-rich user behavior. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 141–150. IEEE, 2016. doi: 10.1109/VAST.2016.7883520

[4] E. T. Brown, A. Ottley, H. Zhao, Q. Lin, R. Souvenir, A. Endert, and R. Chang. Finding Waldo: Learning about Users from their Interactions. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1663–1672, 2014. doi: 10.1109/TVCG.2014.2346575

[5] E. T. Brown, S. Yarlagadda, K. A. Cook, A. Endert, and R. Chang. ModelSpace: Visualizing the Trails of Data Models in Visual Analytics Systems. In *MLUI 2018: Machine Learning from User Interactions for Visualization and Analytics*, p. 11. IEEE, 2018.

[6] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva, and H. T. Vo. VisTrails: visualization meets data management. In *Proceedings of the international conference on Management of data (SIGMOD '06)*, p. 745. ACM Press, 2006. doi: 10.1145/1142473.1142574

[7] H. Chung, S. Yang, N. Massjouni, C. Andrews, R. Kanna, and C. North. VizCept: Supporting synchronous collaboration for constructing visualizations in intelligence analysis. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pp. 107–114. IEEE, 2010. doi: 10.1109/VAST.2010.5652932

[8] P. Cowley, L. Nowell, and J. Scholtz. Glass Box: An Instrumented Infrastructure for Supporting Human Interaction with Information. In *Proceedings of the Conference on System Sciences (HICSS '05)*, p. 296c, 2005. doi: 10.1109/HICSS.2005.286

[9] W. Dou, D. H. Jeong, F. Stukes, W. Ribarsky, H. R. Lipford, and R. Chang. Recovering reasoning process from user interactions. *IEEE Computer Graphics & Applications*, 29(3):52–61, 2009.

[10] C. Dunne, N. Henry Riche, B. Lee, R. Metoyer, and G. Robertson. GraphTrail: Analyzing Large Multivariate, Heterogeneous Networks While Supporting Exploration History. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*, pp. 1663–1672. ACM, 2012. doi: 10.1145/2207676.2208293

[11] M. Feng, E. Peck, and L. Harrison. Patterns and Pace: Quantifying Diverse Exploration Behavior with Visualizations on the Web. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):501–511, 2019. doi: 10.1109/TVCG.2018.2865117

[12] V. A. Fock. Konfigurationsraum und zweite Quantelung. *Zeitschrift für Physik*, 75(9–10):622–647, 1932. doi: 10.1007/BF01344458

[13] M. Gleicher. Considerations for Visualizing Comparison. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):413–423, 2018. doi: 10.1109/TVCG.2017.2744199

[14] S. Gomez and D. Laidlaw. Modeling task performance for a crowd of users from interaction histories. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '12)*, p. 2465. ACM Press, 2012. doi: 10.1145/2207676.2208412

[15] S. Gratzl, A. Lex, N. Gehlenborg, N. Cosgrove, and M. Streit. From Visual Exploration to Storytelling and Back Again. *Computer Graphics Forum*, 35(3):491–500, 2016. doi: 10.1111/cgf.12925

[16] J. Heer, J. Mackinlay, C. Stolte, and M. Agrawala. Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '08)*, 14(6):1189–1196, 2008. doi: 10.1109/TVCG.2008.137

[17] A. Hinterreiter, C. Steinparz, M. Schöfl, H. Stitz, and M. Streit. Exploring Visual Patterns in Projected Human and Machine Decision-Making Paths. *arXiv:2001.08372 [cs]*, Jan. 2020.

[18] J. Hollan, E. Hutchins, and D. Kirsh. Distributed cognition: toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction*, 7(2):174–196, 2000. doi: 10.1145/353485.353487

[19] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian. ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. *PLoS ONE*, 9(6):e98679, 2014. doi: 10.1371/journal.pone.0098679

[20] A. Karpathy. tSNEJS, 2014. https://github.com/karpathy/tsnejs/.

[21] N. Kerracher, J. Kennedy, and K. Chalmers. A Task Taxonomy for Temporal Graph Visualisation. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2015. doi: 10.1109/TVCG.2015.2424889

[22] J. F. Kruiger, P. E. Rauber, R. M. Martins, A. Kerren, S. Kobourov, and A. C. Telea. Graph Layouts by t-SNE. *Computer Graphics Forum*, 36(3):283–294, 2017. doi: 10.1111/cgf.13187

[23] Z. Liu and J. Stasko. Mental Models, Visual Reasoning and Interaction in Information Visualization: A Top-down Perspective. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):999–1008, 2010. doi: 10.1109/TVCG.2010.177

[24] W. C. Mankowski, P. Bogunovich, A. Shokoufandeh, and D. D. Salvucci. Finding canonical behaviors in user protocols. In *Proceedings of the conference on Human factors in computing systems (CHI'09)*, p. 1323. ACM Press, 2009. doi: 10.1145/1518701.1518900

[25] Margit Pohl and Johanna Doppler Haider. Sense-making Strategies for the Interpretation of Visualizations—Bridging the Gap between Theory and Empirical Research. *Multimodal Technologies and Interaction*, 1(3):16, 2017. doi: 10.3390/mti1030016

[26] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426 [cs, stat]*, 2018.

[27] P. H. Nguyen, K. Xu, A. Wheat, B. L. W. Wong, S. Attfield, and B. Fields. SensePath: Understanding the Sensemaking Process Through Analytic Provenance. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):41–50, 2016. doi: 10.1109/TVCG.2015.2467611

[28] L. G. Nonato and M. Aupetit. Multidimensional Projection for Visual Analytics: Linking Techniques with Distortions, Tasks, and Layout Enrichment. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2650–2673, 2019. doi: 10.1109/TVCG.2018.2846735

[29] A. Pandey, U. Syeda, and M. Borkin. Towards Identification and Mitigation of Task-Based Challenges in Comparative Visualization Studies. *OSF Preprints*, 2020. doi: 10.31219/osf.io/5p73v

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[31] People+AI Research (PAIR) Initiative. UMAP-JS, 2019. https://github.com/PAIR-code/umap-js/.

[32] G. Plique. Graphology ForceAtlas2, 2016. https://github.com/graphology/graphology-layout-forceatlas2.

[33] M. Pohl, M. Smuc, and E. Mayr. The User Puzzle - Explaining the Interaction with Visual Analytics Systems. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2908–2916, 2012. doi: 10.1109/TVCG.2012.273

[34] P. G. Poličar, M. Stražar, and B. Zupan. openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding. *bioRxiv*, 2019. doi: 10.1101/731877

[35] E. Ragan, A. Endert, J. Sanyal, and J. Chen. Characterizing Provenance in Visualization and Data Analysis: An Organizational Framework of Provenance Types and Purposes. *IEEE Transactions on Visualization and Computer Graphics (VAST '15)*, 22(1):31–40, 2016. doi: 10.1109/TVCG.2015.2467551

[36] P. E. Rauber, S. G. Fadel, A. X. Falcão, and A. C. Telea. Visualizing the Hidden Activity of Artificial Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):101–110, 2017. doi: 10.1109/TVCG.2016.2598838

[37] H. Rosling and Z. Zhang. Health advocacy with Gapminder animated statistics. *Journal of Epidemiology and Global Health*, 1(1):11, 2011. doi: 10.1016/j.jegh.2011.07.001

[38] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang. Eviza: A Natural Language Interface for Visual Analysis. In *Proceedings of the Symposium on User Interface Software and Technology*, pp. 365–377. ACM, 2016. doi: 10.1145/2984511.2984588

[39] H. Stitz, S. Gratzl, H. Piringer, T. Zichner, and M. Streit. KnowledgePearls: Provenance-Based Visualization Retrieval. *IEEE Transactions on Visualization and Computer Graphics (VAST '18)*, 25(1):120–130, 2019. doi: 10.1109/TVCG.2018.2865024

[40] H. Stitz, S. Luger, M. Streit, and N. Gehlenborg. AVOCADO: Visualization of Workflow–Derived Data Provenance for Reproducible Biomedical Research. *Computer Graphics Forum*, 35(3):481–490, 2016. doi: 10.1111/cgf.12924

[41] S. van den Elzen, D. Holten, J. Blaas, and J. van Wijk. Reducing Snapshots to Points: A Visual Analytics Approach to Dynamic

Network Exploration. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):1–10, 2016. doi: 10.1109/TVCG.2015.2468078

[42] L. van der Maaten and G. Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[43] J. Wei, Z. Shen, N. Sundaresan, and K.-L. Ma. Visual cluster exploration of web clickstream data. In *IEEE conference on Visual Analytics Science and Technology (VAST'12)*, pp. 3–12. IEEE, 2012. doi: 10.1109/VAST.2012.6400494

[44] J. Wood, A. Kachkaev, and J. Dykes. Design Exposition with Literate Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):759–768, 2019. doi: 10.1109/TVCG.2018.2864836

[45] K. Xu, A. Ottley, C. Walchshofer, M. Streit, R. Chang, and J. Wenskovitch. Survey on the Analysis of User Interactions and Visualization Provenance. *Computer Graphics Forum*, 39(3):757–783, 2020. doi: 10.1111/cgf.14035

[46] Yang Chen, Jing Yang, and W. Ribarsky. Toward effective insight management in visual analytics systems. In *2009 IEEE Pacific Visualization Symposium*, pp. 49–56. IEEE, Beijing, China, 2009. doi: 10.1109/PACIFICVIS.2009.4906837

[47] J. Zhao, M. Glueck, P. Isenberg, F. Chevalier, and A. Khan. Supporting Handoff in Asynchronous Collaborative Sensemaking Using Knowledge-Transfer Graphs. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):340–350, 2018. doi: 10.1109/TVCG.2017.2745279