A Theoretical Approach for Structuring and Analysing Knowledge Provenance for Visual Analytics

L. Christino[1,2] ⬤, S. Rezaeipour[1] ⬤, E. Milios[1] ⬤ and F. Paulovich[2] ⬤

[1]Dalhousie University, Canada
[2]Eindhoven University of Technology (TU/e), Netherlands
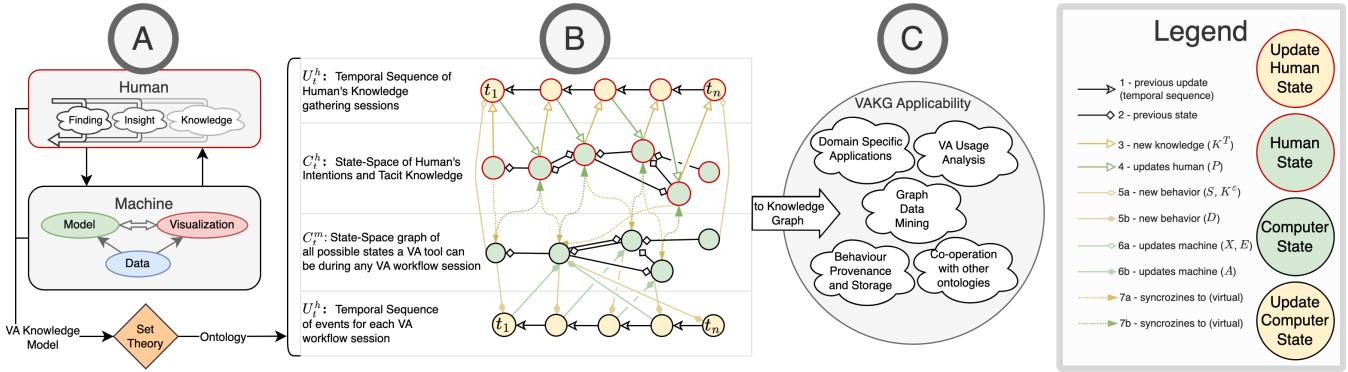
Figure 1: *VAKG unfolds the interactions within the current knowledge model (A) into a temporal knowledge graph (B), which is structured as a 4-way graph containing two temporal (green) and two static (yellow) knowledge graphs. By using VAKG, one can structure and store the user's knowledge-gathering process and all related interactions for eventual analysis (C).*

Abstract

The primary goal of Visual Analytics (VA) is to enable user-guided knowledge generation. Theoretical VA works to explain how the different aspects of a VA tool bring forth new insights through user interactivity, which itself can be captured through tracking methods for reproduction or evaluation. However, the process of automatically capturing the user's thought process, such as intent and insights, and associating it with user's interaction events are largely ignored. Also, two forms of interactivity capture are typically ambiguous and intermixed: the temporal aspect, which indicates sequences of events, and the atemporal aspect, which explains the workflow as sequences of states within a state-space. In this work, we propose Visual Analytics Knowledge Graph (VAKG), a conceptual framework that brings VA modeling theory to practice through a novel Set-Theory formalization of knowledge modeling. By extracting such a model from a VA tool, VAKG structures a 4-way temporal knowledge graph that describes user behavior and its associated knowledge gain process. Such knowledge graphs can be populated manually or automatically during user analysis sessions, which can then be analyzed using graph analysis methods. VAKG is demonstrated by modeling and collecting Tableau and visual text-mining workflows, where comparative user satisfaction, tool efficacy, and overall workflow shortcomings can be extracted from the knowledge graph.

CCS Concepts
*• **Human-centered computing** → **Visual analytics;** • **Computing methodologies** → Knowledge representation and reasoning; Temporal reasoning; Ontology engineering;*

1. Introduction

Visual Analytics (VA) tools allow users to harness insights and knowledge from datasets [SSS*14]. By tracking this insight-generation process with provenance methods, like screen and mouse-click recording, researchers and industry alike can better understand the relationship between their tools and their users. The theoretical

foundation of VA of Sacha et al. [SSS*14] proposes a "knowledge generation model" which not only discusses how the human and the machine interact during an insight-generation process but also discusses what elements within these interactions relate to the various data collected in provenance methods. Among existing works, Federico et al. [FWR*17] use the theory of Sacha et al. [SSS*14] to propose VA as a workflow of *events*, which can potentially be associated with provenance concepts. Other works which propose novel VA systems also use the knowledge modeling theory to describe the process where their users gather knowledge, providing them with a basic best-practice design guideline of how to model [SSS*14, FWR*17], structure [CGJ*17, SKKC18] and understand user behaviour [vLFB*14, MHK*19, HMSA08, XOW*20].

Structuring VA workflows has been a hot research topic [SKKC18, CGJ*17], attracting enormous interest in tracking and analyzing user behavior to understand how knowledge is generated [XOW*20]. In addition to investigating the role of the VA workflow during analysis, such works investigate how the users' pre-existing knowledge [HMSA08, BH19] influence their experience during VA tasks. However, behavioral analytics do not yet use *provenance* in light of the existing knowledge models or ontologies. Instead, each research endeavor develops its own method to acquire, structure, and analyze the users' knowledge-gathering process. Therefore, although Sacha et al. [SSS*14] describes the conceptual relationship between *Machine* and *Human*, some essential aspects are overlooked in behavior analysis. For instance, given that a VA tool follows the workflow of a specific VA model, how would one use this model as the means to acquire, structure, and store ongoing user interactions, or namely *behavioral provenance*? Or how might the recorded data be analyzed to investigate and compare the knowledge gathered among several users, or namely *knowledge provenance*? And how can a single dataset be defined which relates the user's behavior and the gathered knowledge where one can discover which sequences of actions lead to a new insight or which insights were attained due to using a specific, perhaps new, visualization? Or even how to use the answers to these questions for other downstream tasks, such as aiding the development of new tools or comparing different VA tools?

To address these open questions, we propose *Visual Analytics Knowledge Graph (VAKG)*, a novel conceptual framework that proposes a formalized process to extract the underlying VA model of a VA tool, to design a knowledge graph ontology following the model, to define the data to be collected from the user behavior and knowledge gathering which fits said ontology, to populate a knowledge graph containing behavior and knowledge provenance data, and finally to use said knowledge graph for analysis of the relationship of behavior and knowledge. For this, we use existing VA knowledge models [SSS*14, FWR*17, VW05] and reinterpret them as *sets of information* and the process of how these sets interact. This way, VA is separated by its temporal aspect (e.g., temporal sequences of events versus atemporal *state-spaces*) and ownership aspect (e.g., *Human* versus *Machine*). We then define a novel multi-layer knowledge graph structure that follows the *sets of information* and their relationships.

Our main contributions can be summarized as follows:

- A reinterpretation of VA's knowledge model through *Set-Theory* and the relationship between the modeled sets;
- A domain-agnostic knowledge graph structure definition based on VA's knowledge model; and
- A novel usage of a multi-layered Temporal Knowledge Graph architecture as a storage, analysis, and visualization mechanism of VA workflows for understanding the relationship between user behavior and knowledge acquisition.

Consider this sample workflow: two data analysts [BYU, Glo] intend to investigate supermarket transactions dataset [Tab] using Tableau [Mur13]. Their workflow can be summarized as downloading the dataset, verifying it is correct, and checking the store's profitability by creating and analyzing various visualizations. We propose that by mapping each step of the users' workflow to entities of a VA model [FWR*17], VAKG provides a knowledge graph structure that relates user behavior and knowledge acquisition. Then, the knowledge graph can be built by recording each user's behavior and thought process. For instance, "creating a profitability bar chart" would be related to the next task of "inspecting the tallest bar" and the new knowledge of "country X is the most profitable". Finally, this knowledge graph can be used for downstream tasks using graph analytics. Questions like "Which user had more insights during the process?" or "Which user took the least amount of time/steps to find the answer?" can then be answered through the page-rank and shortest-paths algorithms, respectively. Therefore, VAKG not just models a workflow but also defines what data is relevant to be stored, such as user insights and interactions, in order to analyze the users' knowledge-gathering process, providing a unified and repeatable theoretical approach to bridge VA knowledge models, behavior provenance and knowledge provenance.

The remainder of the paper is structured as follows. In Sec. 2 and Sec. 3, we introduce relevant concepts and discuss related work involving techniques that seek to formalize the VA knowledge flow, usages of knowledge graphs within VA, including how they differ from VAKG, and other concepts which tackle the ongoing knowledge evolution during data analysis. In Sec. 4, we extend the existing works of the theoretical knowledge model of VA to formalize VAKG. In Sec. 4.3, we present possible applications of VAKG while comparing it with existing methods and justifications for further extending VAKG. We conducted a case study to demonstrate the practical application of the VAKG to a VA tool that analyses interactive clustering of textual documents in Sec. 4.3 called ModKT. The researchers developing ModKT are using the results to decide the next steps in their work. Finally, in Sec. 5, we discuss current limitations and the next steps within our research plan. In Sec. 6, we draw our conclusions.

2. Theoretical Background and Definitions

Researchers typically prefer to define their workflow descriptively for particular use cases or follow certain well-tested processes. Theoretical research in the model design of VA workflows reflects this diversity very well. To properly position VAKG within the theoretical literature, we first define how the theoretical literature sets itself. Throughout this paper, we will follow the definitions of Chen

et al. [CGJ*17] where the contribution of theoretical VA works is categorized as one or more of the following:

Principles and Guidelines: Qualitative descriptions or rules which define a process that may lead to the desired outcome. Examples can be found in works that extract the qualitative elements of a VA workflow and define rules based on it [SKKC18, BM13].

Taxonomy and Ontology: A collection of concepts that defines a well-defined structure. Such research usually focuses on a novel theoretic ontology to structure the knowledge generation workflow [SSS*14, vLFB*14, SKKC18, CJX20, CE19, PV13].

Conceptual models: Abstract representation of a real-world process using a collection of theoretical taxonomies, typologies, and guidelines. For our purposes, a *VA knowledge model* is a model of a user's knowledge generation throughout a VA process. Arguably the most prominent example of such a model is of [SSS*14]. Generally speaking, knowledge modeling defines a workflow where insights lead to knowledge generation [ALA*18].

Theoretic frameworks: Collection of operators which to measure a process (e.g., mathematical operators). For instance, the *theoretic system* defined by Federico et al. [FWR*17] can describe and measure the process of many existing VA systems and tools.

Quantitative laws: Describes causal relationships between conceptual models by means of a theoretic framework. For example, Federico et al. [FWR*17] applies this concept when comparing multiple VA knowledge models.

Theoretic systems: An extension of a conceptual model which uses theoretic frameworks to define a real-world process formally. Federico et al. [FWR*17] extends several conceptual models in such a way as to formalize its methodology.

These concepts are not consistently used in the VA literature [FWR*17]. In order to better contextualize VAKG's goals, VAKG itself defines a theoretic system based on the set-theory theoretic framework, the conceptual model of Sacha et al. [SSS*14], and the ontology of Federico et al. [FWR*17]. Beyond theory, we propose the practical use of VAKG by applying the proposed theoretic system in practice by performing behavior and knowledge provenance analytics. Because of this duality of VAKG, we classify it as a *conceptual framework*. Nevertheless, the goal of VAKG was defined by investigating the connections between theoretical and practical related works.

3. Related Works

This section presents an overview of how existing theoretical and non-theoretical works are related to VAKG while also considering the definitions of Sec. 2.

3.1. Related Theoretical Works

Knowledge modeling defines a workflow where user insights lead to knowledge generation [SSS*14, ALA*18]. For this, it defines the relationship between users' interactivity and all computer operations and data [SSS*14]. For instance, Fig. 1(A) summarizes this knowledge model, showing how knowledge generation and user

interactivity are linked. Although such work is instrumental as a foundation throughout the VA literature, it cannot be directly applied in practice for provenance analysis.

On the other hand, ontology structures [SSS*14, vLFB*14, SKKC18, CJX20, CE19, PV13] are being used as a means to link knowledge models to real-world workflows. Vis4ML [SKKC18], for instance, describes an ontology for machine learning in VA, and, with it, users can easily model and structure a machine learning workflow. Howsoever relevant these works may be for VAKG, their contribution is still only theoretical, not tackling how to store any data generated from executing a VA workflow nor discussing how or if such data can be collected and used for downstream tasks, such as data analysis. In other words, research on taxonomies and ontologies that structures knowledge gathering in VA does not, by design [CGJ*17], provide an overarching *theoretic system* to link VA theory and the practice of provenance.

Since the origin of VA, significant work has been done to demonstrate the breadth and depth of *knowledge* within VA [SSS*14]. The *theoretic system* of Federico et al. [FWR*17] is versatile enough to describe many existing VA tools. More specifically, they show how the subsequent interactions and *feedbacks* between the user and the computer are related. They also describe how automatic processes in data mining can generate new visualizations or how machine learning can help the user understand the data itself. Nevertheless, although the works listed and described by Federico et al. [FWR*17] may differ, VA's purpose of creating insight or knowledge through a given workflow is common to all of them and is generally done through interactivity between the user and computer [FWR*17, SSS*14, CE19]. Even though their *theoretic system* can formalize the VA knowledge model and exemplify its application in practice, it by itself still lacks an *ontology* to structure, store, and relate the provenance-related data, such as user behavior and the knowledge gathered.

Although the presented theoretical research, such as knowledge models, taxonomies, ontologies, and theoretic systems, are instrumental to understanding how current VA systems produce knowledge, we have also identified their insufficiency in providing insights into the ongoing knowledge generation process throughout a VA workflow. In other words, they cannot be used to simultaneously model, store, and create links between a VA tool's usage, the user's behavior during a VA workflow, and the user's knowledge-gathering process. VAKG attempts to bridge this gap. But our work does not try to redefine any of the taxonomies and principles described so far. Instead, VAKG uses the same taxonomies and principles as most [FWR*17, CE19, PV13]. Also, although VAKG provides a more comprehensive structure to relate user behavior and the knowledge-gathering process, we recognize the existing works' advantage in other areas (e.g., data mining [SKKC18] and machine learning [vRMB*19, SKKC18]). Therefore VAKG does not aim to supersede existing structures or ontologies with its own. Instead, VAKG requires that a given VA tool be modeled using existing VA models, then used to define its *Knowledge Graph* structure. Thus, VAKG bridges the gap between VA theory and its applicability in practice to provide a cohesive structure to relate and analyze user behavior and knowledge gathering.

3.2. Related Applications and Frameworks

Theoretical research on VA's knowledge model has tackled the problem of knowledge gathering in many different ways. However, knowledge gathering within these works and systems is seen only as theoretical background. Federico et al. [FWR*17] lists many systems where a notable example is the work by Keim et al. [KKEM10], which creates an application-specific knowledge-gathering process by utilizing automated analysis with human interaction; however, by verifying these related works, we note a lack of standardization of how to apply the theory in practice. Federico et al. [FWR*17] argues that since this knowledge-gathering loop is conceptual, it is "often inconsistently used," which shows a missed opportunity to define how to apply such theory in practice in a consistent way. This inconsistency has another consequence: although their results relate to each other, these works do not seem to be able to communicate. In other words, we are unable to compare their results.

Furthermore, the two sides of knowledge gathering are often not well separated: the temporal sequence and the workflow's *state space*, which denotes the set of all possible states independent of time. In other words, although a knowledge-gathering process can be defined as a linear sequence of new knowledge "events" over time, it can also be defined as a time-independent set of all gathered knowledge. With VAKG, we first explain the advantages of separating these concepts and the using each of the concepts in a unified framework. Different from other works [FWR*17], VAKG uses this as one of its core design goals.

User Behaviour Tracking and Behavior Provenance: User-tracking and behavior analysis research has also been active [XOW*20]. For instance, the user-tracking taxonomy of von Landesberger et al. [vLFB*14] models user behavior as a graph for analytical purposes. However, VA tools cannot integrate directly with theoretical works such as these. Instead, existing VA systems use these taxonomies as a theoretical or conceptual background while using the user-tracking data solely for specific domain use cases, as is extensively discussed by Xu et al. [XOW*20]. For instance, the user's *Tacit Knowledge* [FWR*17] is tracked in VA by many different feedback methods, such as manual feedback systems [BHZ*17, MHK*19], manual annotations over visualizations [SRE*19], and inference methods that attempt to discover the user's insights by analyzing their interactivity patterns [NXB*16, BH19]. However, these works do not directly use any previously discussed theoretical results. Instead, they are only seen as a motivation for their domain-specific solutions. Among these VA systems, InsideInsights [MHK*19] and SenseMap [NXB*16] are the only ones that get close to addressing this limitation. SenseMap first creates a graph network with behavior provenance, then allows users to analyze the recorded graph by manually constructing a so-called "Knowledge Map". InsideInsights, instead, records user behavior and user annotations simultaneously during the user's analytical process. Though InsideInsights and SenseMap provide a way to record and analyze user behavior, the proposed solutions are domain-specific and do not discuss the relationship between users' behavior and the knowledge gathered by the user. For instance, InsideInsights does not allow tracking auto-generated insights [SSSEA19] and does not account for automatic computer processes [FWR*17] or

external agents [EAM22, MGG*23]. VAKG, however, also tackles these aspects.

Knowledge Provenance: Significant research has been done to better understand the concept and applicability of knowledge gathering in practice regarding *knowledge provenance*. Knowledge provenance is a specialization of *Data Provenance* [dSDMM03, FCM18] for collecting, storing, and tracking users' knowledge-related events. Knowledge provenance researchers argue that tracking user's knowledge gathering can be done by recording any change in the available datasets [dCCM09] (e.g., data pre-processing) or updates in visualizations [BH19, XOW*20, vLFB*14]. Among such works, Chang et al. [CZGR09] attempt to use visual analysis within a Knowledge Base system, storing knowledge extracted from experts into a "compressed" format. Works such as these show examples of applying provenance to understand users' knowledge gathering.

Still, although these works describe ways to link knowledge gathering to user interactions, it is rare to see a differentiation between the temporal sequences of user-generated events and the atemporal *state space* of the VA workflow. Therefore, the following two concepts are either merged or ambiguous in these works: the *temporal* aspect, which indicates what and when users executed VA tasks, and the *atemporal* aspect, which indicates what the possible VA workflow states and how they transition between each other are. Instead, when these works explicitly define a structure, they either store the temporal sequences of events without indicating whether they occurred previously or the state space without recording the temporal sequence of events. Similarly, these works assume that knowledge provenance is a subset of data provenance, or in other words, that all knowledge-related changes can be extracted from the user's behavior. This does not match with the knowledge definition of VA's knowledge models [SSS*14, FWR*17] where certain concepts, like behavior and knowledge, are separate. Likewise, most related works do not tackle how to interpret multi-user VA workflows [BH19], nor allow for comparisons between the user's exploratory space when compared to their motifs [XOW*20]. VAKG bridges these gaps by modeling the difference between behavior and knowledge provenance and the difference between temporal events and temporal state space. VAKG encodes this model into a knowledge graph that relates users' behavior and knowledge-gathering sessions.

Knowledge Graphs (KGs): While Knowledge Provenance focuses on tracking and storing knowledge, *Knowledge Graphs* (KGs) [FŞA*20, CJX20, LAB*23] have aimed to be a proper way to structure and analyze knowledge-related data. KG is a widely used technique to structure knowledge as a graph network, usually done by formalizing the structure as an ontology through the Web Ontology Language format [CE19, SKKC18, vRMB*19]. For instance, DBPedia [ABK*07] uses ontology design and KGs to transform unstructured knowledge into structured knowledge. In other words, KGs are a graph database of knowledge that employs knowledge model [SSS*14] ontologies. Compared to typical databases, the structure of KGs focuses less on the usual row-based structure [CXD*20] but uses the relationships between taxonomies as the foundation of knowledge. Although KG itself focuses on the structure of knowledge-related data, it is supported by various other graph-theory contributions, such as Graph Neural Networks (GNNs) [JWW*20], graph visualizations [CDH*16, HZR*19] and

graph operations [IGC*20], like Page Rank and Traveling Salesman. KGs are, therefore, not limited to only providing a functional structure, but given a KG, users can employ graph analysis techniques to query and analyze the data.

Temporal Knowledge Graphs (TKGs): A notable sub-type of KGs is *Temporal Knowledge Graphs (TKGs)*, where the graph edges encodes the temporal relationship of the data, such as "order of events" or "time difference between events" [GD18]. That is, while a KG is a graph structure where knowledge reasoning is modeled as connections between classes or properties, such as "George Washington *is a* human" and "Canada *is a* country", a *Temporal Knowledge Graph (TKG)* models these connections as the temporal relationship between the classes or properties. Many types of TKGs exist, and their temporal relationship varies among them. For instance, TKGs can relate two nodes by temporal co-occurrence. An example of such a KG would be all purchases done between different businesses within a supply chain, where the product "Mayonese" may have been bought by "Walmart" from the seller "Hellmann's" on "25/06". In this TKG, the connection between the three nodes: Walmart, Hellmann's, and Mayonnaise, would be "25/06". Though some existing works which define knowledge graphs [CXD*20, XOW*20, NVJ20, LAB*23] or ontologies [SKKC18, CE19] are already used for structuring knowledge and behavior provenance, no current work, as far as the authors know, uses TKGs to structure knowledge provenance.

Process Mining: The act of structuring and analyzing a process in a graph format has been extensively researched by Process Mining [VdAW04]. Indeed, the relationship between Process Mining and VA has grown tremendously in recent years. Process Mining proposes a way to define any given process by a workflow consisting of nodes and their relationships. The concepts of *events* and *knowledge graphs*, which are very relevant for VAKG, have appeared in many recent works [Fah22], showing how Process Mining is a proven form of modeling processes for provenance purposes [ZHL*11, VdA15]. Yet, Process Mining is centered on behavior and events, that is, behavior provenance. VAKG aims to relate behavior to *knowledge generation*, which differs from existing works' goals.

3.3. Survey Compilation and Goals

We compiled a survey analyzing the most relevant literature cited so far to verify how the theoretical concepts are applied in practical related works. We found that the main differentiation of VAKG is its theoretically grounded pipeline, which, in a simplified manner, one must: model a given VA tool using a VA model and ontology [FWR*17], declare a knowledge graph structure that matches said ontology, perform data collection through behavior and knowledge provenance to populate the knowledge graph, and finally analyze said knowledge graph (see Sec. 4). Thus, VAKG's major goals are:

- **G1. Analysis-centric VA Model:** Temporal and atemporal interpretations of *Human* and *Machine* components of the VA workflow are used, but inconsistently, so VAKG proposes a consistent one which partitions the VA workflow as:
  - **G1.1:** Temporal-sequences of user's knowledge gathering (Knowledge Provenance or *Human Updates*) [SSS*14,

SKKC18, PV13, vLFB*14, FWR*17, BM13, vRMB*19, CJX20, JJQ*20, dCCM09, BH19, Cli12, MHK*19, BHZ*17];
  - **G1.2:** User intentions and insights which occur within a VA workflow (*Human State-Space* or just Human State) [SSS*14, SKKC18, PV13, FWR*17, BM13, vRMB*19, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, BH19, HMSA08, Cli12, MHK*19];
  - **G1.3:** The VA tool's states during all VA workflows are modified due to user behavior (*Machine State-Space* or just Machine State) [SSS*14, SKKC18, PV13, vLFB*14, BM13, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, CFS*06, dCCM09, BH19, HMSA08, Cli12, SSSEA19];
  - **G1.4:** Temporal-sequences of the VA tool events/tasks which are executed during VA workflow sessions (Behaviour Provenance or *Machine Updates*) [SSS*14, SKKC18, PV13, FWR*17, BM13, ABK*07, JJQ*20, CFS*06, dCCM09, BH19, SSSEA19, MHK*19, BHZ*17];

- **G2. Ontology of the VA Workflow:** Formalization of a structure that, while being rooted in an existing VA knowledge model [SSS*14, FWR*17], describes the VA workflow following **G1** [SKKC18, PV13, BM13, vRMB*19, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, dCCM09];

- **G3. Data Retention:** The structure is used as a schema of a data retention solution where to collect and store user behaviors and interactions during a VA workflow [PV13, vLFB*14, FWR*17, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, CFS*06, BH19, HMSA08, SSSEA19, MHK*19, BHZ*17];

- **G4. Data Analysis Capabilities:** Use the data and/or structure to perform analysis, such as per-user analysis, user comparison, usage comparison, and so on [vRMB*19, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, CFS*06, dCCM09, HMSA08, Cli12, SSSEA19, BHZ*17];

The next section describes how VAKG reaches these goals.

4. The VAKG Conceptual Framework

Let's assume a group of researchers created a VA tool for the analysis of temporal series and now wants to understand if, how, and what users learn while using their tool. The *Visual Analytics Knowledge Graph (VAKG)* method gives this group a formalized process to extract the underlying VA model of a VA tool, design a knowledge graph that follows the model, and define which data from the user needs to be collected for a thorough provenance of the user's behavior when using the tool and their newly acquired knowledge from the tool.

First, VAKG requires that a VA knowledge model is matched to the tool [SSS*14, FWR*17] (see Fig. 1[A]). By VAKG reinterpretation of the model in the lens of *Set Theory* (**G1**), VAKG identifies what are the unique elements that constitute a VA workflow of that specific VA tool and what are their relationships to each other. This VA workflow is then structured following VAKG's *ontology* that relates the users' interaction events and knowledge generation (see Fig. 1(B) and **G2**). The result is a knowledge graph structure that separates the workflow's temporal aspect, which is defined as behavior sequences of events (**G1.4**) and knowledge-gathering sequences of events (**G1.1**), the workflow's atemporal aspect, which is structured as the VA tool's *state-space*(**G1.3**) and the users' knowledge
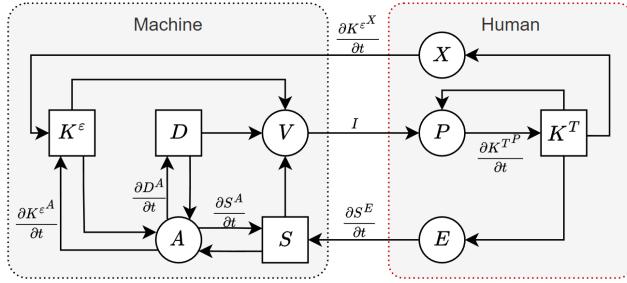
Figure 2: *Conceptual Model of Knowledge-Assisted VA [FWR*17]. VAKG structures its mathematical framework by deriving the equations from this model.*

state-space (**G1.2**). VAKG then uses the knowledge graph structure as the design pattern for a multi-layer *Temporal Knowledge Graph (TKG)* where the VA tool can record user sessions (**G3**). Finally, this populated knowledge graph is available to users, such as the research group of the example above, to apply graph-network techniques to analyze, predict, and recommend user behavior and knowledge-gathering effectiveness when using the tool (**G4**).

4.1. Foundation: VA Knowledge Model and Set Theory Reinterpretation

The theoretical background of VA's knowledge model is a foundation work for research within VA (see Sec. 3). Unlike such works, we use the knowledge model of Sacha et al. [SSS*14] as a foundation to formalize and derive VAKG (**G1**). This section reinterprets the VA knowledge model to define the four aspects of our Analysis-centric VA Model: human update, human state, machine state, and machine update.

The simplistic representation of VA's knowledge model shown in Fig. 1(A) characterizes its two main actors: *Humans* and *Machines*. This concept originates from the literature where knowledge is generated over time [SSS*14] though the interaction between Human and Machine [FWR*17]. The literature also proposes a mathematical interpretation of the VA model called the "Conceptual Model of Knowledge-Assisted VA" as the foundation of the knowledge model, which is expressed visually in Fig. 2.

All in all, the VA interactivity model is divided between two separate actors (machine and human) and describes how knowledge is generated, converted, and used within the VA discourse. Each actor is then associated with a taxonomy of available actions (1): analysis A, visualization V, externalization X, perception/cognition P, and exploration E. These actions are connected by intermediate stateful taxonomies (2): explicit knowledge K^ε, data D, specification S, and tacit knowledge K^T; and (3) a non-persistent artifact: image I" (see Fig. 2).

From Fig. 2, we find that the circle nodes $\{V, P, E, X, A\}$ represent elements that cause changes within a VA tool. For instance, the visualization **V** resides in the machine space, and it causes changes in the perception/cognition of the user **P**, providing new insights. Similarly, the exploration task **E**, which is in the human space, executed by

the user can update the VA tool's specification **S**, which may update the data or visualization being shown within the VA tool. From Fig. 2, we also find that the set of rectangle nodes $\{K^T, D, S, K^\varepsilon\}$ represents static information. For example, elements such as data **D**, specification **S**, and tacit knowledge **Kt** represent the fact that the VA workflow has static information about a dataset, the state of the VA tool, and the user's tacit knowledge, respectively.

From Federico et al. [FWR*17], we can identify all the moving parts within the knowledge model iterative loop of a given VA tool. The first contribution of VAKG is to reinterpret the iterative loop of Fig. 2 through the lens of *Set Theory*, allowing this process to be applied to other VA models and tools. Therefore, first, we define four sets of information: the *Machine Update* $U_t^m = \{V_t, A_t\}$, the *Machine State* $S_t^m = \{D_t, S_t, K_{t+1}^\varepsilon\}$, the *Human Update* $U_t^h = \{X_t, P_t, E_t\}$, and the *Human State* $S_t^h = \{K_t^T\}$. Next, from Fig. 2, we extract how each of these elements relates to each other. Each equation below represents which information (rectangle node) directly depends on a process (circle nodes) and which processes directly depend on information:

$$K_{t+1}^T \Leftarrow P_{t+1} \Leftarrow K_t^T + V_t(I) \tag{1}$$

$$K_{t+1}^\varepsilon \Leftarrow X_{t+1} + A_{t+1} \Leftarrow K_t^T + (K_t^\varepsilon + S_t + D_t) \tag{2}$$

$$S_{t+1} \Leftarrow E_{t+1} + A_{t+1} \Leftarrow K_t^T + (K_t^\varepsilon + S_t + D_t) \tag{3}$$

$$D_{t+1} \Leftarrow A_{t+1} \Leftarrow K_t^\varepsilon + S_t + D_t \tag{4}$$

By using these equations, we reach that the human state and machine state are updated as follows:

$$S_{t+1}^h = \{K_{t+1}^T\} \Leftarrow U_{t+1}^h(S_t^m) \tag{5}$$

$$S_{t+1}^m = \{D_{t+1}, S_{t+1}, K_{t+1}^\varepsilon\} \Leftarrow U_{t+1}^m(S_t^m) + U_{t+1}^h(S_t^h) \tag{6}$$

That is, the *human state* is updated due to a *human update* caused by some change within the *machine state* (Eq. 5). Similarly, the machine state is updated due to a human or machine state change (Eq. 6).

Following this process, any VA tool can be decomposed into the four sets of state and process entities, and the equation list with the relationships between the entities within the sets. Back to the example scenario discussed in the introduction: a data analyst wishes to investigate the supermarket dataset [Tab] using Tableau [Mur13]. In this simplified scenario, the "VA tool" is tableau. The available usages of Tableau can be mapped to the nodes of Fig. 2. For example, let's assume a user wants to create a visualization in Tableau. The data D is the supermarket dataset, the state of tableau S represents what visualization, if any, is currently being shown, and the creation of a new visualization E would update the state of tableau S, generating the new visualization V with which the user can investigate P. In other words, the node E, part of the Human Update set, leads to a new visualization. In mathematical terms: $S_{t+1} \Leftarrow E_{t+1} \equiv S_{t+1}^m \Leftarrow U_{t+1}^h$. That is, in this example a human update U_{t+1}^h led to a new machine state S_{t+1}^m. Still, no new data D has been generated yet, so it was removed from the equation.

Now, if the user discovers a new insight K^T from the visualization and adds it as a custom text or annotation X to the visualization, new

explicit knowledge K^ε would be saved into the tool, causing subsequent updates following the equations above. We see, therefore, that the equations above are helpful not just to define how each of the processes $\{V, P, E, X, A\}$ updates the static information $\{K^T, D, S, K^\varepsilon\}$, but to define how these updates can simultaneously be understood by its ownership (machine or human) and by its timing.

4.2. VAKG Ontology and Knowledge Graph Definition

So far, we have described VAKG's foundation through its four aspects: human update, human state, machine state, and machine update. We also described how each aspect interacts with the others through set equations. Yet, to store data of the users' knowledge generation process, VAKG defines a *Knowledge Graph* (KG) structure where its nodes and relationships correspond to the four aspects of VAKG and their update relationship according to the set equations. This structure allows VAKG to use existing graph databases directly, unlike the domain-specific VA ontologies designed recently [SKKC18, vRMB*19, CE19]. The final structure is exemplified in Fig. 1(C), where the four color-coded horizontal lanes display each of the four aspects.

By following the Web Ontology Language (OWL) [CGJ*17], VAKG divides the space in two ways: by its ownership (human or machine) and by its timing (state or update), which defines the four ontology classes: Human-Update, Human-State, Machine-State, and Machine-Update. From Eqs. 5 and 6, VAKG defines the relationships between the four classes, which are represented in Fig. 3. Namely, the relationship links [1] and [2] found in Fig. 3 relate the previous human/machine state/update to the current one, [3] and [4] represent Eq. 5 where a change in K^T leads to an update in P, and [5] and [6] similarly represent Eq. 6. Finally, VAKG defines two extra relationships [7], synchronizing the two state spaces. This way, if a change in specification (e.g., new visualization) causes the user to perceive something (through [5a]), leading to new knowledge (through [3]), VAKG relates the starting *machine state* and ending *human state* through [7b]. Similarly, if this new knowledge leads the user to externalize [6a] (e.g., add text to the visualization) ending in a new explicit knowledge K^ε, VAKG relates the starting *human state* and ending *machine state* though [7a]. Fig. 1(B) exemplifies a simple knowledge graph following VAKG's ontology.

4.2.1. VAKG Property Map and Data Collection Guideline

An integral part of our proposal is to record users executing a VA workflow and to enable its usage for analysis. This process is called *provenance* (see Sec. 3). While the usual way of thinking of *Knowledge Graph (KG)* is to focus on *classes* and their *relationships*, VAKG instead gives significant importance to *property-maps* (also called *class properties* or *data properties*). Property-maps employ the idea that every *class* can contain attached data. In VAKG, the property-maps of the four classes of nodes are expected to contain the relevant information of that specific class. For instance, in Fig. 3, we see that the class human state should contain the information related to the user's tacit knowledge K^T, and the machine state information related to the dataset D, specification S, and explicit knowledge K^ε. However, the *property-map* design pattern is interchangeable with the other common design patterns [MM09], which removes any perceived limitation of our approach.
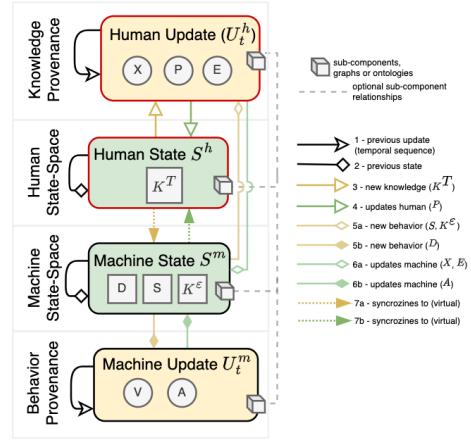

Figure 3: *VAKG ontological design. The four different lanes of VAKG are represented. Two are KGs that describe the possible states of the inner property maps or sub-graphs. Two are TKGs describing the sequences of updates, such as the sequence of insights and knowledge gathering by users in a VA workflow or the sequence of computer events.*

VAKG, therefore, records the information of a node as a property-map, but how should it be recorded? And what information *exactly* should be included? This question is the underlying reason for our descriptive formalization (see Sec 4.1) because, without it, we would not know precisely what information should be stored in each of the node's property-maps. For instance, we have previously described how a machine state would store information related to the dataset D, specification S, and explicit knowledge K^ε, but how much of such information should be stored? Although theoretically, one could argue that storing all information related to a given state is the solution.

It is not reasonable to expect that the usage of VAKG would necessarily require such an amount of information. Therefore, we propose that the property-map of any *State* should, at the very least, uniquely identify that specific *State* within the entire state-space of VAKG. Similarly, the property-map of any *Update* should uniquely identify the changes between the two Machine or two Human *States*, including the timestamp of when the change occurred. This definition establishes that a given Machine or Human *State* can repeat if the same condition occurs multiple times. It is important to note that since each specific use case of VAKG may vary, this part of VAKG is treated as a *design guideline*.

Therefore, it is essential to note that the center two lanes of Fig. 1(B) and Fig. 3 are *atemporal* because their connection is not temporally dependent. In other words, *Machine* and *Human* states are related not through temporal dependency but through their transition relationship. Structures like finite-state machines and discrete-time Markov chains also use atemporal transition relationships similar to VAKG. For example, a machine state is related to a human state through Fig. 3[7b] if that machine state S^m caused the human state S^h to leave a prior state S^h_a and reaches another S^h_b. This may also be read as "S^h_a lead to S^h_b when S^m happened" where

the word "when" does not refer to "exact time" but to the idea of "consequence" instead.

This way, by repeating an earlier example, if a change in specification (e.g., new visualization) within a machine state S^m causes the user to perceive something (through [5a]), leading to new knowledge (through [3]) and consequently a new human state S_b^h, VAKG relates the starting *machine state* S^m and ending *human state* S_b^h through [7b]. VAKG also links the two human states by relationship [2], as shown in Fig. 3. Note that the same process happens when a new human state leads the machine state S_a^m to change to a new state S_b^m.

A consequence of this structure is that nodes in the machine and human space-states which are close (e.g., low number of relationships between the nodes) indicate that these nodes are similar since one state can quickly be reached from another through a low number of "updates". Also, if two machine states or two human states are directly connected, only a single update is responsible.
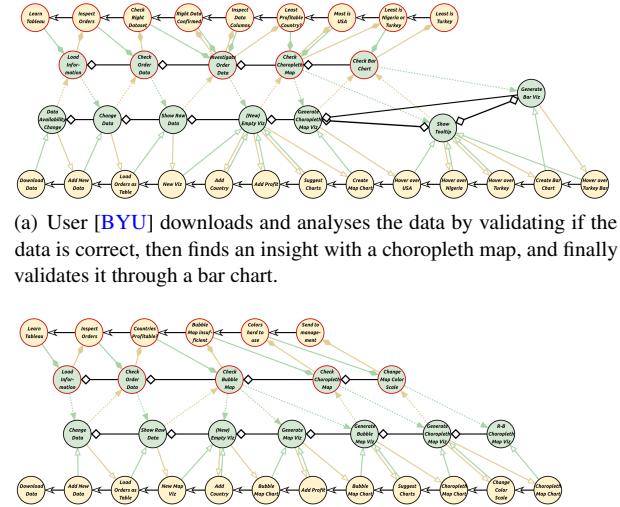
4.3. VAKG in Practice

The running example used until now involves two Tableau users verifying and analyzing a global supermarket store. In this section, we expand on this example as a use case of VAKG. We also discuss another use case with a VA tool called ModKT [RPSM22].
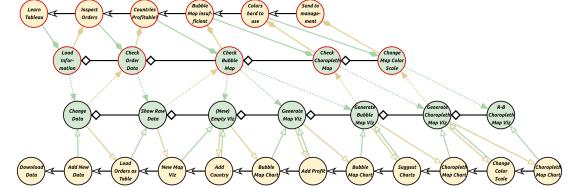
4.3.1. Tableau Use-Case

The first use case to discuss is the running tableau example where two [BYU, Glo] data analysts investigate the supermarket dataset [Tab] using Tableau [Mur13]. By watching the two videos, we can extract a list of tasks, interactions, questions, and insights that each user did. For brevity, here is a small sample of these insights: "task: download data", "task: find least profitable country", "interaction: create new visualization", "interaction: hover over the visualization", and "insight: the least profitable country is C". Each process step can be mapped to one of $\{V, P, E, X, A, K^T, D, S, K^\varepsilon\}$ from Fig. 2. For instance, "download data" is a change to the data D, "create a new visualization" changes the specification S, and "found least profitable country" is a perception process P resulting in new knowledge K^T. By mapping all users' steps to the proper taxonomy, VAKG defines what data of each step one may need, such as the modified data in D, the new visualization type in S, and the new insight in K^T. VAKG also classifies each workflow step as machine update, machine state, human state, and human update (see Fig. 3). For instance, a data change is a new machine state, and a new insight is a new human state. Similarly, VAKG associates the sequence of actions, such as the act of looking at the visualization P is the human update that led to the new insight K^T (see Fig. 2). After applying VAKG to all steps, the result is the knowledge graph seen in Fig. 4 of the videos' content [BYU, Glo].

4.3.2. ModKT Use-Case

We also apply VAKG to ModKT [CMM20, RPSM22], an interactive clustering VA framework, to investigate which features of the tool are being used, how effective the features appear to be given insights gained while using ModKT, and to surface relevant next steps of its authors' research. In this section, we describe the tool, how VAKG was applied to it, and some preliminary results extracted

(a) User [BYU] downloads and analyses the data by validating if the data is correct, then finds an insight with a choropleth map, and finally validates it through a bar chart.

(b) User [Glo] downloads but does not validate the data. He then builds step-by-step a specific choropleth map design to forward to management without discussing any insight.

Figure 4: *VAKG of users performing visual analysis of a global superstore's profitability. The two graph networks are shown separately for better readability. Still, all green nodes (state-space nodes) with the same name are a single node in the VAKG graph, which is composed of both graphs simultaneously where the state-space (green nodes) connects to the individual user's sequence of events (yellow nodes).*

from informal usage of the tool. This example demonstrates how VAKG can be applied to more complex VA tools and workflows.

ModKT is a tool that ingests a set of documents, such as research articles, extracts key terms of each document, and applies key-term-based clustering [SNMM18] to the corpus. ModKT uses the articles' metadata, such as abstract, authors, title, journal, bibliography type, publication year and month, and URL, for clustering. Users can visualize each document through Word Clouds, the corpus of documents through dimensionality reduction, and the comparison of the extracted key terms to custom user-defined words. Users can customize the parameters for clustering and dimensionality reduction to discover sets of (dis)similar documents and visually analyze their (dis)similarities. An overview of the system is presented in Fig. 5.

For this user study, we have set up ModKT with a list of 660 scientific articles in the computer science field covering various text-mining visualization subjects. In order to apply VAKG to it, we follow the methodology process of Sec. 4: model the VA tool, structure the knowledge graph, perform provenance to store user sessions with the tool, and analyze the resulting knowledge graph.

Due to the data and interactions used and expected by ModKT, we notice that even though the VA knowledge model of Federico et al. [FWR*17] (see Fig.2) could be used, it has elements that are not used by the tool, differing from examples given so far. For instance, ModKT does not allow externalizing knowledge X into new explicit knowledge K^ε.
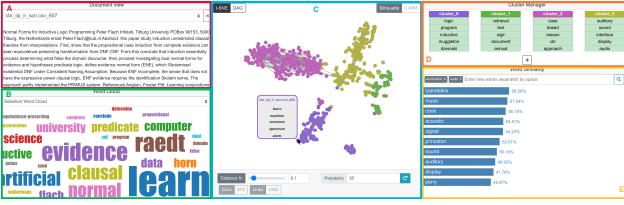
Figure 5: *The modular architecture interface provides a glimpse into its functionality, operating on a collection of 660 documents related to computer science subjects. The interface includes several components: A Document view (A) that shows the content of a selected document; a Word cloud view (B) that presents either the focus document or a cluster in a visual form; a Graph view (C) that illustrates the similarity relationships between the documents in the corpus; a Cluster Manager (D) allowing users to examine clusters and provide feedback to the clustering algorithm; and a Word similarity view (E) which presents a bar chart indicating the similarity between user-provided query words and the most similar identified words.*
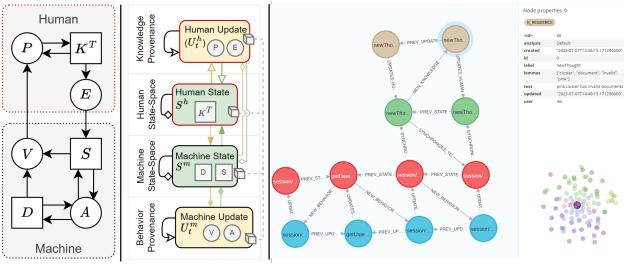

Figure 6: *VA knowledge model of ModKT [Left], its corresponding VAKG structure [Center], and a knowledge graph generated using such structure [Right]. The model and structure only differ from Fig 2 due to the removal of elements X and K^ε. The knowledge graph shows four interactions and two user thoughts, the last of which is highlighted and displayed on the right panel as "Pink cluster has invalid documents". This thought was provided by the user when the pink-colored documents were investigated, as shown by the ModKT dimensionality reduction visualization on the bottom right.*

So far, VAKG has been exemplified only on one of the available VA models [FWR*17]. However, we can apply the same VAKG *theoretic framework* to other VA models by following the same procedure of applying a set Theory reinterpretation to the VA model and extracting the VAKG ontology out of the equations. In the case of ModKT, let's consider that its VA model follows Fig. 6[Left]. The difference between this new model and the one used in Sec. 4 is the absence of X, K^ε and any relationships that either X, K^ε had with any other entity of the model. Following our framework, the VAKG ontology is shown in Fig. 6[Center].

Next, we apply provenance to the ModKT tool. For this, we developed a sample implementation of this VAKG structure [Chr23] that receives an API call from ModKT at every user interaction. This sample implementation collects and populates a knowledge graph

with user behavior data $\{V, S, D, A\}$, which is collected from mouse interactivity, and user thoughts $\{P, E, K^T\}$, which are collected by asking the user to type or speak into the microphone. Using speech-to-text and natural language processing, we extract keywords from the user's text and associate text and keywords with the user's behavior at the time. One of the collected user behavior and thought processes is shown in Fig. 6[Right].

Next, we requested three researchers from our lab to use ModKT with VAKG. They were given 400 documents with abstracts and titles and were tasked with finding visualization-related articles that could be included in their next research article. With the resulting knowledge graph, we could understand how the tool is generally used, list several shortcomings of the tool, and compare how the users differ in their process. The full resulting VAKG knowledge graph is shown in Fig. 7.

With a knowledge graph generated, we can now explore the graph through the node-link diagram of Fig. 7. Although all three users (B, C, and D) started out with a T-SNE projected visualization, user [C] immediately changed to a force-based layout because of the large amount of overlap, which was included in his Human Update "T-SNE not useful, switching to DAG". Though user [C] started by interacting with the visualization, user [B], instead, started by changing the clustering parameters, aiming to create a cluster to show documents related to visualization. Although user [B] could create such a cluster, their process was thwarted because the vast majority of the abstracts found were focused on NLP research and not visualization.

Interactivity-wise, the graph shows that although users took different approaches. To analyze common patterns among users, we could investigate the graph through the visualization, but for better scalability, we opted to run graph queries [Neo12] to fetch certain information. For instance, by querying for the nodes where the users used the forced-based layout (DAG), we can see that all three users used the forced-based layout (DAG) and changed its parameters at some point. Also, by fetching which documents were clicked by each user, all users were shown to have clicked on some of the documents to read more through the abstract and word cloud view. That said, all users were also shown not to have been very successful in finding visualization-related abstracts, which indicates that the issue was not the users nor the tool but the insufficient number of documents loaded into the tool.

Feedback related to the tool functionality was also collected from the users. They discussed topics specific to ModKT, such as layout problems, the aforementioned T-SNE overlap problem, a less-than-ideal experience when reading the abstracts, and little usefulness of the word cloud. VAKG also collected indirect feedback on the tool's functionality. For instance, using simple counting and the aforementioned Page-Rank algorithms, we queried the number of state nodes visited by multiple users, but it was very small. That is, the three users had nearly no overlap in their interactivity, showing that the search space of the tool is vast, likely too vast. The tool's researchers concluded that reducing the possible interactivity and replacing text-only panels with static visualizations is a potentially good next step for the tool. This was corroborated by counting the number of interactions the users had until they reached certain conclusions.
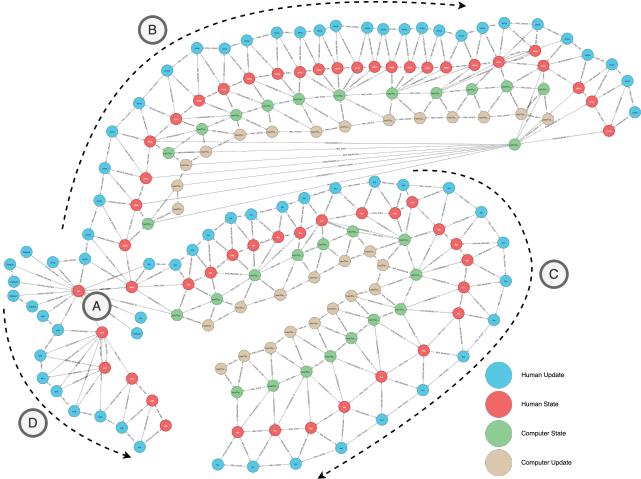
Figure 7: *VAKG generated from three users interacting with ModKT. All users start at [A]. Users B and C had the same first step before diverging. User B performed two investigations, one with many steps [top] and a short one [bottom], ending in the same machine state. User D provided no thoughts, but the collected interactivity showed a back-and-forth interactivity pattern before concluding.*

Though VAKG, ModKT researchers could analyze user exploration paths, check which features of ModKT were most and least used, check which clustering parameters were used, and collect much feedback for future steps. ModKT researchers claim to have gained insights into the tool's capabilities and limitations by visualizing and analyzing the workflow of the individual users, giving them valuable insights into the next step of their research.

While this process could have been done through surveys, thinking-aloud sessions, screen recording, and other manual techniques, the entire process of collection and structuring was done automatically by VAKG. Indeed, ModKT researchers praised VAKG, indicating that future user studies of their tool would be able to be done in a much more automatic and scalable manner. What previously would involve planning and manual labor, now users of ModKT just had to do interactions in an unsupervised environment and write or speak their thoughts into the built-in text widget added to ModKT. The sample implementation provided [Chr23] collected and populated the knowledge graph shown in Fig. 7, which was then analyzed to reach the conclusions above.

4.3.3. Using VAKG for Analysis

VAKG's structure allows users to leverage their existing techniques to perform analysis. The VAKG of Fig. 8 displays features that can occur in a VAKG graph and their respective meaning. Fig. 8 shows two users performing a workflow with diverging paths [A], converging paths [B], backtrack [C], and loop [D]. The example of [D] can also be seen in the Tableau example of Fig. 4(a) when the user interacts with a map visualization through tooltips. Knowing this, we can apply graph analysis techniques to VAKG to answer questions.

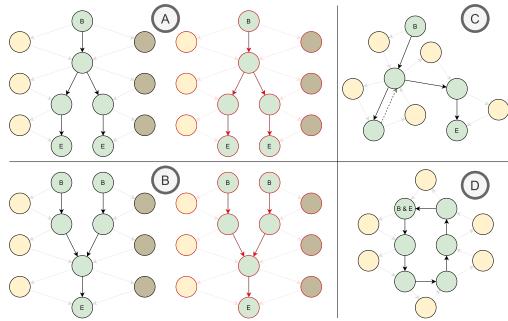One of the most ubiquitous techniques for graph network analysis

Figure 8: *VAKG examples of graph patterns where B is "beginning" and E is "End". In [A], the two users began together but diverged at a certain point, and in [B], they started with different tasks but eventually converged. In [C], users backtrack using an "undo" operation, and in [D], users loop to a prior state.*

is PageRank [Gle15], where it is possible to extract and rank graph nodes based on the number of their connections to other nodes. We can, for instance, extract the users' most "important" state by applying PageRank over VAKG's *Machine-States*. By applying PageRank of the *Machine-States* of the Tableau example (see Fig. 4), we find that user 1 interacted with tooltips more than any other interaction. PageRank also reveals that this specific machine state has the highest amount of *update* relationships among all *Machine-States*. Now, if we consider the full VAKG where both graphs of Fig. 4 are merged, then PageRank of all relationships indicates that the node "New Empty Viz" is the most visited node with 19 connections. By filtering the connections by user, we see that this result is mainly due to the first user, who has 8 connections to his *Machine-Update* timeline versus 4 of the second user. The same analysis can be done from the perspective of the *Human-States* to discover that the "Check Choropleth Map" node is the most connected, which leads us to conclude that the users gathered more insights from the choropleth maps than any other visualizations. It is important to note that although these results can be checked visually in Fig. 4, in examples with hundreds of users where each performs hundreds of interactions, the use of such a ranking algorithm becomes significantly more important.

Other graph network and knowledge graph techniques and tools [HML*14, IGC*20, CXD*20, Neo12, WZW*18] can also be applied. A cycle detection algorithm [QCQ*18] can find any closed cycles within VAKG. By applying it to the same Tableau example, we find that user 2 has no loops within his *Machine-States*, which means that he never retraced his steps (see Fig. 8[D]). By applying a shortest-path algorithm [MAR*17] over the *Human-Update* nodes, we also find that the second user had fewer knowledge-related events, such as insights or questions, than the first, information which can aid in investigating if the tool is properly achieving its goals. We can also apply graph summarization [FCM18] to simplify large graphs, apply KG completion [LNH*18] to analyze whether its users explored all the features of a VA tool, explore KGs through other tools [CXD*20], and analyze the KG by its embedding [WMWG17].

We also extend the same examples to analyze users' workflows while performing tasks with different tools. For instance, assuming

a third user performs a similar workflow to users 1 and 2 [BYU,Glo] but with a different tool, such as PowerBI [FR16]. Graph analysis through PageRank, shortest-path, and other previously discussed techniques can again be used to compare how well the two tools performed. Indeed, if VA tools shared a VAKG of their user evaluation, other researchers would be able to download them and add new data from their own users and/or tools, allowing such researchers to compare their users or tools to existing state-of-the-art tools and past users using techniques like PageRank and shortest path to demonstrate the effectiveness of their new tool in terms of knowledge gathering effectiveness, which can potentially be used to transform the way VA research discusses and discloses user evaluation.

4.4. VAKG Evaluation Discussion

We aimed to follow existing theoretical work's example-based evaluation. However, due to the novelty of VAKG as a conceptual framework with modeling, ontology, structuring, and analysis components, no other work, as far as the authors know, can directly compare. That said, VAKG does not aim to supersede any specialized work in their areas. Instead, VAKG uses related work as its foundation. VAKG can also be easily extended by other works, potentially adding more data to VAKG's property-maps as sub-components.

For instance, here we compare VAKG to Vis4ML [SKKC18], which focuses on proposing an ontology for ML-related tasks. Compared to Vis4ML, VAKG focuses on a different aspect of the VA workflow: the classification of VA taxonomy based on ownership (Human and Machine) and temporality (state-space or process) and the relationship between them. VAKG proposes a knowledge graph structure and a methodology for populating the knowledge graph. Vis4ML only proposes a structure with no direct application to define or populate a knowledge graph or to define how to analyze the resulting data. Indeed, by comparing VAKG to all other related work, VAKG stands out as the only one that proposes a methodology to structure a given VA tool's model as a knowledge graph that can perform knowledge and behavior provenance. Similar results are found when comparing VAKG to other theoretical-focused works [CE19, SSS*14, SBFK16, FWR*17], which are here omitted due to space constraints.

When comparing VAKG to the results of practical related works, we find that VAKG is uniquely positioned to provide a comprehensive knowledge graph for their required behavior and user-knowledge analysis requirements. However, it is important to note that this comparison is limited because VAKG is a conceptual framework. For instance, InsideInsights' results [MHK*19] show that allowing users to visualize the VA workflows of certain analysis processes is highly beneficial through interviews and usage scenarios. In our MobKT user case (see Sec 4.3.2), we confirm that visualizing the resulting knowledge graph is useful. Yet, the results of our work show that VAKG provides a structure for analyzing behavior and knowledge provenance, as opposed to InsideInsights' report of user behavior. Indeed, in practice, most works focus on behavior analysis [vRMB*19, CE19, CJX20, ABK*07, CDH*16, HZR*19, JJQ*20, CFS*06, dCCM09, HMSA08, Cli12, SSSEA19, BHZ*17]. VAKG is novel in its inclusion of knowledge provenance as part of the resulting knowledge graph.

5. Limitations and Future Work

When comparing to other ontologies, it is essential to note that VAKG's focus is not on its descriptive power [SKKC18], but on its ability to model and structure the user's knowledge gain process. Therefore, VAKG does not solve the issue of how to perform user-tracking [MHK*19]. VAKG also does not expand the analytical arsenal of user behavior or provenance techniques [BH19,dCCM09], but provides a novel structure that is optimized for the use of said techniques for various analytical use cases. In future work, we aim to investigate the best approaches for user-tracking, behavior/knowledge provenience, and knowledge graph analysis when applying VAKG in domain-specific use cases. If required, we might contribute novel approaches. We also plan to investigate semi-automatic or automatic provenance techniques to assist the applicability of VAKG.

Although VAKG has focused on defining a property-map way to store the knowledge-gathering process, other works have proposed other methods as well [SKKC18]. That said, knowledge graphs are not limited to a single structure at a time, as is the nature of graph data, so it is easy to imagine that two different knowledge graphs could co-exist. Therefore, although we argued that VAKG's structure is more capable than other existing ontologies, we recognize that this is mainly because the resulting knowledge graph can be extended, allowing others to use different ontologies or models as part of VAKG through custom property-maps or by linking VAKG nodes to a totally separate custom knowledge graphs. However, we believe that this integration needs to be addressed separately in domain-specific frameworks or application use cases. Results and evaluation of these future works will also be driven by their use cases, which do not fit within the contribution presented in this paper. Since existing ontologies [FWR*17,SKKC18,Cur20,XOW*20,vLFB*14] can then co-exist with VAKG, we plan for future work to explore possible combinations of related work's ontologies as future domain-specific contributions.

We have experienced that VAKG can quickly result in large and complex KGs, which are hard to visualize and may cause issues related to storage space if used indiscriminately. So far, we have provided examples that were simple enough to be explained and visualized. Still, we attempted to store dozens of user workflows as a VAKG, and the result was too complex to visualize. Indeed, we recognize that the complexity depends on the modeled and recorded workflow, though graph network analysis is always possible. We plan on investigating better ways to visualize both simple and complex VAKGs, especially when considering what analysis is being done as future work.

The most critical limitation of VAKG, perhaps, is that user-tracking has been broadly seen negatively. User protection laws and initiatives, like Europe's General Data Protection Regulation (GDPR) [Wol] and Apple's "Ask not to track" features, are just a few examples. Although VAKG is not a novel way to perform user-tracking, user consent for tracked and behavior analysis is undoubtedly a relevant concern. However, this concern is not new and is shared by all related works which tackle user-tracking or behavior analysis. We also argue that in many cases, the users of VAKG are the same whose behavior is being tracked, which means that they probably would accept and welcome the necessary tracking since

they would do the analysis. Further study is needed to analyze how impactful this would be.

6. Conclusion

We have presented VAKG, a conceptual framework to structure a given VA tool as a 4-way temporal knowledge graph that describes user behavior and knowledge gathering during the execution of a VA workflow. We propose that by modeling a VA tool with VAKG, we obtain a knowledge graph structure that captures the required substances from user knowledge-gathering sessions. Users then populate the knowledge graph with behavior events, such as interactions, and knowledge events, such as intents and insights. Then, the knowledge graph can be used to analyze user behavior, the knowledge-gathering process, and the interactive relationship between the two. The resulting knowledge graph is by design standardized across users and tools, allowing for graph-based analytics of domain-specific processes (e.g., EDA), usage patterns, and user knowledge gain performance among multi-user and multi-tool scenarios.

In practice, VAKG's resulting graph represents an overview of the VA workflows' usage and the collective experiences and knowledge generated by their users. VAKG is extensible and adaptable to various situations and domains, including its extension to incorporate other models or ontologies. Using VAKG as a provenance architecture, the generated knowledge graph can also be analyzed through existing graph-analytics techniques, such as visualizations, shortest path analysis, and page ranking. We applied VAKG to two examples: data analysis with Tableau and ModKT [RPSM22], and discussed how the resulting knowledge graph allows us to better understand the path taken by the user to reach new knowledge, how users differ in their experience of seeking knowledge, and which parts of the tool were most and least used, among other results. When compared to existing works, VAKG was shown to be unique in its approach in bringing VA model theory into practice for behavior and knowledge provenance tasks.

Acknowledgment

References

[ABK*07] AUER S., BIZER C., KOBILAROV G., LEHMANN J., CYGANIAK R., IVES Z.: Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 2007, pp. 722–735. 4, 5, 11

[ALA*18] ANDRIENKO N., LAMMARSCH T., ANDRIENKO G., FUCHS G., KEIM D., MIKSCH S., RIND A.: Viewing visual analytics as model building. *Computer Graphics Forum 37*, 6 (2018), 275–299. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13324, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13324, doi:https://doi.org/10.1111/cgf.13324. 3

[BH19] BATTLE L., HEER J.: Characterizing exploratory visual analysis: A literature review and evaluation of analytic provenance in tableau. *Computer Graphics Forum 38*, 3 (2019), 145–159. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13678, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13678, doi:https://doi.org/10.1111/cgf.13678. 2, 4, 5, 11

[BHZ*17] BERNARD J., HUTTER M., ZEPPELZAUER M., FELLNER D., SEDLMAIR M.: Comparing visual-interactive labeling with active learning: An experimental study. *IEEE transactions on visualization and computer graphics 24*, 1 (2017), 298–308. 4, 5, 11

[BM13] BREHMER M., MUNZNER T.: A multi-level typology of abstract visualization tasks. *IEEE transactions on visualization and computer graphics 19*, 12 (2013), 2376–2385. 3, 5

[BYU] BYU I.: Tableau practice problems. URL: https://www.youtube.com/embed/B3jKKQrhTko?start=0&end=255. 2, 8, 11

[CDH*16] CHANG S., DAI P., HONG L., SHENG C., ZHANG T., CHI E. H.: Appgrouper: Knowledge-graph-based interactive clustering tool for mobile app search results. In *Proceedings of the 21st International Conference on Intelligent User Interfaces* (2016), pp. 348–358. 4, 5, 11

[CE19] CHEN M., EBERT D. S.: An ontological framework for supporting the design and evaluation of visual analytics systems. *Computer Graphics Forum 38*, 3 (2019), 131–144. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13677, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13677, doi:https://doi.org/10.1111/cgf.13677. 3, 4, 5, 7, 11

[CFS*06] CALLAHAN S. P., FREIRE J., SANTOS E., SCHEIDEGGER C. E., SILVA C. T., VO H. T.: Vistrails: visualization meets data management. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data* (2006), pp. 745–747. 5, 11

[CGJ*17] CHEN M., GRINSTEIN G., JOHNSON C. R., KENNEDY J., TORY M.: Pathways for theoretical advances in visualization. *IEEE computer graphics and applications 37*, 4 (2017), 103–112. 2, 3, 7

[Chr23] CHRISTINO L.: christinoleo/vakg: Zenodo release, July 2023. URL: https://doi.org/10.5281/zenodo.8124221, doi:10.5281/zenodo.8124221. 9, 10

[CJX20] CHEN X., JIA S., XIANG Y.: A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications 141* (2020), 112948. 3, 4, 5, 11

[Cli12] CLIFTON B.: *Advanced web metrics with Google Analytics*. John Wiley & Sons, 2012. 5, 11

[CMM20] CABRAL E. M., MILIOS E. E., MINGHIM R.: Visual analysis of interactive document clustering streams. In *Proceedings of the International Conference on Advanced Visual Interfaces* (2020), pp. 1–3. 8

[Cur20] CURRY E.: *Real-time linked dataspaces: Enabling data ecosystems for intelligent systems*. Springer Nature, 2020. 11

[CXD*20] CASHMAN D., XU S., DAS S., HEIMERL F., LIU C., HUMAYOUN S. R., GLEICHER M., ENDERT A., CHANG R.: Cava: A visual analytics system for exploratory columnar data augmentation using knowledge graphs. *IEEE Transactions on Visualization and Computer Graphics* (2020). 4, 5, 10

[CZGR09] CHANG R., ZIEMKIEWICZ C., GREEN T. M., RIBARSKY W.: Defining insight for visual analytics. *IEEE Computer Graphics and Applications 29*, 2 (2009), 14–17. 4

[dCCM09] DA CRUZ S. M. S., CAMPOS M. L. M., MATTOSO M.: Towards a taxonomy of provenance in scientific workflow management systems. In *2009 Congress on Services-I* (2009), IEEE, pp. 259–266. 4, 5, 11

[dSDMM03] DA SILVA P. P., DEBORAH S., MCGUINNESS D. L., MCCOOL R.: *Knowledge provenance infrastructure*. PhD thesis, Stanford, 2003. 4

[EAM22] EL-ASSADY M., MORUZZI C.: Which biases and reasoning pitfalls do explanations trigger? decomposing communication processes in human–ai interaction. *IEEE Computer Graphics and Applications 42*, 6 (2022), 11–23. 4

[Fah22] FAHLAND D.: Process mining over multiple behavioral dimensions with event knowledge graphs. In *Process Mining Handbook*. Springer, 2022, pp. 274–319. 5

[FCM18] FUJIWARA T., CRNOVRSANIN T., MA K.-L.: Concise provenance of interactive network analysis. *Visual Informatics 2*, 4 (2018), 213–224. 4, 10

[FR16] FERRARI A., RUSSO M.: *Introducing Microsoft Power BI*. Microsoft Press, 2016. 11

[FŞA*20] FENSEL D., ŞIMŞEK U., ANGELE K., HUAMAN E., KÄRLE E., PANASIUK O., TOMA I., UMBRICH J., WAHLER A.: Introduction: what is a knowledge graph? In *Knowledge Graphs*. Springer, 2020, pp. 1–10. 4

[FWR*17] FEDERICO P., WAGNER M., RIND A., AMOR-AMORÓS A., MIKSCH S., AIGNER W.: The role of explicit knowledge: A conceptual model of knowledge-assisted visual analytics. In *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2017), IEEE, pp. 92–103. 2, 3, 4, 5, 6, 8, 9, 11

[GD18] GOTTSCHALK S., DEMIDOVA E.: Eventkg+ tl: creating cross-lingual timelines from an event-centric knowledge graph. In *European Semantic Web Conference* (2018), Springer, pp. 164–169. 5

[Gle15] GLEICH D. F.: Pagerank beyond the web. *siam REVIEW 57*, 3 (2015), 321–363. 10

[Glo] GLOBAL S.: Tableau tutorial - global superstore performance dashboard. URL: https://www.youtube.com/embed/kIZDb_pHvX0?start=187&end=452. 2, 8, 11

[HML*14] HAN W., MIAO Y., LI K., WU M., YANG F., ZHOU L., PRABHAKARAN V., CHEN W., CHEN E.: Chronos: a graph engine for temporal graph analysis. In *Proceedings of the Ninth European Conference on Computer Systems* (2014), pp. 1–14. 10

[HMSA08] HEER J., MACKINLAY J., STOLTE C., AGRAWALA M.: Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE transactions on visualization and computer graphics 14*, 6 (2008), 1189–1196. 2, 5, 11

[HZR*19] HE X., ZHANG R., RIZVI R., VASILAKES J., YANG X., GUO Y., HE Z., PROSPERI M., HUO J., ALPERT J., ET AL.: Aloha: developing an interactive graph-based visualization for dietary supplement knowledge graph through user-centered design. *BMC medical informatics and decision making 19*, 4 (2019), 1–18. 4, 5, 11

[IGC*20] ILIEVSKI F., GARIJO D., CHALUPSKY H., DIVVALA N. T., YAO Y., ROGERS C., LI R., LIU J., SINGH A., SCHWABE D., ET AL.: Kgtk: a toolkit for large knowledge graph manipulation and analysis. In *International Semantic Web Conference* (2020), Springer, pp. 278–293. 5, 10

[JJQ*20] JIN W., JIANG H., QU M., CHEN T., ZHANG C., SZEKELY P., REN X.: Recurrent event network : Global structure inference over temporal knowledge graph, 2020. URL: https://openreview.net/forum?id=SyeyF0VtDr. 5, 11

[JWW*20] JIN Z., WANG Y., WANG Q., MING Y., MA T., QU H.: Gnnvis: A visual analytics approach for prediction error diagnosis of graph neural networks. *arXiv preprint arXiv:2011.11048* (2020). 4

[KKEM10] KEIM D., KOHLHAMMER J., ELLIS G., MANSMANN F.: Mastering the information age: solving problems with visual analytics, 2010. 4

[LAB*23] LI H., APPLEBY G., BRUMAR C. D., CHANG R., SUH A.: Characterizing the users, challenges, and visualization needs of knowledge graphs in practice. *arXiv preprint arXiv:2304.01311* (2023). 4, 5

[LNH*18] LI Q., NJOTOPRAWIRO K. S., HALEEM H., CHEN Q., YI C., MA X.: Embeddingvis: A visual analytics approach to comparative network embedding inspection. In *2018 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2018), IEEE, pp. 48–59. 10

[MAR*17] MADKOUR A., AREF W. G., REHMAN F. U., RAHMAN M. A., BASALAMAH S.: A survey of shortest-path algorithms. *arXiv preprint arXiv:1705.02044* (2017). 10

[MGG*23] MONADJEMI S., GUO M., GOTZ D., GARNETT R., OTTLEY A.: Human-computer collaboration for visual analytics: an agent-based framework. *arXiv preprint arXiv:2304.09415* (2023). 4

[MHK*19] MATHISEN A., HORAK T., KLOKMOSE C. N., GRØNBÆK K., ELMQVIST N.: Insideinsights: Integrating data-driven reporting in collaborative visual analytics. In *Computer Graphics Forum* (2019), vol. 38, Wiley Online Library, pp. 649–661. 2, 4, 5, 11

[MM09] MYROSHNICHENKO I., MURPHY M. C.: Mapping er schemas to owl ontologies. In *2009 IEEE International Conference on Semantic Computing* (2009), IEEE, pp. 324–329. 7

[Mur13] MURRAY D. G.: *Tableau your data!: fast and easy visual analysis with tableau software*. John Wiley & Sons, 2013. 2, 6, 8

[Neo12] NEO4J: *Neo4j - The World's Leading Graph Database*, 2012. URL: http://neo4j.org/. 9, 10

[NVJ20] NGUYEN H. L., VU D. T., JUNG J. J.: Knowledge graph fusion for smart systems: A survey. *Information Fusion 61* (2020), 56–70. 5

[NXB*16] NGUYEN P. H., XU K., BARDILL A., SALMAN B., HERD K., WONG B. W.: Sensemap: Supporting browser-based online sense-making through analytic provenance. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2016), IEEE, pp. 91–100. 4

[PV13] POLOWINSKI J., VOIGT M.: Viso: A shared, formal knowledge base as a foundation for semi-automatic infovis systems. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. ACM digital library, 2013, pp. 1791–1796. 3, 5

[QCQ*18] QIU X., CEN W., QIAN Z., PENG Y., ZHANG Y., LIN X., ZHOU J.: Real-time constrained cycle detection in large dynamic graphs. *Proceedings of the VLDB Endowment 11*, 12 (2018), 1876–1888. 10

[RPSM22] REZAEIPOURFARSANGI S., PEI N., SHERKAT E., MILIOS E.: Interactive clustering and high-recall information retrieval using language models. In *Proceedings of the 2022 International Conference on Advanced Visual Interfaces* (2022), pp. 1–5. 8, 12

[SBFK16] SACHA D., BOESECKE I., FUCHS J., KEIM D. A.: *Analytic behavior and trust building in visual analytics*. The Eurographics Association, 2016. 11

[SKKC18] SACHA D., KRAUS M., KEIM D. A., CHEN M.: Vis4ml: An ontology for visual analytics assisted machine learning. *IEEE transactions on visualization and computer graphics 25*, 1 (2018), 385–395. 2, 3, 4, 5, 7, 11

[SNMM18] SHERKAT E., NOURASHRAFEDDIN S., MILIOS E. E., MINGHIM R.: Interactive document clustering revisited: A visual analytics approach. In *23rd International Conference on Intelligent User Interfaces* (2018), pp. 281–292. 8

[SRE*19] SOARES A., ROSE J., ETEMAD M., RENSO C., MATWIN S.: Vista: A visual analytics platform for semantic annotation of trajectories. In *Proceedings of the 22nd international conference on extending database technology (EDBT)* (2019). 4

[SSS*14] SACHA D., STOFFEL A., STOFFEL F., KWON B. C., ELLIS G., KEIM D. A.: Knowledge generation model for visual analytics. *IEEE transactions on visualization and computer graphics 20*, 12 (2014), 1604–1613. 1, 2, 3, 4, 5, 6, 11

[SSSEA19] SPINNER T., SCHLEGEL U., SCHÄFER H., EL-ASSADY M.: explainer: A visual analytics framework for interactive and explainable machine learning. *IEEE transactions on visualization and computer graphics 26*, 1 (2019), 1064–1074. 4, 5, 11

[Tab] TABLEAU: Tableau training dataset - global superstore. URL: http://www.tableau.com/sites/default/files/training/global_superstore.zip. 2, 6, 8

[VdA15] VAN DER AALST W. M.: Extracting event data from databases to unleash process mining. *BPM-Driving innovation in a digital world* (2015), 105–128. 5

[VdAW04] VAN DER AALST W. M., WEIJTERS A. J.: Process mining: a research agenda. *Computers in industry 53*, 3 (2004), 231–244. 5

[vLFB*14] VON LANDESBERGER T., FIEBIG S., BREMM S., KUIJPER A., FELLNER D. W.: Interaction taxonomy for tracking of user actions in visual analytics applications. In *Handbook of Human Centric Visualization*. Springer, 2014, pp. 653–670. 2, 3, 4, 5, 11

[vRMB*19] VON RUEDEN L., MAYER S., BECKH K., GEORGIEV B., GIESSELBACH S., HEESE R., KIRSCH B., PFROMMER J., PICK A., RAMAMURTHY R., ET AL.: Informed machine learning–a taxonomy and survey of integrating knowledge into learning systems. *arXiv preprint arXiv:1903.12394* (2019). 3, 4, 5, 7, 11

[VW05] VAN WIJK J. J.: The value of visualization. In *VIS 05. IEEE Visualization, 2005.* (2005), IEEE, pp. 79–86. 2

[WMWG17] WANG Q., MAO Z., WANG B., GUO L.: Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering 29*, 12 (2017), 2724–2743. 10

[Wol] WOLFORD B.: 2018 reform of eu data protection rules. URL: https://gdpr.eu/what-is-gdpr/. 11

[WZW*18] WANG H., ZHANG F., WANG J., ZHAO M., LI W., XIE X., GUO M.: Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (2018), pp. 417–426. 10

[XOW*20] XU K., OTTLEY A., WALCHSHOFER C., STREIT M., CHANG R., WENSKOVITCH J.: Survey on the analysis of user interactions and visualization provenance. *Computer Graphics Forum 39*, 3 (2020), 757–783. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14035, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14035, doi:https://doi.org/10.1111/cgf.14035. 2, 4, 5, 11

[ZHL*11] ZENG R., HE X., LI J., LIU Z., VAN DER AALST W. M.: A method to build and analyze scientific workflows from provenance through process mining. In *TaPP* (2011). 5