A Multi-Level Task Framework for Event Sequence Analysis

Kazi Tasnim Zinat, Saimadhav Naga Sakhamuri, Aaron Sun Chen and Zhicheng Liu

Abstract—Despite the development of numerous visual analytics tools for event sequence data across various domains, including but not limited to healthcare, digital marketing, and user behavior analysis, comparing these domain-specific investigations and transferring the results to new datasets and problem areas remain challenging. Task abstractions can help us go beyond domain-specific details, but existing visualization task abstractions are insufficient for event sequence visual analytics because they primarily focus on multivariate datasets and often overlook automated analytical techniques. To address this gap, we propose a domain-agnostic multi-level task framework for event sequence analytics, derived from an analysis of 58 papers that present event sequence visualization systems. Our framework consists of four levels: `objective, intent, strategy, and technique`. Overall `objectives` identify the main goals of analysis. `Intents` comprises five high-level approaches adopted at each analysis step: augment data, simplify data, configure data, configure visualization, and manage provenance. Each `intent` is accomplished through a number of strategies, for instance, data simplification can be achieved through aggregation, summarization, or segmentation. Finally, each `strategy` can be implemented by a set of `techniques` depending on the input and output components. We further show that each technique can be expressed through a quartet of *action-input-output-criteria*. We demonstrate the framework's descriptive power through case studies and discuss its similarities and differences with previous event sequence task taxonomies.

Index Terms—Task Abstraction, Event Sequence Data

1 Introduction

Event sequence data contains a wealth of knowledge, but because of their complexity, it can be challenging for analysts to extract useful insights. Numerous visual analytics systems for event sequence data have thus been developed, each tailoring to particular application domains. For instance, Tipovis [26] assists developmental psychology researchers in the visual analysis of children's behavioral patterns. In clickstream analysis, tools such as Patterns and Sequences [43], Coreflow [42], and Segmentifier [11] enable market analysts to explore and understand customer journeys. SessionViewer [36] helps analysts examine web session logs. Careflow [50], DecisionFlow [17], OutFlow [18, 66], and Cadence [19] help healthcare researchers in identifying trends in patient treatment histories and support clinical decision-making.

These tools vary in application domains, objectives, and techniques. Some tools focus on sequence comparison [26, 55, 72], while others emphasize overview generation [64, 67], pattern exploration [42, 51] or finding similarities [47, 65] across sequences. These tools utilize various techniques, such as sequence alignment [5, 72], pattern mining [42, 43, 63], rule-based exploration [9] and novel visual encodings [62]. Although these domain-specific investigations sometimes hint at the possibility of wider applicability (e.g., [19, 42]), the diversity of objectives and techniques presents challenges in generalizing the findings, comparing tools developed for different applications, and transferring knowledge across domains.

The development of task frameworks has consistently been instrumental in advancing the field of data visualization, however, existing frameworks [2, 7, 59, 61, 70] fall short of fully capturing the unique challenges associated with event sequence data. First, most of these frameworks assume a multivariate data model, overlooking the complexities of event sequence data [5], such as high dimensionality and time variance [32]. Second, they focus mostly on visualization tasks, but event sequence analytics employ visualization in conjunction to pattern mining, unsupervised learning, and data transformation. We need an integrative task framework accounting for all types of techniques. To date, only a few attempts have been made to describe the task space of event sequence visual analytics [14, 49, 52]. However, they tend to mix task descriptions of different levels of granularity in a single taxonomy and do not fully capture the complexities of event sequence analysis tasks.

Inspired by previous works [7, 56] highlighting the multi-dimensional and multi-level nature of visualization tasks, we present a comprehensive end-to-end task framework for event sequence analysis, capturing the analysis process from data preprocessing to provenance tasks. Derived from an extensive analysis of 58 papers, our framework consists of four levels (objective, intent, strategy, and technique), each capturing a different level of abstraction.

The highest level, `objectives`, represents overarching goals such as cohort comparison, anomaly detection, and identifying common behavior. To achieve these `objectives`, users form specific `intents` at each analysis step. We identify five high-level `intents`: augment data, simplify data, configure data, configure visualization, and manage provenance. Each intent is realized through a set of `strategies`, which define the methods used to accomplish the intent. For instance, data simplification can be achieved through aggregation, summarization, or segmentation strategies. Finally, `techniques` are specific implementations of each strategy, expressed as a quartet of dimensions: the *action* performed, *input* data components, desired *output* components, and *criteria* specifying the parameters or conditions for the action. By organizing tasks into this hierarchical structure with multidimensional characterization at the technique level, our framework bridges the gap between high-level objectives and low-level techniques, providing a comprehensive and systematic approach to event sequence analysis.

We evaluate the expressiveness and precision of our framework by comparing it with existing task abstractions for event sequences [14, 49, 54] through case studies, and discuss its implications for future research on event sequence visual analytics.

2 Related Work

We first review task taxonomies for interactive visualizations and their limitations in describing event sequence visual analysis. Then we discuss existing works that discuss tasks in event sequence data.

2.1 Task Abstractions for Visualizations

Numerous theoretical abstractions of visualization tasks have been proposed, including but not limited to: Shneiderman's data types by tasks taxonomy [59], Amar et al.'s low-level analytic tasks [2], Valiati et al.'s formulation encompassing analytic, cognitive, and operational tasks to guide evaluation and design of multidimensional data visualizations [61], Schulz et al.'s design space for visualization task [57], Andrienko and Andrienko's exploratory data analysis tasks [3], Yi et al.'s categorization of user intents in interactive visualization [70], and Heer

• *Kazi Tasnim Zinat, Saimadhav Naga Sakhamuri, Aaron Sun Chen, and Zhicheng Liu are with the University of Maryland. E-mail: {kzintas, leozcliu}@umd.edu, {ssakhamu, achen151}@terpmail.umd.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

and Shneiderman's taxonomy of interactive tasks in visual analysis [27]. These taxonomies, however, mostly focus on a single level of abstraction, and some of them mix low-level actions with high-level goals.

Researchers have argued that tasks are complex theoretical constructs and cannot be fully understood along a single dimension or at a single level of abstraction. Rind et al. propose the idea of task cube to capture the multi-dimensional nature of visualization tasks [56]. Gotz and Zhou [20] present a hierarchy of tasks, sub-tasks, actions, and events to characterize user behavior in visual analytics at multiple degrees of semantic granularity. Brehmer and Munzner [7] introduce a multi-level typology for abstract visualization tasks. The typology is organized around three key questions: why the task is performed, how it is executed, and what the task inputs and outputs are. By utilizing several levels of abstraction, the typology facilitates a more systematic analysis and comparison of visualization tasks across tools and domains. Building upon Brehmer and Munzner's work, Lam et al. [37] derive a framework based on the analysis of design study papers for mapping high-level analysis goals and low-level visualization tasks. Our task framework is inspired by these works' emphasis on the multi-granularity and multi-dimensionality of tasks.

These task abstraction frameworks, designed primarily for multivariate data visualization and analysis, are not readily applicable to event sequence visual analytics due to two main reasons. First, event sequence analysis possess unique complexities, including temporal ordering of events within sequences, and sequences having their own attributes. Additionally, sequences can be grouped into cohorts or divided into segments based on various criteria. These hierarchical complexities and temporal relationships are not addressed in multivariate visualization frameworks [60]. Consequently, event sequence analysis requires a broader set of techniques and a closer examination of the *what* aspects of user actions. For instance, filtering in multivariate datasets is unambiguously about filtering data items based on attribute values, whereas in event sequence analysis, filtering can target various components (e.g., filtering events by frequency, filtering sequences based on milestone events, filtering sequences by sequence attribute)

Second, previous works on task abstraction primarily focus on visualization tasks and do not adequately cover the wide range of machine learning and data transformation tasks that are crucial in event sequence analytics. For instance, pattern mining techniques are frequently employed to discover recurring event subsequences, and clustering algorithms are used to group similar sequences together. Data transformation tasks, such as attribute or value based filtering, sequence segmentation, and event aggregation, are also common in event sequence analysis. However, existing task frameworks often emphasize visualization tasks, such as looking up data items or comparing attributes, and do not provide a comprehensive taxonomy for data transformation tasks. A comprehensive task framework for event sequence analysis must take into consideration both data manipulation and visualization tasks, and how they are integrated in the end-to-end analysis process.

2.2 Surveys and Taxonomies for Event Sequence Visual Analysis

We identify two pieces of work on high-level tasks in event sequence analysis. Plaisant et. al [54] propose a characterization of high-level user tasks for event analytics, such as 'Identify a set of records of interest' and 'Compare two or more sets of records'. They also stress the importance of task descriptions for event sequence analytics. This work sets the stage for developing a common language for comparing tools, and applications of event analytics. However, it does not provide a mapping of high-level goals to low-level techniques.

Guo et al. [25] provide a comprehensive survey of visual analytics tools for event sequence data. They review and categorize the tools by five high-level analytical tasks (summarization, prediction & recommendation, anomaly detection, comparison, and causality analysis), and application domains targeted. They further identify seven common interaction techniques for event sequence visual analytics (e.g., filter, segmentation). Our work shares the motivation of Guo et al. to provide a systematic framework for understanding event sequence analysis. While Guo et al. approach this by categorizing tool functionalities according to high-level analytical tasks, we instead propose a hierarchical typology that maps analytic intents to specific task strategies and techniques.

For lower-level tasks, Du et. al [14] presents 15 strategies to reduce the volume and variety of temporal event sequence data. While these strategies are a valuable tool set of data manipulation tactics, they do not offer task abstractions that link user goals to system actions, nor do they examine tasks aimed at objectives other than reducing volume and variety (e.g., augmenting event sequences for analysis).

A recent workshop paper by Peiris et al. [49] presents a methodology and task typology for time-stamped event sequences (TSES) that has similarities to our work. They provide a list of 23 tasks described in terms of the action, target, and criteria triangle. The paper has a narrower focus on TSES which has continuous values, whereas our work considers a broader scope including both TSES and event sequences with ordered events. These strategies also vary in terms of level of abstraction. In addition, while the paper addresses the multi-dimensional nature of tasks, it does not provide an account of tasks across multiple levels of granularity. In section 5, we provide a comparison of tasks from our framework with the tasks proposed by these three works [14, 49, 54].

3 Method

We adopt an iterative approach to develop the task framework based on empirical data from literature review. Literature reviews are a widely adopted method for task abstraction [1, 20, 70]. Our process consists of two main stages: open coding and axial coding.

3.1 Corpus Assembly

We conducted web searches to curate papers presenting event sequence systems published at HCI and visualization venues. A complete list of keywords is in the supplemental materials [1]. Additionally, we examined references from Peiris et al. [49] and Guo et al. [25] to ensure comprehensive coverage. We conducted a first pass through titles, abstracts, and keywords of papers from selected sources, identifying 105 papers that potentially met our criteria for analyzing sequential data.

After the first pass, we refined our selection by including papers that analyze event sequence data with at least one categorical attribute, excluding those with only numerical attributes [39] or time series data [6], as these types of datasets have different characteristics and analysis requirements compared to categorical event sequences. We read the full text of the marked papers to confirm they met our inclusion criteria, resulting in 58 papers from 16 venues. The supplemental materials include detailed information on the distribution of papers across venues and years. We also noticed that some systems were covered by multiple papers, such as EventAction [12, 13], LifeLines2 [64, 65]. Thus, our final list contained 52 systems.

3.2 Open Coding

In open coding stage, we first identified paper sections with sufficient information to describe low-level actions. These sections typically included system descriptions. The coders carefully read through each relevant section and split the text into micro parts, consisting of sentences or clauses that described a single user or system action. For example, a micro part could be "*pattern mining based on MDL*" or "*double-click the glyphs to expand them for detailed analysis*" [10].

Each micro part was then tagged with one or more labels based on the action(s) described. The coders used a preliminary set of labels derived from their domain knowledge [14, 54] and expanded the set as new actions emerged from the data. In our labeling, we paid special attention to two issues. First, a micro part could comprise more than one action and hence be associated with multiple labels. For instance, the micro part "*drilling down into a branch to get sub-pattern*" [42] involves two actions, filtering and segmenting. Second, a verb alone often could not capture the richness and multi-dimensional nature of an action, and we need to include information on the data components associated with each action. For instance, in "*user can sort page groups alphabetically, by volume, or difference. Sorting can be performed*

[1] All the supplemental materials are made available as an OSF project

within a specific step or across all steps" [72], while the fundamental action remains the same (i.e., sorting), the specific outcome of the action can vary significantly depending on the data elements being sorted (within a specific step or across all steps) and the conditions used to determine the sorting order (alphabetically, by volume, or difference).

While the bottom-up coding approach provides a detailed description of each analytic step, we recognize that data analysis is an emergent process [15] that cannot be effectively expressed by simply accumulating discrete steps. Therefore, we also considered the holistic objectives of each analysis, leading us to include high-level goals or objectives as a separate level in our framework, This dual perspective ensures our framework comprehensively represents the complex and emergent nature of data analysis in the context of event sequence visualization.

To reduce potential bias, three authors independently performed coding on the paper set, with a fourth author acting as a tie-breaker in case of disagreements. The open coding process was iterative and collaborative, involving weekly meetings held over six months to discuss and compare results. This ensured consistency and refinement of the coding scheme until all disagreements were resolved.

It is important to note we *focused primarily on actions that result in observable changes in data representation or visual display; leading to tangible transformations, such as the way data is processed, analyzed, or presented*. As such, actions that involve only cognitive processing in humans (e.g., saw, noticed, followed, observed, visually scanned for comparison) without any direct manifestation in the system are outside the scope of this work. This decision ensures the coded actions have a clear and traceable connection to system functionality.

We also omitted low-level interaction details on how an action is accomplished through user interface. For instance, zooming can be implemented using dedicated zoom controls (e.g., buttons, sliders, scale), double-clicking, or through brushing. These low-level features vary significantly across tools and are less critical for understanding high-level tasks and actions performed by users [70].

3.3 Axial Coding

The open coding resulted in the identification of four primary dimensions for each labeled micro part: *action*, *input*, *output*, and *criteria*. For example, in " ... *incorporate HMMs for exploring disease progression patterns from longitudinal health records*" [35], the action is *extract*, input is a set of sequences (health records), output is a set of latent patterns, and the criteria is Hiddden Markov Models or HMM. We then performed axial coding to group these labels into categories spanning three levels: techniques, strategies, and intents. We used affinity diagramming to organize the identified *action-input-output-criteria* quartets into `techniques`. Through this process, we formed clear definitions to ensure consistency, merged similar techniques, and split compound techniques into smaller units.

Techniques: each technique is characterized by the following four dimensions:
the *action* performed,
the *input* data or visual components acted upon,
the generated or modified *output* components, and
the *criteria* specifying the parameters or conditions associated with the action.

We organized techniques into groups based on similarities in nature of their *action*, *input* and *output* components, which led to the creation of higher-level abstractions called `strategies`. For example, 'identifying common patterns' and 'extracting latent patterns' in the data were grouped under the 'Summarize' strategy as they both produce patterns (*output*) from sequences (*input*), and provide a summary of the dataset.

While `techniques` and `strategies` answer 'how' each analysis step is performed, they do not shed light upon 'why' these steps are being executed. Therefore, we introduced an additional level of abstraction called `intents` by grouping related strategies based on the underlying motive. For instance, the `strategies` 'Summarize' and 'Aggregate' are grouped under the intent of 'Data Simplification' as they both seek to simplify event sequence data.

Finally, we identified overarching objectives that encompass the entire analysis process, called `objectives`. These `objectives` guide the design and development of strategies and techniques in the event sequence analysis tools. For instance, earlier tools like LifeFlow [67] and Lifeline [53] primarily facilitate pattern exploration, while subsequent tools such as MatrixWave [72] and Coco [45] focus on supporting cohort comparison. DPVis [35], on the other hand, specifically aims to model progression pathways. One system may cover multiple analysis goals, for example, the analysis goals in Sequence Synopsis [10] are temporal pattern exploration and correlation analysis.

4 Framework

Our framework comprises four levels of tasks (Fig. 1). At the highest level, the overarching analysis `objectives` capture six high-level goals that users aim to achieve: anomaly detection, stage progression, pattern exploration, prediction & recommendation, correlation & causality analysis, and cohort comparison.

Each objective involves realizing five main types of analysis `intents`: augment data, simplify data, configure data, configure visualization, and manage provenance.

Each intent can be accomplished through different `strategies`, depending on the data or visualization components involved and the analysis approach. For example, the intent to simplify sequence data can be achieved using aggregation, summarization, or segmentation.

At the most granular level, the framework defines `techniques`, which are the concrete, actionable analysis steps, such as creating new attributes, coalescing repeating events, or modifying summary patterns. Each technique can be described via a quartet of *action*, *input*, *output*, and *criteria*, which have been defined in Sec. 3.3.

Objectives

We identified six common high-level goals providing a broad categorization of `objectives` encompassing the majority of the papers:

Anomaly Detection: Identifying rare or unexpected patterns or events that deviate significantly from expected behavior. Example: healthcare [22]

Stage Progression: Capturing evolution of processes across distinct stages or phases within sequences, by segmenting sequences and uncovering trends, paths, and factors influencing the transition over time. Example: healthcare [35]

Pattern Exploration: Discovering common patterns in event sequence data through exploratory analysis. Example: healthcare [44], sports [68], clickstream [43], manufacturing [31]

Prediction & Recommendation: Modeling historical event sequence data to forecast future outcomes, and provide recommendations. Example: marketing [21], healthcare [34]

Correlation & Causality: Investigating dependencies, and potential causal links between events. Example: social media [69], healthcare [30]

Cohort Comparison: Analyzing the event sequences of different subpopulations to identify similarities, differences, and patterns specific to each cohort. Example: healthcare [44], social media [33]

Intents, Strategies, and Techniques

We now describe the `intents`, `strategies`, and `techniques` in our task framework. Fig. 1 lists the identified `intents` and `strategies`. Tab. 1 further elaborates on each technique along the *action, input, output,* and *criteria* dimensions. It is important to note that the list of `techniques` is **not exhaustive** and can be extended based on new systems and case studies. Depending on system design and implementation, the techniques may or may not involve a visualization display that dynamically updates with user action.

4.1 Augment Data

Data Augmentation refers to enhancing raw event sequence dataset by adding, modifying, or deriving new components, attributes, representations, or relationships that were not initially present in the dataset.

Table 1: The *action-input-output-criteria* quartet characterizing `techniques` in our multi-level task framework. Each technique is defined by the *action* performed, the *input* data components, the desired *output* components, and the *criteria* guiding the transformation.

Action	**Input**	**Output**	**Criteria**
Obtain	Event Sequence Data	Embeddings/ Projections/ Attributes	Statistical or machine learning algorithms, such as TF-IDF, dimensionality reduction technique (e.g., t-SNE)
Compute	Events, Sequences, Patterns	Similarity	Statistical or machine learning algorithms, such as distance metric (e.g., Jaccard Index, Levenshtein Distance)
Align	Set of Sequences	Aligned Sequences	Common reference point (e.g., origin time), specialized alignment procedures, (e.g., proxy event insertion, Dynamic Time Warping, shortest common supersequence)
Group	Set of Sequences	Grouped Sequences	Event: entry event, focal event of interest
			Event attribute: category, event attribute ranges
			Sequence attribute: categorical & continuous
			Derived: clustering, similarity to prototype
			Pattern: event subsequence (milestone)
	Set of Events	Event Hierarchy	Automated algorithms, existing hierarchical structures, regular expression or user-defined criteria
Mutate	Event, Event Attribute, Patterns, Sequence Attribute	Mutated sequences	User-defined scenario (e.g., altered event sentiment)
Generate	Sequence	Next possible event	Machine learning model (e.g., trained on historical event sequences)
Coalesce	Sequence	Simplified Sequence	Repeating events of same type, frequently occurring event bundle
Combine	Set of Sequences	Tree/DAG	Event and ancestor events
Calculate	Set of Sequences, Events, Event Attributes, Sequence Attributes	Statistical Summary (e.g., histograms, charts)	Statistical formulae
Extract	Set of Sequences	Common Patterns	Mining algorithms (e.g., VMSP, Frequent Sequential Pattern, PrefixSpan, Multiple Sequence Alignment)
		Latent Patterns	Statistical or machine learning algorithms (e.g., HMM, CP Decomposition, hierarchical Bayesian models)
Split	Sequence, Pattern	Sequence or Pattern Segments	Event (e.g., landmark events), pattern (e.g., milestones), derived latent stages (e.g., content vector segmentation)
			Time: fixed time intervals (e.g., year, month, day),
Modify	Summary Patterns	Modified Summary Patterns	User domain knowledge (e.g., add/delete/edit events, merge/split patterns), thresholds (e.g. minimum support, gap tolerances, number of hidden states)
Adjust	Current data processing and modelling specifications	Updated data processing and modelling specifications	Setting value to parameters, (e.g., window size, similarity threshold, importance measure, constraint/filter criteria)
Query	Set of Sequences, Patterns, Events, Event Attributes, Sequence Attributes	Query Output (filtered data)	Event Attribute: categorical event types, event attribute ranges, event occurrence time, intervals, duration, derived event attribute
			Sequence Attribute: categorical and continuous sequence attributes, derived sequence properties (e.g., total duration, event count)
			Pattern: subsequences, time intervals, keywords, transitions, presence/absence of event set
			Advanced querying techniques: graphical queries, regular expressions
Cross-Filter	Data (e.g., Events, Event Attributes, Sequence, Sequence Attributes, Patterns) glyph in one view	Filtered Data in coordinated views	Data attributes or dimension selection in other views
Display	Data (e.g., event, sequence, pattern) glyph	Detailed Information	Selected data component
Drill down - Roll up	Hierarchical data organization	Detailed View (Drill down) or Summary View (Roll up)	User selected granularity
Produce	Event Sequence Data	Visual Representation (e.g., timeline, graph, aggregate view, multivariate view)	Visual encoding rules (e.g., position, size, color, shape, opacity)
Customize	Visual Representation	Customized Visual Representation	User preferences (e.g., color palette, node size/shape, edge style, glyph design, visual variable mapping)
Update	Layout	Updated layout	System-specific control(e.g., comparison mode toggle, update canvas size)
Zoom	View	Zoomed View	Zoom level
Pan	View	Panned View	Pan direction
Highlight	Marks (e.g., events, patterns, sequences)	visually emphasized mark (e.g., change in color, opacity, border, glyph)	User selection
Select	Data (e.g., Event, Event Attributes, Sequence, Sequence Attributes, Patterns) glyph	Linked Highlighting	User selection and underlying data corrrespondence
Reposition	Set of Sequences	Repositioned Sequences	Alignment event
Reorder/ Sort	Events, Event Attributes, Sequence, Sequence Attributes, Patterns	Reordered/Sorted Components	Sorting criteria (e.g., frequency, alphabetical, temporal, correlation, prediction accuracy, attribute values, similarity, custom order)
Annotate	Data glyph	Annotated Representation	User notes, comments, descriptions
Save/ Record	Analysis State	Saved/Recorded Analysis	User-specified snapshot
Insert	Sequence	Sequence with Marker Event	User-defined timestamp

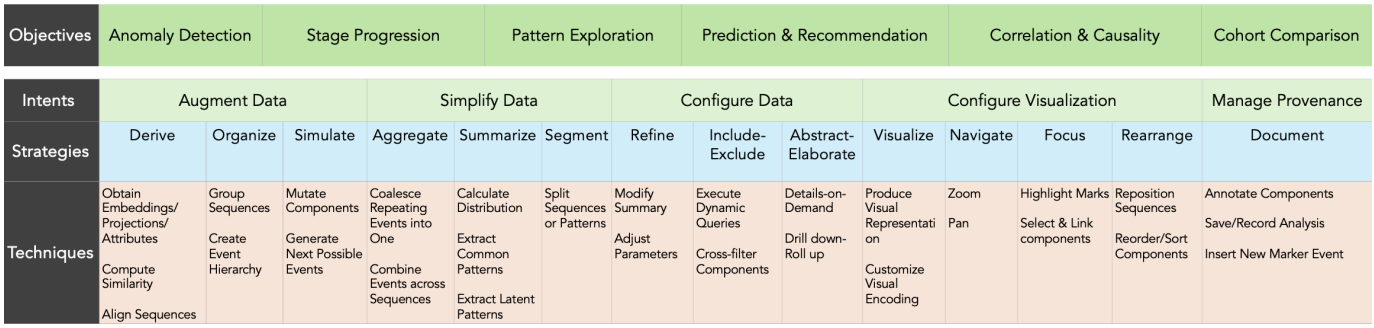

Figure 1: Our multi-level task framework consists of four hierarchical levels: `objectives`, `intents`, `strategies`, and `techniques`. We identify six overarching `objectives` of event sequence analysis. Each analysis step is associated with one of the five `intents`, depicting the purpose of the analysis step. Intents are realized through multiple `strategies`. Finally, a wide range of `techniques` are available for implementing each strategy.

4.1.1 DERIVE

Derive entails conducting diverse operations on event sequence data to compute additional attributes or representations necessary for downstream analysis. In the data preprocessing pipeline, multiple techniques can be sequentially employed to achieve this strategy.

`Obtain Embeddings/Projections/Attributes:` transforming raw event sequence data into a more structured, and analysis-ready format via augmenting embeddings or projections to facilitate subsequent modeling, and analysis. This often involves applying statistical or machine learning methods. For example EventThread [24] calculates TF-IDF and creates a three-way tensor representation of sequence data. ProtoSteer [46] calculates event and sequence embedding, and employs t-SNE to project the high-dimensional sequence embedding into a lower-dimensional space.

`Compute Similarity:` quantifying the similarity or proximity between events, sequences, or patterns via distance metrics. For example, Sequence Synopsis [10] uses Jaccard Index to calculate event similarity, and Levenshtein distance to calculate pattern similarity.

`Align Sequences:` establishing correspondences between events across multiple sequences based on a common reference point or shared semantic meaning. Event sequence alignment is crucial when dealing with variable-length sequences, inconsistent event orders, or gaps. Event sequence tools typically align sequences using a common origin time, such as first event. Some tools use specialized alignment algorithms, such as EventThread2 [23] uses Dynamic Time Warping, Sequence Braiding [5] finds the Shortest Common Supersequence.

4.1.2 ORGANIZE

ORGANIZE refers to assembling and grouping similar sequences or events based on specific criteria or attributes.

`Group Sequences:` organizing event sequences into distinct groups or clusters. *Criteria* for grouping include:

events, e.g., group sequences by entry event (CoreFlow [42], EventFlow [47]); event subsequences (milestones) or focal events of interest (GapFlow [16])

event properties, to support attribute-based analysis, e.g., group by categorical event types (MatrixWave [72]) or partitioning continuous event attributes into meaningful ranges (Segmentifier [11]).

sequence-level metadata, such as demographics or other non-temporal attributes to support analyzing population subgroups. Common methods include grouping by sequence attributes: both categorical like gender and numerical like age (TipoVis [26]); or derived sequence properties like total duration, event count (SessionViewer [36]).

clustering on derived attributes, involves attribute-based grouping derived as discussed in 4.1.1. Sequence Synopsis [10] clusters sequences mapping to the same pattern. EventThread [24], groups sequence segments by cluster similarity.

`Create Event Hierarchy:` combining available event types into higher-level categories or meta-events based on semantic similarity, hierarchical relationships, or user-defined criteria. The definition of event categories can vary based on criteria, such as: automated algorithms (e.g., Cadence [19]), existing hierarchical structures (e.g., MatrixWave [72], Segmentifier [11]), regular expression (e.g., Eventpad [8]) or user-defined criteria (e.g., Patterns and Sequences [43], Eloquence [63]).

4.1.3 SIMULATE

Simulate refers to generating hypothetical or predictive scenarios, enabling users to analyze the potential outcomes, dependencies, and implications of different events or patterns.

`Mutate Components:` altering specific components or properties to simulate and analyze alternative scenarios, enabling what-if analysis. In OpinionFlow [69], the impact of changing event attributes can be observed on downstream events via adjusting tweet sentiment.

`Generate Next Possible Events:` predicting the likelihood of future events based on patterns observed in historical event sequences through machine learning models. Generating next possible events enables analysts to anticipate future outcomes, plan for potential scenarios, and make data-driven decisions (OpinionFlow [69]).

4.2 Simplify Data

Data Simplification refers to reducing the complexity or scale of an event sequence dataset to make it more amenable to automated processing or human inspection.

4.2.1 AGGREGATE

Aggregate involves replacing multiple events with a single representative event, reducing the number or type of events in the visualization.

`Coalesce Repeating Events into One:` merging consecutive events within a sequence into a single representative event. The merging can be done by selecting one event as representative and discarding duplicates (e.g., Segmentifier [11]), or by creating a new event that encompasses the duration or attributes of the merged events (e.g., EventFlow [47]).

`Combine Events across Sequences:` aggregating similar events across multiple sequences and organizing them into a hierarchical or graph-based structure, such as a tree (e.g., EvenFlow [47]) or a directed acyclic graph (DAG) (e.g., Sankey diagram). This transformation provides a more compact representation of the event sequences.

4.2.2 SUMMARIZE

Summarize involves generating a concise overview of the entire dataset or its selected subsets. The process includes computing statistical distributions to present the data in an easily understandable form or applying algorithms to extract representative patterns. The primary goal of summarization is to provide a high-level understanding of the key characteristics and statistical properties without being overwhelmed by the full complexity of the data.

`Calculate Distribution:` apply statistical methods to form mathematical summaries of event sequences, enabling both confirmatory and

exploratory analysis. These summaries encompass metrics on distributions and probabilities over event sequence attributes. Examples include histograms (e.g., Segmentifier [11], Cadence [19]), type distributions (e.g., SessionViewer [36]), and inter-arrival times between event types (e.g., EventFlow [47])

`Extract Common Patterns:` algorithmically extracting frequently occurring subsequences based on events that are present in the data. Examples include temporal pattern mining techniques, such as recursive branching pattern mining (e.g., CoreFlow [42]), VMSP (e.g., Patterns and Sequences [43], MAQUI [38]), Frequent Sequential Pattern (e.g., SentenTree [28]), PrefixSpan (e.g., Eloquence [63]), or Multiple Sequence Alignment (e.g., EventPad [8]).

`Extract Latent Patterns:` using statistical or machine learning methods to discover hidden sequential structures and trends implicitly embedded within event sequences. Examples include probabilistic methods, such as Hidden Markov Models (e.g., DPVis [35]), Algorithms like hierarchical Bayesian Rose Tree models (e.g, OpinionFlow [69]) to group sequences with similar semantic content.

4.2.3 SEGMENT

Segment refers to breaking down lengthy sequences into smaller, more manageable segments based on specific criteria or temporal boundaries.

`Split Sequences or Patterns:` decomposing event sequences into meaningful subsequences or segments. *Criteria* for splitting include:

temporal folding, dividing large-scale event sequence data collected over extended periods into segments of fixed time intervals, such as a year, a month, or a day) (e.g., EventThread [24], STBins [55]).

key event, dividing sequences into pre- and post-event segments to analyze antecedent and sequelae patterns relative to the key event (e.g., MAQUI [38], EventThread [24]), or segmenting sequences based on the presence or absence of a key event or pattern to compare alternative trajectories (e.g., DecisionFlow [17], MAQUI [38])

derived attributes, such as content vector segmentation (e.g., EventThread2 [23]).

dynamic splitting of segments with a shared pattern (e.g., CoreFlow [42], Patterns & Sequences [43]).

4.3 Configure Data

Configuring Data comprise modifying and experimenting with different components, properties, and granularity of an event sequence dataset for exploration, aiming to identify the optimal amount of data required for subsequent analysis.

4.3.1 REFINE

Refine refers to iteratively improving or fine-tuning the analysis data based on domain knowledge, or evolving requirements. Refinement enables progressive adjustments of analysis outcomes, incorporating user expertise and insights to obtain precise and meaningful results.

`Modify Summary:` supporting modifications and calibrations of system-generated summarized representations or patterns, enabling users to incorporate their domain knowledge. Users can iteratively fine-tune the automatically generated summaries until they are satisfied with the representation. Examples of modification operations include adding/deleting/editing events in patterns (e.g., ProtoSteer [46]), and merging/splitting patterns (e.g., EventThread2 [23]). Summaries produced by extracting common patterns (4.2.2) can be calibrated using minimum support thresholds and gap tolerances. Similarly, summaries generated from extracting latent patterns (4.2.2) can be customized by controlling the number of hidden states.

`Adjust Parameters:` fine-tuning parameters that affect data processing and modeling, without directly manipulating the visual representation itself. Users can adjust various parameters, including temporal settings (e.g. window size or duration in STBins [55]), grouping settings (e.g., similarity measure thresholds in EventThread [24]), aggregation settings (e.g., importance measures for grouping events in Cadence [19]), constraint/filtering parameters (e.g., toggling rules in EventPad [8] or modifying constraints in Eloquence [63]).

4.3.2 INCLUDE-EXCLUDE

Include-Exclude refers to controlling the subset of event sequence data under consideration, enabling users to focus on relevant subsets and eliminate irrelevant information during analysis.

`Execute Dynamic Queries:` refer to applying filters, selections, or search criteria to retrieve matched subset. Dynamic queries span various components of event sequence data and visual elements, including filtering by event types, categories, or attributes (e.g., Cadence [19], EventPad [8]), selecting specific subsequences, stages, or time intervals (e.g., DPVis [35], Segmentifier [11]), searching for keywords or patterns (e.g., ProtoSteer [46], Eloquence [63]), filtering based on sequence attributes (e.g., Sequence Braiding [5], Eloquence [19]), filtering based on transitions (e.g., DPVis [35]), querying for the presence or absence of specific events, milestones, or patterns (e.g., Eloquence [63]), and applying advanced querying techniques such as graphical queries, or regular expressions (e.g., Cadence [19], EventFlow [47]).

`Cross-filter Components:` applying filters in one view and automatically updating other coordinated views. By applying filters or selections in one context and observing the immediate impact on related contexts, analysts can gain insights into how different data attributes or dimensions correlate. Examples include cross-filtering via interactions (OpinionFlow [69], MAQUI [38]) or query languages (DPVis [35]).

4.3.3 ABSTRACT-ELABORATE

Abstract-Elaborate involves interactively adjusting the level of detail, granularity, or abstraction of event sequence data representation. This strategy enables dynamic exploration of different levels of abstraction, from high-level overviews to detailed, fine-grained representations.

`Details-on-Demand:` providing users with additional information about specific event sequence components upon interaction. This technique helps analyst gain insights into specific aspects while maintaining the overall context. This information can be presented in various forms, such as: tooltips (e.g., Segmentifier [11]), sidebar (e.g., STBins [55], OutFlow [66]), separate information panels (e.g., Sequence Braiding [5], EventPad [8]) or Expanded views (e.g., DPVis [35]).

`Drill down-Roll up:` supporting data navigation at various levels of detail. transitioning between detailed, lower-level representations (drill down) and summarized, higher-level abstractions (roll up). For example, Lifeline [53] provides semantic zooming to expand desired facets (low-level abstraction), and use silhouettes and shadows when full details can't be shown on the overview (high-level abstraction). The technique is facilitated by underlying hierarchical data organization. Implementation ideas include showing or hiding labels or annotations (e.g., Eloquence [63], OpinionFlow [69]), creating a new context view (e.g., EventPad [8], Patterns and Sequences [43]) or modifying the layout (e.g., LifeLines [53]) to accommodate new level of detail.

4.4 Configure Visualization

Configuring visualization refers to modifying and adjusting different visual properties, and layouts of an event sequence visualization to explore alternative representations, and enhance the communication of insights. Visual configuration allows users to tailor the visualization to their perceptual preferences and storytelling needs.

4.4.1 VISUALIZE

Visualize refers to applying or updating rules that map data to visual marks or mark properties, creating graphical representations.

`Produce Visual Representation:` converting sequences and associated attributes into visual forms that effectively communicate patterns, relationships, and insights. The generated visual representations leverage various visual elements, such as timelines, graphs, charts, glyphs, and diverse color schemes, to convey the structure, temporal dynamics, and characteristics of the event sequences.

The data elements are mapped to visual channels such as position, size, color, shape, or opacity. The choice of visual encodings depends on the nature of the data, the analysis goals, and the desired level of detail or abstraction. A detailed description of all kinds of visual

representation in beyond the scope of this work, and has been covered extensively in prior works [29, 73].

`Customize Visual Encoding:` directly modifying visual representations of event sequences and associated attributes. The primary focus is enhancing readability, interpretability, and communication of insights by aligning visual representations with user preferences or analysis goals.

Customization may include visual encoding adjustments such as line thickness or spacing (e.g., sequence braiding [5]), selecting different visual designs (e.g., ActiviTree [62], STBins [55]), adjusting transparency levels (e.g., Eloquence [63]), changing canvas size (e.g., CoreFlow [42]), selecting color palettes (e.g., Eloquence [63]), defining node shapes, edge styles, or glyph designs.

`Update Layout:` adapting the presentation mode of an event sequence visualization to accommodate diverse analysis requirements. It involves adjusting layout parameters, such as canvas dimensions (e.g., CoreFlow [42]) or visualization modes (e.g., (s|q)ueries [71], ProtoSteer [46]).

4.4.2 NAVIGATE

Navigate depicts interactively exploring and modifying viewport or visible area of a visualization to focus on specific regions or time periods.

`Zoom:` dynamically adjusting the magnification level within a single, consistent view. Zooming enables multi-scale exploration to ensure readability and clarity of the displayed information. Visualization tools may employ animated transitions or progressive loading to enhance the zooming experience. Users can initiate the zooming action through dedicated zoom controls (e.g., MatrixWave [72], LifeLines2 [64])

`Pan:` interactively exploring different parts of the displayed event sequences by shifting the viewport. Panning enables inspection of lengthy or complex sequences in a continuous and fluid manner without losing context. Usually, horizontal panning shifts the visible area along the time axis, revealing the temporal progression and relationships among events (e.g., LifeLines2 [64], MatrixWave [72]). Some tools also support vertical panning, especially when dealing with a large number of sequences (e.g., EventPad [9]).

4.4.3 FOCUS

Focus denotes visually emphasizing specific elements, patterns, or subsets of interest, guiding user attention towards the emphasized element.

`Highlight Marks:` emphasizing or accentuating specific components of interest within the visualization. This involves applying distinct visual attributes or effects to the selected marks, such as changing color (e.g., SentenTree [28]), opacity (e.g., MatrixWave [72], SentenTree [28]), adding border (e.g., Lifelines [53], Patterns and Sequences [43]), or using visual cues like glyphs (e.g., TipoVis [26]) or annotations.

`Select & Link Components:` synchronized selection and highlighting of elements across different views. By visually connecting related elements across views, linked selection helps users gain a holistic understanding. For example, in Sequence Synopsis [10] selecting a pattern in the summary view highlights individual sequences containing the pattern in the detailed view. In DecisionFlow [17], selection of an event in other panels remain consistent across pattern panel.

4.4.4 REARRANGE

Rearrange depicts changing the spatial arrangement or positioning of visual items within the visualization layout. It involves modifying the order of visual elements to improve overall readability and interpretability.

`Reposition Sequences:` changing the visual arrangement by centering sequence layout around a specific event of interest, facilitating exploration of temporal relationships before and after the alignment point. Repositioning can help uncover insights that may be obscured in default configuration. This technique is often accompanied by an animated transition for smooth user experience (e.g., Sequence Synopsis [10]).

`Reorder/Sort Components:` reordering the displayed components, in ascending or descending order. Sorting across multiple attributes is possible, with primary criteria taking precedence and subsequent ones breaking ties or refining the ordering. Sorting can be performed on subsets, such as sorting sequences within a cluster, events within a time range, or attributes within a category.

There is a wide range of sorting criteria, including, frequency (LifeFlow [67]), number of occurrences (TipoVis [26]), rarity (EventPad [8]), similarity (Sequence Synopsis [10]), alphabetical order (MatrixWave [72]), average time to previous event (LifeFlow [67]), correlation with outcome (Cadence [19]), accuracy or performance (ProtoSteer [46]), attribute values (quantitative) (Sequence Braiding [5]), attribute name (qualitative) (TipoVis [26]), custom (MatrixWave [72]).

4.5 Manage Provenance

Provenance encapsulates tracking, recording, and managing the history of data states, insights, and actions throughout exploration and analysis. It involves documenting applied transformations and their validations, result interpretations, and decision-making processes. The aim is to ensure transparency, reproducibility, and trustworthiness of analysis results by providing a comprehensive trail of the steps taken, assumptions made, and conclusions drawn.

4.5.1 DOCUMENT

Documenting involves providing the means to create, manage, and organize insights or metadata associated with analysis.

`Annotate Components:` providing mechanisms to add textual notes, comments, or descriptions to specific elements, subsets, or regions of the visualized data. This allows users to document their findings, insights, hypotheses, or any relevant information discovered during the analysis process. Annotation facilitates collaboration, knowledge sharing, enabling future reference (e.g., SessionViewer [36]).

`Save/ Record Analysis:` storing specific snapshots of the analysis process for future reference or further investigation. Save/Record enables iterative analysis, and ensures reproducibility. This technique enables users to create a record of their analytical progress (e.g., Segmentifier [11], ProtoSteer [46]), key findings, or interesting subsets of the data (e.g., EventPad [8], DPVis [35]),as well as design preferences (e.g., Eloquence [63]) facilitating the ability to resume the analysis at a later time, or compare different stages of the exploration.

`Insert New Marker Event:` manual insertion of new events at specific points within existing event sequences. These user-defined events, known as marker events, serve as reference points deemed significant by the users for analysis or interpretation (e.g., EventFlow [47], GapFlow [16]). By creating new marker events, users can enrich the event sequences with additional contextual information.

5 Evaluation of the Framework

In this section, we evaluate existing task taxonomies through case studies. We then discuss future work to investigate the framework's evaluative and generative powers.

5.1 Descriptive Power

To showcase the descriptive power of our multi-level task framework, we apply our framework and three existing frameworks [14, 49, 54] to analyze real-world case studies reported in the literature. We use the following criteria to choose the case studies: 1) they should cover different application domains and analysis objectives, 2) are preferably published in recent years to reflect current challenges, and 3) are preferably not included in our coding and derivation of the framework. Based on these criteria, we chose the following three case studies: **C1**: causality in electronic health records from SeqCausal [30], **C2**: tennis tactics analysis from RASIPAM [68], **C3**: neonatal data similarity analysis from FlexEvent [41].

C1 and **C3** both were conducted in the healthcare domain, but they address different `objectives`: "Correlation & Causality" (**C1**) vs. "Pattern Exploration" (**C3**). Conversely, **C2** and **C3**, though addressing the same `objective` ("Pattern Exploration"), come from different domains: healthcare (**C3**) and sports analytics (**C2**). All the selected case studies were published in the last three years: **C1** (2021), **C2** (2022), and **C3** (2023). Among these three, only the paper for **C1** is included in our original corpus, while papers for **C2** and **C3** are not part of the original corpus. The total number of tasks is 25 across the case studies

Table 2: Comparative analysis of task mappings for three case study excerpts from SeqCausal [30], RASIPAM [68] and FlexEvent [41] using our multi-level task framework and existing task taxonomies (Plaisant et al. [54], Du et al. [14], Peiris et al. [49]). The color scheme indicates the level of alignment: no match and partial match. Our framework demonstrates more comprehensive mapping compared to existing taxonomies

Tasks	**C1.T1** [30]	**C1.T3** [30]	**C1.T7** [30]	**C2.T7** [68]	**C2.T9** [68]	**C3.T7** [41]
Excerpt	*The doctors queried a group of 127 middle-aged patients aging from 50 to 60 who were diagnosed with pneumonia.*	*After several iterations of confirming causalities and model updates, …*	*The doctors saved the final causality to the analysis history view.*	*E2 re-ranked the tactics in Tactic View based on the tactical importance …*	*Experts applied this merging adjustment and obtained a more accurate estimate of the win rate for this serving tactic.*	*Changing the color attribute to sepsis, …*
Plaisant et al. [54]	*Prepare or select data for further study* Identify a set of records of interest	*Prepare or select data for further study* Review data quality and inform choices to be made in order to model the data	n/a	*Prepare or select data for further study* Identify a set of records of interest	*Prepare or select data for further study* Review data quality and inform choices to be made in order to model the data	n/a
Du et al. [14]	*Extraction Strategies* Goal-Driven Record Extracting	n/a	n/a	n/a	n/a	n/a
Peiris et al. [49]	action: Filter target: Event Sequences criteria: Metadata Attributes	action: Derive Metrics target: n/a criteria: n/a	action: Annotate target: n/a criteria: n/a	action: Sort/Rank target: Event Sequences criteria: Metrics/Features	action: Add/Modify target: Event Sequences criteria: Metrics/Features	n/a
Ours	Intent: Configure Data Strategy: Include-Exclude Technique: Execute Dynamic Queries action: Query input: Event Sequences output: Filtered Event Sequences criteria: Age	Intent: Configure Data Strategy: Refine Technique: Adjust Parameters action: Adjust input: Current causal model output: Updated causal model criteria: Domain knowledge	Intent: Manage Provenance Strategy: Document Technique: Save/Record Analysis action: Save/Record input: Analysis State output: Saved/Recorded Analysis criteria: User-specified snapshot	Intent: Configure Visualization Strategy: Rearrange Technique: Reorder/Sort Components action: Reorder/Sort input: Tactics output: Reordered Tactics criteria: Tactical importance metric	Intent: Configure Data Strategy: Refine Technique: Modify Summary action: Modify input: Tactics output: Modified Tactics criteria: Domain knowledge	Intent: Configure Visualization Strategy: Visualize Technique: Produce Visualization action: Produce input: Event sequence data output: Visual representation criteria: Visual encoding rules

For each case study, we coded the analysis process using our framework and three existing frameworks. Two authors independently performed the coding for each case study across all four frameworks, then compared results and reached a consensus through discussion. This thorough examination demonstrates our framework's ability to comprehensively describe and characterize complex tasks and workflows. Here we present some key examples from the case studies. The full coding is added to the supplemental materials.

5.1.1 Takeaways from Three Case Studies

The comparative analysis of task mapping across the three case studies reveals several key differences among the approaches. We present six example mappings from the case studies in Tab. 2.

Granularity and Specificity: Plaisant et al.'s taxonomy [54] provides only high-level descriptions of tasks, lacking low-level details that capture individual steps of analysis workflows. This limitation becomes evident when diverse low-level actions, such as grouping, sorting and filtering, are required to accomplish the same high-level task, e.g., 'Identify a set of records of interest ' (**C1.T1, C2.T7**).
Similarly, Peiris et al.'s taxonomy [49] exhibits limitations in granularity for certain tasks, indicating the need for a more comprehensive and precise task framework. For example, both updating a model and performing dimensionality reduction tasks are mapped to action 'Derive Metrics' (**C1.T3**). This mapping fails to capture the distinct nature of these tasks, as updating a model involves refining an existing model, while dimensionality reduction focuses on projecting data to a lower-dimensional space. In contrast, our framework captures this complexity through two levels of abstraction: 'Refine' as strategy, and 'Adjust Parameters' as technique for model update, and 'Derive' as strategy and 'Obtain Embeddings/Projections/Attributes' as technique for dimensionality reduction (**C1.T3**). These mappings align more closely with the actual tasks performed in the respective case studies.

Comprehensiveness: Du et al.'s strategies [14] primarily addresses tasks related to reducing volume and variety, limiting their applicability to the full range of tasks involved in event sequence analysis. This narrow scope is apparent: tasks such as updating models, managing provenance, grouping sequences and refinement of the analysis are not captured by their strategies (**C1.T3-C3.T7**).
In contrast, our framework captures tasks that are not adequately represented in the other taxonomies, showcasing its compatibility in covering the end-to-end tasks involved in event sequence analysis.

However, It is important to acknowledge our technique list is not exhaustive, leaving room for extension. This is evident in **Task C3.T6 in the supplemental materials (Table 3)**, where our framework identifies only a partial match. In this specific task, the grouping operation is being conducted on projection embeddings rather than event sequences. While our framework correctly identifies the action as 'Group', it does not provide a perfect match for the input of the grouping operation.

Triplet vs. Quartet: Peiris et al. [49] mention *criteria* as a data component, limiting its ability to fully address the *how* aspect of task characterization. In contrast, *criteria* in our framework may capture both the methods and conditions of an action, offering a more complete characterization of how tasks are performed. Additionally, their *action-target-criteria* triplet lacks an *output* component, obscuring important details about the results. For instance, when analyzing summary patterns, it is crucial to discern whether these patterns are directly observable or represent latent structures requiring further interpretation. Our framework includes the output component, precisely distinguishing latent and common patterns. Another notable difference is the handling of *targets* and *criteria*. Peiris et al. categorize four target types and five criteria types. However, we believe that *input* (analogous to *targets*) and *criteria* are too expansive to be limited to a predefined set of elements.

The comparative analysis of the task taxonomies across the three case studies highlights the strengths of our multi-level task framework. By encompassing a broad spectrum of tasks, our framework enables a holistic tracing of the event sequence analysis workflow.

5.2 Evaluative Power

Our framework lays the foundation for structured evaluation of the effectiveness and completeness of event sequence analysis tools across diverse domains. Researchers can compare their capabilities by aligning the supported `objectives`, `intents`, `strategies`, and `techniques` of each tool to our framework levels, identifying commonalities, differences, and potential gaps. This comparative analysis can reveal

strengths and limitations of individual tools and highlight areas for improvements. Moreover, the framework can provide a base for defining evaluation criteria and metrics to assess performance and usability.

Further research and validation are necessary to fully establish the framework's evaluative capabilities. While the framework provides a comprehensive structure for assessment, its effectiveness in practice needs to be demonstrated through case studies and applications in diverse domains, which are beyond the scope of this paper.

5.3 Generative Power

Similar to Brehmer and Munzner's typology [7], our framework has the potential to guide practitioners through the 'discover' and 'design' stages of design studies [58]. Practitioners can translate their domain problems into abstract task descriptions by utilizing the `objectives` outlined in our framework, therefore identifying high-level analysis goals that need to be supported.

The hierarchical nature of our framework also allows practitioners to focus on high-level `intents` without worrying about low-level details. For example, if the dataset is large and complex, the initial analysis steps may require "Data Simplification" to make the data more manageable. Similarly, if the analysis goal is "Prediction & Recommendation", practitioners should consider if data augmentation is necessary first.

Once an intent has been identified, our framework offers an array of `strategies` to support these `intents` based on data characteristics. For instance, both 'Include-Exclude' and 'Abstract-Elaborate' strategies cater to the 'Configure Data' intent, yet they serve different purposes. This flexibility enables practitioners to tailor the data configuration to the specific demands of the analysis step. In addition, if practitioners have already started exploring certain strategies without considering the higher-level `intents`, our framework can be useful for them to examine alternative `strategies` that fulfill the same intent in different ways.

Finally, based on the input and desired output of each analysis step, practitioners can choose the *specific* action and *criteria* from the `techniques` defined in our framework. The *action-input-output-criteria* quartet provides a structured approach to selecting the most suitable technique for each analysis step.

We plan to enhance the generative power of our framework by developing a formal specification in future. This structured representation will enable automated generation of task-specific design recommendations and validation of event sequence analysis tool designs.

6 Limitations

While our framework provides a structured examination of event sequence visual analytic tasks, it is important to acknowledge certain limitations. First, the framework is derived from existing research and aims to capture a broad spectrum of tasks supported by various systems. However, we do not claim it to be exhaustive. As research advances, the framework will need to be revised and enhanced, incorporating additional strategies and techniques. The taxonomy should be viewed as a living document to be expanded and refined as new methods emerge.

Second, the boundaries between categories in any classification effort can be fuzzy and subject to multiple interpretations. For instance, at the `objectives` level, "Stage Progression" can be interpreted as a type of pattern, and may be subsumed under "Pattern Exploration". We decided to keep these two categories separate because "Stage Progression" involves segmenting sequences and tracking the progression *across different segments*. In contrast, "Pattern Exploration" focuses on identifying recurring patterns *across sequences* that may not involve segmentation. At the `intents` level, categories such as "Configure Data" and "Configure Visualization" can overlap significantly. For example, zooming in on a visualization not only adjusts the visual representation but also acts as a filter on the underlying data. This overlap demonstrates the interconnected nature of data manipulation and visualization configuration, posing challenges in distinct categorization.

In the same light, the distinction between "Simplify Data" and "Configure Data" can be blurry, as simplification can be seen as a subset of configuration. We differentiate these two intents because data reduction is a crucial aspect of event sequence analytics [73]. Strategies listed under "Simplify Data" focus on performing data reduction computations, while "Configure Data" is about dynamically choosing which aspects of data should be the focus of the current inspection.

The distinction between "Aggregate" and "Summarize" `strategies` is another example of potential contention. "Aggregate" involves merging multiple events without reducing the number of sequences or unique events. In contrast, "Summarize" focuses on extracting representative subsequences or patterns, potentially creating a lossy presentation by excluding rare events. The boundary between these `strategies` can be subtle, as one may argue that aggregation is one way to summarize.

At the `techniques` level, the term "Alignment" is used in the literature to refer to both visual and data configuration. In our framework, alignment falls under the "Augment Data" intent, focusing on data alignment operations such as establishing correspondences between events across multiple sequences based on a common reference point. This is conceptually similar to sequence alignment in fields like genomics [4, 40]. Nevertheless, alignment can also be a visual operation, involving repositioning of event sequences in a visualization (e.g., [43, 47]). To avoid confusion, we have used the term "Reposition Sequences" under "Configure Visualization" intent to describe such visual alignment actions. While this distinction helps maintain clarity within our framework, we acknowledge that our definition of the term may differ from its typical usage in visualization literature.

These complexities underscore the evolving nature of our framework and the inherent challenges in categorizing diverse strategies and techniques. While we strive to provide a structured framework, some aspects defy clear-cut classification.

Third, our literature review was extensive but did not follow a formal systematic review process like PRISMA [48] in the healthcare domain, which ensures rigorous and reproducible examination. Future work could benefit from adopting such systematic approaches to minimize the risk of gaps and biases.

Despite these limitations, we believe our framework takes an important step towards advancing the understanding of event sequence analysis tasks, setting the stage for future research and innovation.

7 Discussion and Future Work

Our framework establishes a theoretical basis for characterizing a diverse range of tasks in event sequence analysis. Its hierarchical organization, from high-level `strategies` to low-level `techniques`, enables domain adaptation and knowledge transfer, by providing a common vocabulary for describing and comparing tasks across applications.

A promising direction for future work involves developing formal specifications based on our framework. Task specifications hold potential for constructing automated analysis pipelines and provenance. Formal specifications can facilitate creation of intelligent assistance tools that suggest appropriate `techniques` and effective analysis workflows based on analysis `objectives` and dataset characteristics.

Another consideration is integrating data properties in the framework. Data characteristics, such as size, complexity, and heterogeneity, can impact both the choice of high-level intents and strategies as well as the performance of low-level operations. By analyzing the implications of data properties on these operations, researchers can develop more targeted and efficient analysis strategies.

These ideas for future work can pave the path towards creating executable benchmarks for tool evaluation [52]. By formalizing existing case studies along with associated `objectives` and `techniques`, researchers can develop standardized datasets and evaluation protocols.

8 Conclusion

We present a multi-level framework to describe the `objectives`, `intents`, `strategies`, and `techniques` in event sequence visual analytics. Compared to existing event sequence task taxonomies, our framework enables more precise descriptions of tasks at multiple levels of granularity, and along multiple dimensions such as the *action*, *input*, *output*, and *criteria* associated with each technique. The framework has the potential to promote knowledge transfer and generalization across domain-specific investigations, and lays the foundation for future research on formal specification languages and analysis strategy recommendations for event sequence data.

REFERENCES

[1] J.-w. Ahn, C. Plaisant, and B. Shneiderman. A task taxonomy for network evolution analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):365–376, 2014. doi: 10.1109/TVCG.2013.238 2

[2] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pp. 111–117, 2005. doi: 10.1109/INFVIS.2005.1532136 1

[3] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag, Berlin, 2005. doi: 10.1007/3-540-31190-4 1

[4] A. APOSTOLICO and R. GIANCARLO. Sequence alignment in molecular biology. *Journal of Computational Biology*, 5(2):173–196, 1998. PMID: 9672827. doi: 10.1089/cmb.1998.5.173 9

[5] S. Bartolomeo, Y. Zhang, F. Sheng, and C. Dunne. Sequence braiding: Visual overviews of temporal event sequences and attributes. *IEEE Transactions on Visualization and Computer Graphics*, 27(02):1353–1363, 2021. doi: 10.1109/TVCG.2020.3030442 1, 5, 6, 7

[6] J. Bernard, T. Ruppert, M. Scherer, T. Schreck, and J. Kohlhammer. Guided discovery of interesting relationships between time series clusters and metadata properties. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*. ACM, New York, 2012. doi: 10.1145/2362456.2362485 2

[7] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, 2013. doi: 10.1109/TVCG.2013.124 1, 2, 9

[8] B. C. Cappers, P. N. Meessen, S. Etalle, and J. J. van Wijk. Eventpad: Rapid malware analysis and reverse engineering using visual analytics. In *2018 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pp. 1–8, 2018. doi: 10.1109/VIZSEC.2018.8709230 5, 6, 7

[9] B. C. Cappers and J. J. van Wijk. Exploring multivariate event sequences using rules, aggregations, and selections. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):532–541, 2018. doi: 10.1109/TVCG.2017.2745278 1, 7

[10] Y. Chen, P. Xu, and L. Ren. Sequence synopsis: Optimize visual summary of temporal event data. *IEEE Transactions on Visualization and Computer Graphics*, 24(01):45–55, 2018. doi: 10.1109/TVCG.2017.2745083 2, 3, 5, 7

[11] K. Dextras-Romagnino and T. Munzner. Segmentifier: Interactive refinement of clickstream data. *Computer Graphics Forum*, 38(3):623–634, 2019. doi: 10.1111/cgf.13715 1, 5, 6, 7

[12] F. Du, C. Plaisant, N. Spring, K. Crowley, and B. Shneiderman. Eventaction: A visual analytics approach to explainable recommendation for event sequences. *ACM Transactions on Interactive Intelligent Systems*, 9(4):1–31, 2019. doi: 10.1145/3301402 2

[13] F. Du, C. Plaisant, N. Spring, and B. Shneiderman. Eventaction: Visual analytics for temporal event sequence recommendation. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 61–70, 2016. doi: 10.1109/VAST.2016.7883512 2

[14] F. Du, B. Shneiderman, C. Plaisant, S. Malik, and A. Perer. Coping with volume and variety in temporal event sequences: Strategies for sharpening analytic focus. *IEEE Transactions on Visualization and Computer Graphics*, 23(06):1636–1649, 2017. doi: 10.1109/TVCG.2016.2539960 1, 2, 7, 8

[15] F. Emmert-Streib, S. Moutari, and M. Dehmer. The process of analyzing data is the emergent feature of data science. *Frontiers in Genetics*, 7, 2016. doi: 10.3389/fgene.2016.00012 3

[16] D. Gotz, N. Cao, E. Goldbraich, and B. Carmeli. Gapflow : Visualizing gaps in care for medical treatment plans. `https://gotz.web.unc.edu/research-project/gapflow/`, 2013. 5, 7

[17] D. Gotz and H. Stavropoulos. Decisionflow: Visual analytics for high-dimensional temporal event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1783–1792, 2014. doi: 10.1109/TVCG.2014.2346682 1, 6, 7

[18] D. Gotz and K. Wongsuphasawat. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668, 2012. doi: 10.1109/TVCG.2012.225 1

[19] D. Gotz, J. Zhang, W. Wang, J. Shrestha, and D. Borland. Visual analysis of high-dimensional event sequence data via dynamic hierarchical aggregation. *IEEE Transactions on Visualization and Computer Graphics*, 26(01):440–450, 2020. doi: 10.1109/TVCG.2019.2934661 1, 5, 6, 7

[20] D. Gotz and M. X. Zhou. Characterizing users' visual analytic activity for insight provenance. *Information Visualization*, 8(1):42–55, 2009. doi: 10.1057/ivs.2008.31 2

[21] S. Guo, F. Du, S. Malik, E. Koh, S. Kim, Z. Liu, D. Kim, H. Zha, and N. Cao. Visualizing uncertainty and alternatives in event sequence predictions. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, p. 1–12. ACM, New York, 2019. doi: 10.1145/3290605.3300803 3

[22] S. Guo, Z. Jin, Q. Chen, D. Gotz, H. Zha, and N. Cao. Visual anomaly detection in event sequence data. In *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019. doi: 10.1109/bigdata47090.2019.9005687 3

[23] S. Guo, Z. Jin, D. Gotz, F. Du, H. Zha, and N. Cao. Visual progression analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 25:417–426, 2018. doi: 10.1109/TVCG.2018.2864885 5, 6

[24] S. Guo, K. Xu, R. Zhao, D. Gotz, H. Zha, and N. Cao. Eventthread: Visual summarization and stage analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):56–65, 2018. doi: 10.1109/TVCG.2017.2745320 5, 6

[25] Y. Guo, S. Guo, Z. Jin, S. Kaul, D. Gotz, and N. Cao. Survey on visual analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 28(12):5091–5112, 2022. doi: 10.1109/TVCG.2021.3100413 2

[26] Y. Han, A. Rozga, N. Dimitrova, G. D. Abowd, and J. Stasko. Visual analysis of proximal temporal relationships of social and communicative behaviors. *Computer Graphics Forum*, 34(3):51–60, 2015. doi: 10.1111/cgf.12617 1, 5, 7

[27] J. Heer and B. Shneiderman. Interactive dynamics for visual analysis: A taxonomy of tools that support the fluent and flexible use of visualizations. *Queue*, 10(2):30–55, 2012. doi: 10.1145/2133416.2146416 2

[28] M. Hu, K. Wongsuphasawat, and J. Stasko. Visualizing social media content with sententree. *IEEE transactions on visualization and computer graphics*, 23(1):621–630, 2016. 6, 7

[29] W. Jentner and D. A. Keim. *Visualization and visual analytic techniques for patterns*. Springer, 2019. 7

[30] Z. Jin, S. Guo, N. Chen, D. Weiskopf, D. Gotz, and N. Cao. Visual causality analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1343–1352, 2021. doi: 10.1109/TVCG.2020.3030465 3, 7, 8

[31] J. Jo, J. Huh, J. Park, B. Kim, and J. Seo. Livegantt: Interactively visualizing a large manufacturing schedule. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2329–2338, 2014. doi: 10.1109/TVCG.2014.2346454 3

[32] H. J. Kim, S. E. Hong, and K. J. Cha. seq2vec: Analyzing sequential data using multi-rank embedding vectors. *Electronic Commerce Research and Applications*, 43:101003, 2020. doi: 10.1016/j.elerap.2020.101003 1

[33] J. Krause, A. Perer, and H. Stavropoulos. Supporting iterative cohort construction with visual temporal queries. *IEEE Transactions on Visualization and Computer Graphics*, 22(01):91–100, 2016. doi: 10.1109/TVCG.2015.2467622 3

[34] B. Kwon, M. Choi, J. Kim, E. Choi, Y. Kim, S. Kwon, J. Sun, and J. Choo. Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics*, 25(01):299–309, 2019. doi: 10.1109/TVCG.2018.2865027 3

[35] B. C. Kwon, V. Anand, K. A. Severson, S. Ghosh, Z. Sun, B. I. Frohnert, M. Lundgren, and K. Ng. Dpvis: Visual analytics with hidden markov models for disease progression pathways. *IEEE Transactions on Visualization and Computer Graphics*, 27(9):3685–3700, 2021. doi: 10.1109/TVCG.2020.2985689 3, 6, 7

[36] H. Lam, D. Russell, D. Tang, and T. Munzner. Session viewer: Visual exploratory analysis of web session logs. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pp. 147–154, 2007. doi: 10.1109/VAST.2007.4389008 1, 5, 6, 7

[37] H. Lam, M. Tory, and T. Munzner. Bridging from goals to tasks with design study analysis reports. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):435–445, 2018. doi: 10.1109/TVCG.2017.2744319 2

[38] P.-M. Law, Z. Liu, S. Malik, and R. C. Basole. Maqui: Interweaving queries and pattern mining for recursive event sequence exploration. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):396–406, 2019. doi: 10.1109/TVCG.2018.2864886 6

[39] F. Lekschas, B. Peterson, D. Haehn, E. Ma, N. Gehlenborg, and H. Pfister. Peax: Interactive visual pattern search in sequential data using unsupervised deep representation learning. *Computer Graphics Forum*, 39(3):167–179, 2020. doi: 10.1111/cgf.13971 2

[40] H. Li and N. Homer. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in Bioinformatics*, 11(5):473–483, 2010. doi: 10.1093/bib/bbq015 9

[41] S. v. d. Linden, B. M. Wulterkens, M. M. v. Gilst, S. Overeem, C. v. Pul, A. Vilanova, and S. v. d. Elzen. FlexEvent: going beyond Case-Centric Exploration and Analysis of Multivariate Event Sequences. *Computer Graphics Forum*, 2023. doi: 10.1111/cgf.14820 7, 8

[42] Z. Liu, B. Kerr, M. Dontcheva, J. Grover, M. Hoffman, and A. Wilson. Coreflow: Extracting and visualizing branching patterns from event sequences. *Computer Graphics Forum*, 36(3):527–538, 2017. doi: 10.1111/cgf.13208 1, 2, 5, 6, 7

[43] Z. Liu, Y. Wang, M. Dontcheva, M. Hoffman, S. Walker, and A. Wilson. Patterns and sequences: Interactive exploration of clickstreams to understand common visitor paths. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):321–330, 2017. doi: 10.1109/TVCG.2016.2598797 1, 3, 5, 6, 7, 9

[44] J. Magallanes, T. Stone, P. D. Morris, S. Mason, S. Wood, and M.-C. Villa-Uriol. Sequen-c: A multilevel overview of temporal event sequences. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):901–911, 2022. doi: 10.1109/TVCG.2021.3114868 3

[45] S. Malik, F. Du, M. Monroe, E. Onukwugha, C. Plaisant, and B. Shneiderman. Cohort comparison of event sequences with balanced integration of visual analytics and statistics. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, p. 38–49. ACM, New York, 2015. doi: 10.1145/2678025.2701407 3

[46] Y. Ming, P. Xu, F. Cheng, H. Qu, and L. Ren. Protosteer: Steering deep sequence model with prototypes. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):238–248, 2020. doi: 10.1109/TVCG.2019.2934267 5, 6, 7

[47] M. Monroe, R. Lan, H. Lee, C. Plaisant, and B. Shneiderman. Temporal event sequence simplification. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2227–2236, 2013. doi: 10.1109/TVCG.2013.200 1, 5, 6, 7, 9

[48] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan, R. Chou, J. Glanville, J. M. Grimshaw, A. Hróbjartsson, M. M. Lalu, T. Li, E. W. Loder, E. Mayo-Wilson, S. McDonald, L. A. McGuinness, L. A. Stewart, J. Thomas, A. C. Tricco, V. A. Welch, P. Whiting, and D. Moher. The prisma 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, 372, 2021. doi: 10.1136/bmj.n71 9

[49] Y. Peiris, C. Barth, E. M. Huang, and J. Bernard. A data-centric methodology and task typology for time-stamped event sequences. In *2022 IEEE Evaluation and Beyond - Methodological Approaches for Visualization (BELIV)*, pp. 66–76. IEEE Computer Society, Los Alamitos, 2022. doi: 10.1109/BELIV57783.2022.00012 1, 2, 7, 8

[50] A. Perer and D. Gotz. Data-driven exploration of care plans for patients. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, p. 439–444. ACM, New York, 2013. doi: 10.1145/2468356.2468434 1

[51] A. Perer and F. Wang. Frequence: interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th International Conference on Intelligent User Interfaces*, IUI '14, p. 153–162. ACM, New York, 2014. doi: 10.1145/2557500.2557508 1

[52] C. Plaisant. The challenge of information visualization evaluation. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, p. 109–116. ACM, New York, 2004. doi: 10.1145/989863.989880 1, 9

[53] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. Lifelines: visualizing personal histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, p. 221–227. ACM, New York, 1996. doi: 10.1145/238386.238493 3, 6, 7

[54] C. Plaisant and B. Shneiderman. The diversity of data and tasks in event analytics. https://eventevent.github.io/papers/EVENT_2016_paper_13.pdf, 2016. 1, 2, 7, 8

[55] J. Qi, V. Bloemen, S. Wang, J. van Wijk, and H. van de Wetering. Stbins: Visual tracking and comparison of multiple data sequences using temporal binning. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1054–1063, 2020. doi: 10.1109/TVCG.2019.2934289 1, 6, 7

[56] A. Rind, W. Aigner, M. Wagner, S. Miksch, and T. Lammarsch. Task cube: A three-dimensional conceptual space of user tasks in visualization design and evaluation. *Information Visualization*, 15(4):288–300, 2016. doi: 10.1177/1473871615621602 1, 2

[57] H.-J. Schulz, T. Nocke, M. Heitzler, and H. Schumann. A design space of visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2366–2375, 2013. doi: 10.1109/TVCG.2013.120 1

[58] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012. doi: 10.1109/TVCG.2012.213 9

[59] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE Symposium on Visual Languages*, pp. 336–343, 1996. doi: 10.1109/VL.1996.545307 1

[60] B. Shneiderman. The event quartet: How visual analytics works for temporal data. https://eventevent.github.io/papers/EVENT_2016_paper_7.pdf, 2016. 2

[61] E. R. A. Valiati, M. S. Pimenta, and C. M. D. S. Freitas. A taxonomy of tasks for guiding the evaluation of multidimensional visualizations. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, p. 1–6. ACM, New York, 2006. doi: 10.1145/1168149.1168169 1

[62] K. Vrotsou, J. Johansson, and M. Cooper. Activitree: Interactive visual exploration of sequences in event-based data using graph similarity. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):945–952, 2009. doi: 10.1109/TVCG.2009.117 1, 7

[63] K. Vrotsou and A. Nordman. Exploratory visual sequence mining based on pattern-growth. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2597–2610, 2019. doi: 10.1109/TVCG.2018.2848247 1, 5, 6, 7

[64] T. D. Wang, C. Plaisant, A. J. Quinn, R. Stanchak, S. Murphy, and B. Shneiderman. Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, p. 457–466. ACM, New York, 2008. doi: 10.1145/1357054.1357129 1, 2, 7

[65] T. D. Wang, C. Plaisant, B. Shneiderman, N. Spring, D. Roseman, G. Marchand, V. Mukherjee, and M. Smith. Temporal summaries: Supporting temporal categorical searching, aggregation and comparison. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1049–1056, 2009. doi: 10.1109/TVCG.2009.187 1, 2

[66] K. Wongsuphasawat and D. Gotz. Outflow : Visualizing patient flow by symptoms and outcome. https://gotz.web.unc.edu/wp-content/uploads/sites/5664/2013/10/wongsuphasawat_ieee_visweek_vahc_2011.pdf, 2011. 1, 6

[67] K. Wongsuphasawat, J. A. Guerra Gómez, C. Plaisant, T. D. Wang, M. Taieb-Maimon, and B. Shneiderman. Lifeflow: visualizing an overview of event sequences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, p. 1747–1756. ACM, New York, 2011. doi: 10.1145/1978942.1979196 1, 3, 7

[68] J. Wu, D. Liu, Z. Guo, and Y. Wu. Rasipam: Interactive pattern mining of multivariate event sequences in racket sports. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):940–950, 2023. doi: 10.1109/TVCG.2022.3209452 3, 7, 8

[69] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu. Opinionflow: Visual analysis of opinion diffusion on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1763–1772, 2014. doi: 10.1109/TVCG.2014.2346920 3, 5, 6

[70] J. S. Yi, Y. a. Kang, J. Stasko, and J. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007. doi: 10.1109/TVCG.2007.70515 1, 2, 3

[71] E. Zgraggen, S. M. Drucker, D. Fisher, and R. DeLine. (s|qu)eries: Visual regular expressions for querying and exploring event sequences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, p. 2683–2692. ACM, New York, 2015. doi: 10.1145/2702123.2702262 7

[72] J. Zhao, Z. Liu, M. Dontcheva, A. Hertzmann, and A. Wilson. Matrixwave: Visual comparison of event sequence data. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, p. 259–268. ACM, New York, 2015. doi: 10.1145/2702123.2702419 1, 3, 5, 7

[73] K. T. Zinat, J. Yang, A. Gandhi, N. Mitra, and Z. Liu. A comparative evaluation of visual summarization techniques for event sequences. *Computer Graphics Forum*, 42(3):173–185, 2023. doi: 10.1111/cgf.14821 7, 9