Visual Anomaly Detection in Event Sequence Data

Shunan Guo, Zhuochen Jin, Qing Chen, David Gotz, Hongyuan Zha, Nan Cao

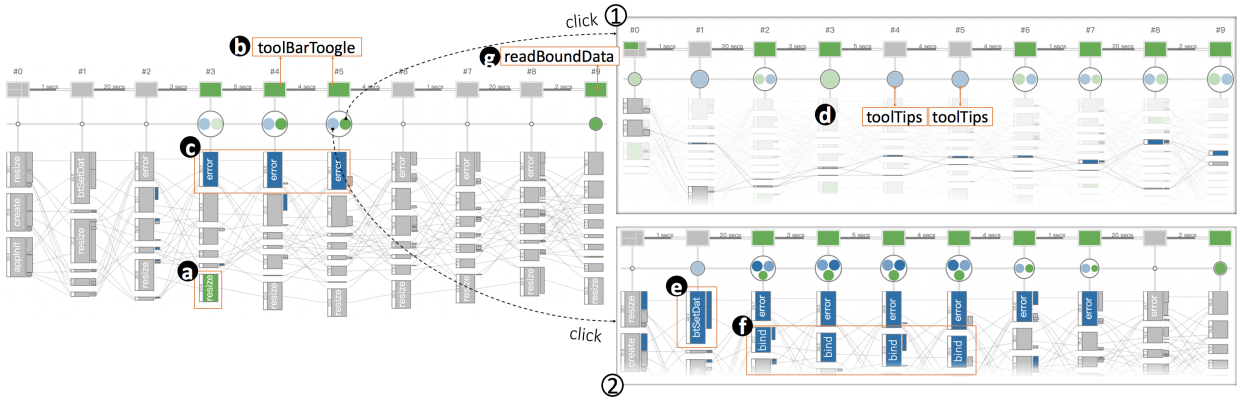

Fig. 1. The anomaly detection result of the Agavue dataset using ET^3 system. Two anomalous pattern were identified: (1) tool-tips failing to show when users interact with toolbars; (2) users encountering an error after requesting data binding.

Abstract— Anomaly detection is a common analytical task that aims to identify rare cases that differ from the typical cases that make up the majority of a dataset. Anomaly detection takes many forms, but is frequently applied to the analysis of event sequence data in addressing real-world problems such as error diagnosis, fraud detection, and vital sign monitoring. With general event sequence data, however, the task of anomaly detection can be complex because the sequential and temporal nature of such data results in diverse definitions and flexible forms of anomalies: a specific event type, a sub-sequence or the entire sequence. This, in turn, increases the difficulty in interpreting detected anomalies, a critical element in raising human confidence with the analysis results. Most prior work in event sequence anomaly detection focuses on detecting only one type of anomaly with a specific application domain. Such approaches do not provide flexible ways of identifying diverse types of anomalies, and provide only limited support for result interpretation. In this paper, we propose an unsupervised anomaly detection algorithm based on Variational AutoEncoders (VAE). The model learns latent representations for all sequences in the dataset and detects anomalies that deviate from the overall distribution. Moreover, the model can estimate an underlying normal progression for each given sequence represented as occurrence probabilities of events along the sequence progression. Events in violation of their occurrence probability (i.e., event occurrences with small occurrence probability, and absent events with large occurrence probability) are identified as abnormal. We also introduce a visualization system, EventThread3, to support interactive exploration of the analysis result. The system facilitates interpretations of anomalies within the context of normal sequence progressions in the dataset through comprehensive one-to-many sequence comparison. Finally, we quantitatively evaluate the performance of our anomaly detection algorithm and demonstrate the effectiveness of our system through case studies in three different application domains and report feedback collected from study participants and expert users.

Index Terms—Illustrative Visualization, Time Series Data, Visual Knowledge Discovery, Visual Knowledge Representation

1 Introduction

Anomaly detection is a common task for event sequence data analysis as it often contributes to the discovery of critical and actionable information [12]. Effective use of event sequence data can require identifying sequences that deviate from the typically occurring behavior [37]. For example, a doctor may be interested in finding patients whose postoperative response is different from other patients who have had the same surgery, so that the doctors can provide personalized care plans for similar patients in the future.

A variety of work has been developed for detecting anomalies in a wide range of areas, such as medical diagnosis [24], fraud detection [42], and application log surveillance [40]. Especially for event sequence data, traditional statistical models [38, 46], supervised or semi-supervised approaches [33], and unsupervised methods [35] have been applied. Nevertheless, there are three major challenges we face for anomaly detection in event sequence data.

The first critical challenge is the development of an unsupervised anomaly detection method that can efficiently handle the complex temporal structure of event sequences. Regarding the performance of anomaly detection methods, traditional statistical models are sub-optimal for event sequence data because they fail to capture complex structures in the data and require fine tuning of parameters for different training sets [18]. To circumvent the issue of algorithm tuning, supervised and semi-supervised approaches have become increasingly popular and applied to diverse areas of anomaly detection [8, 16]. However, since anomalies are rare and real-world event sequence datasets are often huge in scale, it can be difficult, if not impossible, to obtain the required labels for these approaches.

The second challenge is that anomalies themselves are hard to precisely define. Since the nature of anomalies can differ fundamentally across domains, anomaly detection techniques are often tailored to specific application domains such as computer system diagnosis [26], medical treatment examination [3], and financial fraud detection [5]. Furthermore, even within a given domain, there is often no clear definition or criteria to distinguish between normal and abnormal cases.

Beyond these two challenges for general anomaly detection–the lack

- *Shunan Guo and Hongyuan Zha are with East China Normal University. E-mail: g.shunan@gmail.com, zha@sei.ecnu.edu.cn*
- *Zhuochen Jin is with iDVX lab at Tongji University.E-mail: zhuochen.jin@tongji.edu.cn*
- *Qing Chen is with Aviz at Inria. Email: jane.qing.chen@gmail.com*
- *David Gotz is with University of North Carolina at Chapel Hill. Email: gotz@unc.edu*
- *Nan Cao is with iDVX lab at Tongji University and is the corresponding author. Email: nan.cao@tongji.edu.cn*

of labeled data, and the fuzzy boundary between abnormal and normal cases–the third key challenge lies in providing interpretable information about anomalies that supports understanding of the analyzed result. Due to the temporal characteristics of event sequence data and the black-box nature of machine learning models, it is especially challenging to interpret anomalous sequences once identified. For analysts to derive actionable insights, they must be able to understand how anomalies are different from "normal" sequences, which event or series of events characterize the anomaly, and which events suggest actions that could help avoid such cases in the future.

To tackle the three challenges, we propose an unsupervised anomaly detection model for event sequence data that builds upon LSTM-based Variational AutoEncoders (VAE) [36]. A recent advance over traditional autoencoder-based anomaly detection techniques, VAE use a probabilistic encoder for modeling the distribution of the latent variables [2]. Such probabilities give more principled criteria for identifying anomalies and do not require model-specific thresholds. As a result, VAE better facilitate objective judgments for deciding the boundary of anomalous sequences compared to other unsupervised algorithms. Moreover, to account for the temporal dependencies of event sequence data into a VAE, we replace feed-forward networks with LSTM neural networks to incorporate the sequential nature of event sequence data. The model is trained to learn a latent representation for each event sequence and identify anomalous sequences based on their deviation from the overall distribution. A mean sequence is computed from the reconstruction probabilities for each sequence detected as an anomaly, which shows the occurrence probabilities of events in normal circumstances, representing a corresponding "normal" sequence progression for the anomaly. For example, a patient having internal bleeding should normally be sent to emergency for surgery, thus the reconstruction probabilities shall identify surgical events with high probabilities after hospital admission. To facilitate anomaly interpretation, we also compare the anomaly sequence with a collection of sequences having similar progression but identified as "normal" so as to uncover their critical differences.

The system was iteratively developed and guided by feedback from domain experts. To incorporate experts' domain knowledge within the anomaly detection and interpretation process, we present an interactive graphical interface which allows one-to-many comparisons between a selected anomalous sequence and a set of normal sequences at multiple levels of granularity. The interface design includes a comparison glyph to highlight suspicious events and incorporates rich interactions to enable dynamic tweaking of the analysis result. Interactions also support the interpretability of the anomalies through additional contextual information. The main contributions are as follows:

- **Anomaly Detection Algorithm.** We introduce a VAE-based anomaly detection algorithm for detecting anomalous sequences within a collection of temporal event sequences. The algorithm infers a normal progression of events for each anomalous sequence and identifies anomalous events that deviate from the normal progression in an interpretable manner.

- **System for Interactive Anomaly Analysis.** We present an interactive visual analysis system for dynamic exploration and interpretation of the anomaly detection result. The system allows users to investigate the anomalous sequence in the context of normal sequences through one-to-many sequence comparisons at different levels of granularity. A comparison glyph is designed to highlight differences and facilitate visual comparison. Rich interactions are provided to allow flexible exploration and incorporate human knowledge during the analysis.

- **Evaluation.** We evaluate our proposed method both quantitatively and qualitatively through (1) a performance evaluation of our VAE-based anomaly detection algorithm by comparing with two baseline methods, (2) three case studies conducted with real-world datasets in different application domains. We also report the comments collected from study participants.

2 Related Work

In this section, we provide an overview of the analytical and visualization techniques that are most related to our work, including (1) anomaly detection algorithms, (2) visualization for anomaly detection, and (3) visual comparison techniques.

2.1 Anomaly Detection Algorithms

Anomaly detection has been extensively studied over the past years [12]. Methods for anomaly detection can be broadly categorized into tensor-based algorithms [13], statistics-based algorithms [41], classification-based algorithms [33], and neighbor-based or distance-based algorithms [6]. Although these methods are effective in identifying anomalies with numeric results, they are not capable of considering the sequential structure when detecting anomalies for event sequences.

Meanwhile, as the data volume grows, it becomes increasingly difficult to apply traditional anomaly detection algorithms. More recent work with deep learning-based anomaly detection (DAD) algorithms has been developed to meet this challenge. They have been applied to a variety of anomaly detection applications including fraud detection, cyber-intrusion detection, medical anomaly detection, sensor networks, video surveillance, internet-of-things, log analysis, and industrial damage detection [10]. Types of DAD models include unsupervised (e.g., autoencoder, generative adversarial, variational), semi-supervised (e.g., reinforcement learning), hybrid (e.g., feature extractor+traditional algorithms) [15], and one-class neural networks [11].

In this work, we leverage a variant of Variational AutoEncoders (VAE) which can both deal with large volumes of unlabeled data, and identify anomalous patterns with probability measures [2]. Furthermore, given the temporal information inherent to event sequence data, our approach replaces the feed-forward networks with LSTM neural networks. This generates a output close to the original input and provides latent feature vectors for each event sequence in the dataset. However, the boundary between normal and anomalous behavior is often not precisely defined. This lack of a well-defined boundary poses challenges for traditional and deep learning-based algorithms alike. For this reason, incorporating human domain knowledge through interaction can benefit the anomaly detection process.

2.2 Visual Anomaly Detection

To facilitate anomaly detection and reasoning over the results, researchers have developed many visual anomaly detection tools [8, 43]. As previously mentioned, two major challenges in anomaly detection are (1) the fuzzy boundary between normality and abnormality, and (2) the absence of high quality labeled data. Visual anomaly detection tools that allow domain experts to leverage their knowledge and experience can help overcome these challenges. For these tools, effectiveness and intuitiveness are both key design priorities, and a number of alternative visual analysis approaches have been proposed. This includes methods for the detection of anomalous user behaviors from sequence data [4]. Chae et al. [9] applied traditional control chart methods together with seasonal trend decomposition to extract outliers. Thom et al. [43] introduced a visual analysis system to monitor for anomalous bursts of keywords. More recently, FluxFlow [48] was developed to reveal and analyze anomalous information processes in social media.

Although systems mentioned above are often designed to help detect anomalous points, few approaches focus on identifying anomalous sequences or on the comparison between the detected outliers and "normal" sequences. To enhance the interpretability of the analyzed results, our system supports one-to-many sequence comparison at multiple granularities. We also design comparison glyphs to help with discovery and support rich interactions to facilitate result interpretation.

2.3 Visual Comparison

Visual comparison is a common task when investigating data similarities and differences [29]. In the information visualization domain, Gleicher et al. [20] classify visual comparison techniques into three categories: juxtaposition by comparing objects side-by-side, superposition by overlaying data with a shared reference in the same space (e.g., [44]), and explicit encoding by directly computing and presenting

the differences or correlations (e.g., [21]). Each approach has its advantage, and multiple methods can be employed in combination to make a comparison. Within the three major categories, a variety of alternatives have been developed for specific tasks. For example, Kehrer et al. [30] proposed a formal model for hierarchically-partitioned category comparison with small-multiple displays. This approach was inspired by the ineffectiveness of juxtaposition when dealing with a large number of categories. Their work supports superposition and explicit encoding of differences for semantically meaningful comparisons.

Visual comparisons have also been studied in the context of event sequence analysis. MatrixWave [49] applied superposition with an explicit encoding of sequence differences when comparing two event sequences. EventAction [14] used a calendar view to show several temporal event sequences and placed them in a ranked list to compare different sequences via juxtaposition. Most of the previous work focuses on the visual comparison of single sequences (one-to-one), or of event sequence groups (many-to-many). However, for anomaly detection, the one-to-many comparison is crucial. In ET^3, we provide an interactive comparison between the anomalous sequence and normal progressions (one-to-many) at three levels of granularity.

3 System Overview

Our system is designed to support interactive exploration and interpretation of anomalies in event sequence data. The designs were iteratively improved over four months through regular discussions with a domain expert in the anomaly detection field. We identified the following design requirements (R1-R4) for detecting anomalies in event sequences. These requirements were distilled from (1) feedback from the domain expert, (2) the authors' experiences with event sequence analysis, and (3) a thorough review on existing techniques and their limitations.

- **R1 Remove noise from the data and extract key features for anomaly detection.** Real-world event sequence datasets often include a large amount of noise as reflected by highly heterogeneous event types and uncertain event ordering. The system should be able to filter out unimportant variation to allow for higher quality subsequent analysis.
- **R2 Detect anomalous instances within unlabeled datasets.** Real-world event sequence datasets are often large and rare contain labels. It can be enormously time-consuming to manually find and label rarely occurring anomalous instances. Users, therefore, often wish to focus their inspection on detected outliers rather than conduct a comprehensive search of the entire dataset. Thus, the system should incorporate an unsupervised anomaly detection mechanism that can narrow the scope of analysis.
- **R3 Localize anomalous events within abnormal sequences.** The interpretation of the detected anomalous sequences relies on the analysis of low-level events. For example, in medical health record analysis, the clinical path of a patient may be detected as an anomaly due to a misused medicine. However, real-world event sequences can be long in length and heterogeneous in types, which makes it difficult to identify anomalous events. Therefore, the system should be able to highlight anomalous events within the abnormal sequence to facilitate reasoning and interpretation.
- **R4 Analyze anomalous sequences within the context of normal progressions.** Instead of focusing on a single anomalous event, analyzing anomalies within the context of normal progressions can provide insights for interpretation. For example, a medical expert may inspect how well treatment plans are followed under normal circumstances to understand how an anomalous treatment event deviates from the typical population. Moreover, comparing normal and abnormal sequences can help reveal higher-level anomalous patterns (e.g., anomalous sub-sequences, anomalous event ordering) beyond low-level anomalous events. Thus, the system should allow users to view anomalous sequences within the context of normal progressions to help interpret the anomalies.
- **R5 Support interactive sequence analysis and exploration at different levels of granularity.** Sequences of events may vary significantly in both events observed and the speed of progression. Different levels of sequence aggregation may yield different insights during the comparison. The system should, therefore, provide users with an interactive environment to enable flexible comparisons under various levels of aggregation, from detailed low-level events to aggregated high-level progressions.

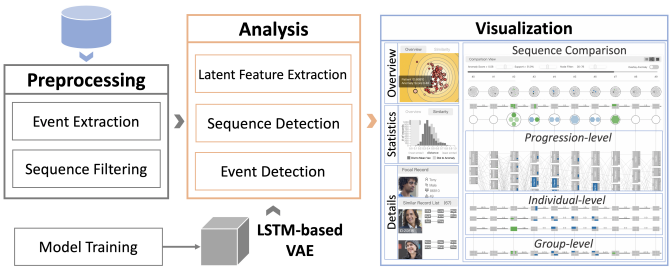

Fig. 2. The ET^3 system integrates three major modules to support interactive visual anomaly detection of event sequence data, including a preprocessing module, an analysis module and a visualization module.

Motivated by these requirements, we developed ET^3, an interactive visualization system for detecting and visualizing anomalies in temporal event sequences. As illustrated in Fig. 2, the system includes three major modules: (1) a data preprocessing module, (2) an anomaly detection module, and (3) a visualization module.

The data preprocessing module focuses on reducing noise to prepare high-quality event sequence data for subsequent training of the anomaly detection model (**R1**). In particular, we measure the importance of each event using Term Frequency-Inverse Document Frequency (TF-IDF) scores to remove noisy events and exclude extremely short sequences (i.e., sequence length < 2) [23]. In the analysis module, we identify anomalous sequences (**R2**) and uncover event occurrence probabilities of normal progressions from a trained Variational AutoEncoders (VAE) model with the Long Short Term Memory networks (LSTMs). We further localize anomalous events within the detected outliers by referring to the occurrence probabilities of each event in normal progressions (**R3**). The analysis results are then sent to the visualization module for interactive visual analysis of the anomalous sequences via multi-granular sequence exploration and data comparison (**R4, R5**).

4 VAE-Based Anomaly Detection

In this section, we formalize the analytical tasks of event sequence anomaly detection and introduce the unsupervised VAE-based anomaly detection model we propose to solve this problem.

4.1 Algorithm Overview

Detecting anomalies in event sequences is analytically challenging for three reasons. First, the sequential and temporal nature of event sequences results in complex anomaly structures, which makes it difficult to determine the abnormality of the entire corresponding sequences. Second, event sequence datasets are typically diverse with different lengths and progression patterns, resulting in high variability within the training data. Third, given the characteristics above, it is difficult to characterize abnormalities for interpretation, which makes it difficult to understand the reasons for the model's decisions.

To address these challenges, we adapted a Sequence-to-Sequence Variational AutoEncoder (VAE) to interpretably detect anomalies in event sequences. In particular, we leverage the merits of deep neural networks in learning complex sequential patterns to address the first challenge and the probabilistic foundation of VAE in capturing data variability to solve the second. Finally, we employ the reconstruction probabilities output from the VAE to facilitate the interpretation of the anomalous sequences. As shown in Fig. 3, the algorithm consists of three major steps: (1) latent feature extraction using an LSTM-based VAE, (2) anomalous sequence detection, and (3) anomalous event analysis. In the first step, we train the VAE-based model to extract low-dimensional feature representations (i.e. the latent vector $\boldsymbol{z}$) to characterize the progression of each input sequence. The second step employs the latent vectors to measure the outlierness for each sequence based on their Local Outlier Factor (LOF), which is then to used to

identify anomalous sequences **(R2)**. In the third step, the latent vectors are fed to the decoder of the VAE model for sequence reconstruction and event anomaly detection **(R3)**.

4.2 LSTM-Based Variational AutoEncoder

We first introduce the structure of the Sequence-to-Sequence VAE model. The model contains two modules: the VAE encoder and the VAE decoder. Both modules are designed using Recurrent Neural Networks to better extract sequential patterns from event sequence data. In particular, the encoder captures the latent distribution of sequences and the decoder inversely restores the distribution to estimate the occurrence probabilities of events in each time slot.

VAE Encoder. The encoder is trained to abstract the input sequence $\{X = \boldsymbol{x}_i\}_{i=1}^{n}$ into a low-dimensional latent feature vector that describes a sequential distribution of events occurring in the sequence. In this input, n is the length of the sequence and $\boldsymbol{x}_i \in \{0,1\}^{|E|}$ is the multi-hot encoding of the events occurring in the i-th time step. Each coordinate represents an event type, which is marked 1 if the corresponding event occurs in the i-th time step, or 0 otherwise. After feeding the multi-hot vectors into the corresponding layer of RNN, the state of the entire sequence is extracted and represented in the hidden state vector $\boldsymbol{h}_{enc}$ of the last layer, which is denoted as follows:

$$\boldsymbol{h}_{enc} = encoder(X) \tag{1}$$

The hidden state vector $\boldsymbol{h}_{enc}$ is projected into two vectors $\boldsymbol{\mu}$ and $\boldsymbol{\delta}$ to parameterize a normal distribution, representing the mean value and standard deviation of the normal distribution respectively. To take the variability of the latent space into account (i.e., to represent the diversity present in normal cases), we draw a low-dimensional latent vector $\boldsymbol{z}$ by randomly sampling from the distribution. We then use this vector as a representative of the original distribution for subsequent decoding.

VAE Decoder. In the decoder, we reconstruct the input sequence from the extracted latent feature vector $\boldsymbol{z}$. Specifically, $\boldsymbol{z}$ is fed to each layer of the RNN to estimate the probability distribution of events for each time slot. We formally define the decoding procedure as follows:

$$X^{'} = decoder(z) \tag{2}$$

where $X^{'}$ is a sequence of probability distributions denoted as $X^{'} = \{\boldsymbol{x}_i^{'}\}_{i=1}^{n}$, and the element $x_{i,j}^{'}$ in $\boldsymbol{x}_i^{'} \in R^{|E|}$ represents the occurrence probability of j-th event at the i-th time step.

Training Process. We train the model with a goal of narrowing the gap between the original input sequence and its reconstruction, which can be formally defined as minimizing the following loss function:

$$L = L_r + w_{kl} \cdot L_{kl} \tag{3}$$

$$L_r = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{|E|}(w_{e_j} x_{i,j} log(x_{i,j}^{'}) + (1 - x_{i,j}) log(1 - x_{i,j}^{'})) \tag{4}$$

$$L_{kl} = -\frac{1}{M_z}\sum_{i=1}^{M_z}(1 + log(\sigma_i^2) - \mu_i^2 - \sigma_i^2) \tag{5}$$

The first term L_r is the reconstruction loss which calculates the weighted cross entropy between $x_{i,j}$ and $x_{i,j}^{'}$, indicating an event-level difference between the reconstruction and the original input with respect to the j-th event at the i-th time step. In particular, a parameter $w_{e_j} = 1/log(n_j)$ is introduced to reduce the marginal importance of high-frequency events so as to address the issue of skewed dataset, where n_j is the number of occurrences for event e_j. The second term L_{kl} is the Kullback-Leibler Divergence Loss which estimates a distribution-level difference between the distribution of the latent vector $\boldsymbol{z}$ and a normal distribution $N(0,1)$, where M_z is the dimension of the latent vector $\boldsymbol{z}$. These two terms are balanced with a parameter w_{kl}.

Parameter Settings. Both the encoder and decoder employ LSTM units [25] with 300 hidden nodes. We set the dimension of the latent vector to 16. The parameter w_{kl} adaptively increases from 0.1 to 0.5 during the training process to make sure the reconstruction loss is optimized with high priority. Moreover, we optimize the loss function with the Adam optimizer [31] with training data batch size of 80 for each training step. We train the model on an Nvidia Tesla K80 graphics card. Each training epoch takes approximately 10.5 seconds on average.

4.3 Anomalous Sequence Detection

After training the model, we employ the latent vector $\boldsymbol{z}$ of each input sequence to detect anomalous sequences in the dataset **(R2)**. Specifically, we adapt the Local Outlier Factor (LOF) to evaluate the degree of anomaly for each sequence in the latent space. We formally define the anomaly degree as follows:

$$LOF(z) = \frac{\sum_{y \in N_k(z)} D_k(y)}{|N_k(z)| D_k(z)} \tag{6}$$

$$D_k(z) = \frac{|N_k(z)|}{\sum_{y \in N_k(z)} (max(d_k(y), d(z,y)))} \tag{7}$$

where $N_k(z)$ is a set of the k-nearest neighbors of $\boldsymbol{z}$, $D_k(z)$ is the neighborhood density of the vector in the latent space, $d(z,y)$ is the Euclidean distance between z and y, and $d_k(y)$ is the maximum distance between y and its neighbors.

In the unsupervised anomaly detection process, it is assumed that the majority of the sequences in the dataset are normal. As the LOF score compares the local density of the latent vector with its neighborhood vectors, normal sequences should group within a dense space with smaller LOF scores, while instances in sparse areas will have larger LOF scores and will be identified as outliers.

4.4 Anomalous Event Analysis

To facilitate the interpretation of sequence anomalies, we further identify anomalous events that contribute to sequence abnormality by analyzing the reconstruction probabilities **(R3)**. As mentioned earlier, the reconstruction probabilities are restored from the latent vector $\boldsymbol{z}$ that is sampled from the latent sequence distribution. As we assumed that the majority of the sequences are normal, the reconstruction probabilities shall be similar to the normal progression of sequences. Moreover, the training objective ensures that the reconstruction probabilities are also similar to the original input sequence. Combining these two points, the reconstruction probabilities of the anomalous sequences can be used to infer an expected "normal" progression for the anomalous sequence. From this, we can identify the anomaly events within the anomalous sequence that deviate from the expected normal progression.

We categorize the anomalous events into two types: missing events and redundant events. Defined intuitively, missing events (noted as x^{mis}) represent the cases where an event shows a high occurrence probability in the reconstruction but does not appear in the sequence. Conversely, redundant events (noted as x^{red}) indicate events that exist in the anomalous sequence but are not expected to occur. Based on this intuition, we calculate an **anomaly score** for each event to assess its level of abnormality. The anomaly scores for missing events and redundant events are $Pr(X = x^{mis})$ and $1 - Pr((X = x^{red}))$, respectively, where $Pr(X = x)$ indicates the occurrence probability of the corresponding event derived from the reconstruction. Consequently, events with an anomaly level higher than a user-defined threshold are identified as anomalous. The threshold is by default set as 0.6, which can be adjusted by users during an analysis via the visualization module.

5 VISUALIZATION

This section presents the visualization and interaction designs for the ET^3 system. We first introduce a set of tasks that guide the design choices of our system. We then describe ET^3's key design elements.

5.1 Design Tasks

The ET^3 design was iteratively refined through a four-month development cycle based on feedback collected from regular meetings with our domain expert in the field of anomaly detection. In response to that feedback, we formulated a set of design tasks to solve the key challenges in visually analyzing anomalies in event sequence data and also to further meet the design requirements (R3-R5) discussed in Section 3.

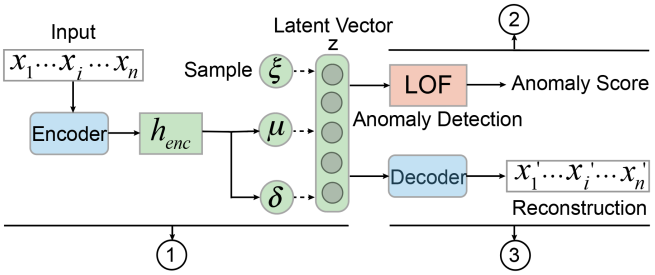

Fig. 3. Schematic diagrams of the model, (1) the VAE model to obtain the latent vector of the input sequence, (2) anomaly detection of the overall sequence, and (3) anomalous event detection based on the reconstruction of the input sequence.

- **T1 Provide an overview of the analysis scope.** The analyzed event sequence dataset may contain many anomalous event sequences with different levels of abnormality. To help users find anomalous sequences of interest for subsequent analysis, the system should provide an overview of all sequences detected as anomalies in the dataset and illustrate their level of abnormality.
- **T2 Emphasize anomalous events within the sequence.** To help quickly explore complex event sequences and uncover the reason behind an abnormality, the visualization should be designed to highlight key events that are suspicious of being anomalous.
- **T3 Facilitate result interpretation in context.** Interpreting anomaly detection models is critical for establishing users' trust in the analysis result. It is also challenging because the behavior of deep neural networks is difficult to explain. The designed visualization should help users effectively analyze the detected anomalies within the context of the entire training set, to uncover the difference between abnormal and normal sequence progressions and facilitate reasoning about the analyzed result.
- **T4 Support sequence exploration at multiple levels of granularity.** Applying different levels of aggregation for a group of sequences can result in distinct interpretations of the result. For example, the anomalous events detected by comparing an anomalous sequence with an individual normal sequence may be different from the result when comparing with a subgroup. To support more accurate findings, the system should support the exploration of normal sequences at different levels of granularity.
- **T5 Easy access to raw data and auxiliary context.** While aggregating event sequences and extracting high-level patterns are essential for easy exploration, access to raw data is also important in raising users' confidence in assessments of data anomalies. Thus, the system should enable users to easily access low-level details on demand.
- **T6 Incorporate human judgement in the analysis loop.** Anomaly detection is prone to error due to its subjective nature. The definition of the anomaly may also vary in different applications. The system should provide rich interactions to help users refine the analysis results according to their domain knowledge.

5.2 User Interface

Guided by the tasks above, ET^3 incorporates seven key views to visually analyze the anomalous sequences (Fig. 4). A user starts with the *anomaly overview* (Fig. 4(1)), which provides an MDS projection of the latent vectors for all anomalous sequences in the dataset and allows users to select an anomalous sequence for subsequent analysis **(T1)**. The *similarity view* (Fig. 4(2)) displays the distribution of all normal sequences, and their similarities to both the mean sequence and selected anomalous sequence **(T1)**. From this view, users can select a group of records for review in the main panel.

The main panel supports visual anomaly detection via comparison and is composed of two major parts. First, a *reconstruction view* (Fig. 4(3)) shows the occurrence probabilities of the events in each time slot. Second, two coordinated views (Fig. 4(4-5)) support comparison of a selected anomaly with normal progressions in different modes. Specifically, the *flow overview* (Fig. 4(4)) aggregates the flow of normal sequences into a Sankey-like format with the evolution of the selected anomaly overlaid at the top to show differences. The comparison view (Fig. 4(5)) separates the flow of the anomalous sequence from the normal sequences with *comparison glyphs* (Fig. 4(a)). These glyphs are designed to facilitate visual comparison and highlight suspicious events. The normal sequences are visualized with three different variants to support analysis at various levels **(T4)**. The three variants—*sequence comparison view*(Fig. 4(5a)), *flow comparison view*(Fig. 4(5b)), and *summarization view*(Fig. 4(5c))—display individual sequences, progression patterns, and event distributions, respectively.

Access to raw sequence data is provided via two panels **(T5)**. Details about the selected anomaly are displayed in the *anomalous record view* (Fig. 4(6)) while data for similar normal sequences are displayed in the *similar record list* (Fig. 4(7)).

Usage scenario. To understand how these different views work together to identify and interpret event sequence anomalies, let us consider a use case of a drug regulator, Jim, who is responsible for analyzing the medication records for 5,000 patients to investigate misuse of prescription drugs. Taking the records of all patients as the training set, the model outputs 50 patients who exhibit anomalous medication usage. Jim needs to further exploration to determine the true misuses. He first selects a sequence with high LOF score from the *overview*. In response, the mean sequence of the selected anomaly is displayed in the *mean sequence view*. Jim switches to the *statistics view* and selects a group of normal sequences that are most similar to the mean sequence. The *flow overview* presents the evolution pathways for all the selected sequences, from which Jim notices that most patients following a common treatment plan. However, the flow of the anomalous patient contains a number of medicines that deviate from the common pattern. Jim splits the flow of the anomalous sequence from the normal sequences for a clearer illustration of their differences. He explores the progression of normal patients to select different subgroups for comparison and checks the comparison glyphs to find unexpected medications identified by the system. Jim notices medicine *A* is highlighted in several comparison glyphs. However, Jim also knows that *A* should not be taken simultaneously with medicine *B*. Jim selects a subgroup of patients taking medicine *B* for further comparison, and realizes none of the normal patients took medicine *A*. The *sequence view* and *summarization view* further confirm his finding.

5.3 Interactive Anomalous Event Analysis

To help interpret a selected anomaly in the context of the progression of normal sequences **(T3)**, our system is designed to support interactive one-to-many visual comparison. The comparison view in Fig. 4(b) is vertically divided into three regions: an *anomalous sequence* at the top, a group of *comparison glyphs* in the middle, and a summarization of normal sequences at the bottom.

5.3.1 Anomalous Sequence

The selected anomalous sequence is displayed using a line of rectangular nodes ordered by time of occurrence. To deal with the issue of event co-occurrence and avoid event overlap, we display the sequence with a visual technique introduced in [27]. Specifically, concurrent events are grouped into treemaps at each time slot, and all event nodes are color-coded according to the type of anomaly. To make full use of the horizontal space, event nodes are spaced with equal distance and connected with duration bars to reveal the span of time. The time span between events is proportional to the duration bar.

5.3.2 One-to-Many Sequence Comparison

Our system incorporates a one-to-many sequence comparison mechanism, which allows users to validate the anomalies detected by the model by comparing the anomalous sequence with a collection of similar sequences from the normal group. This aims to help users establish confidence in the analysis result based on evidence in the dataset.

The comparative analysis consists of two steps: sequence alignment and support rate calculation. In the first step, we employ a sequence alignment technique introduced in [22] to semantically map each normal sequence to the focal anomaly based on Dynamic Time Warping

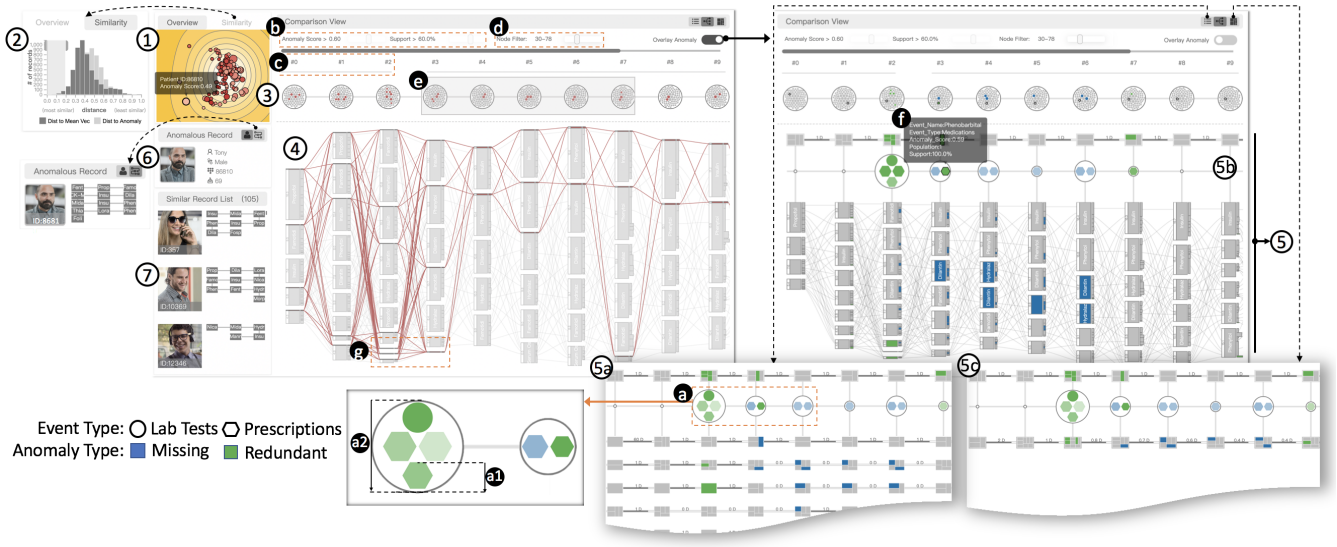

Fig. 4. The user interface of ET^3 consists of seven key views to support comparison-based visual anomaly detection, which includes a (1) anomaly overview, a (2) similarity view, a (3) reconstruction view, a (4) flow overview, a (5) comparison view with three variants (5a-5c), an (6) anomalous record view and a (7) similar record list. The comparison glyphs (a) are designed to intuitively highlight anomalous events in the sequences so as to facilitate visual comparison.

(DTW) [34]. The intention is to address the issues of variable sequence length and progression rate, so as to support a more precise comparison of events. After sequence alignment, we compare events occurring in each time slot to calculate a **support rate** for each anomalous event identified in Sec. 4.4. Intuitively, the support rate represents the proportion of normal sequences that "support" the corresponding event to be abnormal. More specifically, the support rate of a missing event x^{mis} is the proportions of sequences that include x^{mis} in the corresponding time slot, while the support rate for x^{red} is the contrary.

5.3.3 Comparison Glyph

To facilitate visual comparison, we design a *comparison glyph* (Fig. 4(a)) that highlights the anomalous events in each time slot. We encode four critical variables to help quickly identify problematic time slots and events that need further inspection: the overall abnormality of the time slot, the abnormality of each event, the type of anomaly, and the support rate for each anomalous event. Specifically, each circle inside the glyph represents an anomalous event. The size of each internal circle(Fig. 4(a1)) indicates the anomaly score of the corresponding event derived from Sec. 4.4, and the size of outer circle (Fig. 4(a2)) represents the overall abnormality at the corresponding time slot. The type of abnormality (e.g., missing event or redundant event) is distinguished with different colors, consistent with other views. The support rate of each anomalous event (Sec. 5.3.2) is encoded with color saturation.

Updating with user feedback. To leverage analyst domain knowledge, the system allows users to interactively tweak the anomalous events displayed in the comparison glyphs. As shown in Fig. 4(b), users can tune the thresholds for the anomaly score and support rate that determine the conditions at which an event is identified as anomalous. Moreover, when users select a subgroup of normal sequences during the analysis, the comparison glyphs will also be updated simultaneously to reflect the support rate within the subgroup (Fig. 1(1)).

5.3.4 Multi-granular Sequence Aggregation

To support more comprehensive one-to-many sequence comparisons, the design provides three coordinated comparison views (Fig. 4(5a-c)). The views support comparison at different levels of aggregation **(T4)**, and transitions allow users to move smoothly from one view to another.

Sequence Comparison View. The *sequence comparison view* displays the sequences of normal records individually, which aims to support sequence-to-sequence level comparison and efficient access to the raw data. For example, to avoid introducing noisy events, a doctor may be interested in comparing the anomaly only with the most similar "normal" patients. As shown in Fig. 4(5a), the normal sequences are displayed in a scrollable list, ranked from top to bottom according to the degree of similarity. As described in Sec. 5.3.2, each normal sequence is temporally warped and aligned to the anomalous sequence for comparison. We then apply this alignment to each normal sequence to map events into corresponding time slots. To allow an intuitive sequence-to-sequence comparison, the encoding schema of each individual sequence is kept consistent with the anomalous sequence. Users can select any individual sequence to update the comparison glyphs with their differences during the analysis.

Flow Comparison View. The *flow comparison view* (Fig. 4(5b)) provides a progression-level summarization on all normal sequences by aggregating them into a flow-based visualization. This view aims to incorporate confidence of abnormality for anomalous events by comparing the anomalous sequence with subgroups of sequences having particular progression patterns. Specifically, identical events in each time slot are grouped into nodes, and the transition paths among events in adjacent time slots are merged into links. Note that when multiple events co-occur at the same time slot, an individual will be equally divided by the number of event types to ensure the total population remains consistent throughout the sequence progression. The height of each node represents the population (weighted by event co-occurrence) having the event at the corresponding time slot, with the exact number displayed in a label to the left side of each node. Event nodes are connected with links to represent a sequence path from one event to another. To incorporate time information into the flow diagram, we employ a specially designed link introduced in [27]. Specifically, each link is consist of two key components: a duration bar and a connection line. The height of the duration bar shows the proportion of the population corresponding to the link, while the width indicates the average time gap between events. The connection line links the end of the duration bar to the destined event and reveals patterns of sequence progression.

Summarization View. In the *summarization view* (Fig. 4(5c)), nodes in each time slot are further aggregated into a more compact form, illustrating the highest-level summarization of the distribution of events. This view aims to support a comparison of the anomalous sequence against the overall progression of the entire set of similar records. To facilitate the comparison, we encode the summarized sequences in a way similar to the anomalous sequence, with the only difference that the size of each inner rectangle represents the size of the population.

5.4 Other Views

The system also includes a number of contextual views to display auxiliary information and provide access to raw data **(T5)**. These views are coordinated with the selections and filters made in other views to

support the interpretation of anomalies.

Anomaly Overview. The *anomaly overview* (Fig. 4(1)) is designed to help analysts choose sequences of high anomaly degree for subsequent analysis. It shows the distribution of all anomalous sequences based on multidimensional scaling (MDS) projection of the latent vector z. Each anomalous sequence is represented as a circle with the size indicating the LOF score, and the color saturation indicating the sequence length. The distance between two circles reflects their similarity, and we further incorporate a colored contour map to illustrate the local density of circles. Intuitively, circles with larger size and in low-density areas are the most likely anomalies.

Similarity Distribution View. The *similarity distribution view* (Fig. 4(2)) displays the distribution of all normal sequences in the dataset based on their similarity to the progression of the selected anomaly. This view aims to help users select a proper group of normal sequences to support comparative analysis. We measure sequence similarity in two ways: their distance to the mean sequence, and their distance to the selected anomaly. The distances are calculated by aligning each normal sequences to the mean sequence and the anomalous sequence, respectively. Intuitively, sequences similar to the mean sequence would include the events with high occurrence probability at each time slot, though not necessarily with the same progression. Sequences similar to the anomalous sequence, meanwhile, tend to have events and progression patterns that exactly match the anomalous sequence. The system allows users to switch between the two types of measurement according to their preference.

Reconstruction View. The reconstruction probabilities (given by Equation 2) of the selected anomaly are shown in the *reconstruction view* (Fig. 4(3)) with the intent of providing an overview of the occurrence probabilities of events for each time slot. The reconstruction probabilities are shown as a line of circle packings arranged in time order. The size of each circle shows the value of probability, and the color indicates different anomaly types (with colors consistent with other views). Circles close to the center generally have larger occurrence probability compared to those towards the border, providing users with an overview on the probability distribution of the anomalous events.

Flow Overview. The *flow overview* presents progression-level differences between normal sequences and the abnormal sequence using a flow-based visualization. As shown in Fig. 4(4), the flow of normal sequences are colored in light grey as background, with the flow of the abnormal sequence overlaid on the top. This view serves as a quick overview at the beginning of sequence comparison to illustrate how the progression of the anomalous record deviates from the normal group.

Anomalous Record View and Similar Record List. The *anomalous record view* (Fig. 4(6)) and the *similar record list* (Fig. 4(7)) provide access to raw event sequence data for the anomalous sequence and similar sequences, respectively. These low-level details provide detailed evidence to support interpretation **(T4)**.

5.5 Interactions

The ET^3 system includes several user interactions **(T6)** to support exploratory analysis.

Stage Merging. We leverage a recently proposed progression analysis technique [22] to segment the anomalous sequence into different stages. As illustrated in Fig. 4(c), stages are marked with line segments under the identifier of the time slots. Users can click on a stage identifier to merge or expand all visual elements in the main panel that align to the corresponding time slots. This interaction aims to reduce the length of sequences for a more efficient exploration while also providing a high-level summarization of an anomaly's progression stages.

Selecting and Filtering. Our system allows users to navigate the visualization and make more focused inspection through flexible data selection and filtering. For example, the system allows users to make selections of both individual sequences of interest or subgroups of sequences following particular progression patterns in the *sequence comparison* and *flow comparison* views, respectively. After a selection, the system reruns the comparison between the anomalous sequence and the selected normal sequences to update the comparison glyphs based on the analysis result.

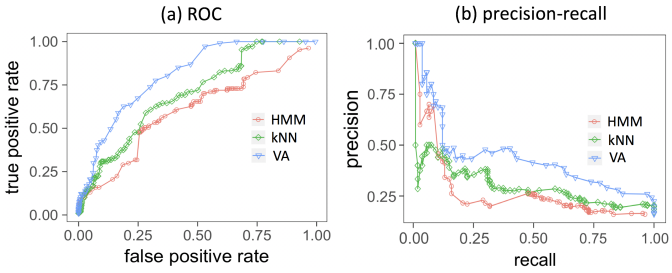

Fig. 5. Performance evaluation results of our VAE-based algorithm (VA) in comparison with two baseline methods (kNN, HMM). The (a) ROC curves and (b) precision-recall curves indicate that our approach (VA) effectively detects anomalies and outperforms the baseline methods.

The system incorporates three types of filters for users to tune the visualization result. This includes a probability filter and a support rate filter for supporting dynamic adjustment of the detection boundaries (as introduced in Sec. 5.3.3). In addition, a node filter allows control tune the *flow overview* and the *flow comparison view*. To prevent event anomalies in small populations from being filtered out, nodes in the flow diagram that are identified as abnormal are preserved regardless of the current filter threshold. Moreover, the system allows users to brush the *reconstruction view* to zoom in to a specific range of time slots to narrow the analysis scope (Fig. 4(e)).

Highlights and Tooltips. The system is equipped with linked-highlighting, which helps users to track the occurrence of event types across different views. Specifically, when users hover their mouse over an event, all visual elements representing the same event type will be simultaneously highlighted in all views. Moreover, when users select an individual sequence in the *sequence comparison view*, or a progression path in the *flow comparison view* or *flow overview*, all corresponding visual elements will be highlighted to mark the users' selection. Finally, descriptive tooltips (Fig. 4(f)) are triggered when hovering over any visual elements throughout the system.

Details-on-Demand. The system also supports a details-on-demand model of exploration. Apart from providing informative tooltips during the analysis, the low-level details displayed in the *similar record list* and the *sequence comparison view* are updated in response to changes in the users' selection in other views.

6 Evaluation

The ET^3 system provides a unique solution for detecting anomalous sequences within general event sequence datasets. The visualization is designed to support the interpretation of detected anomalies with visual comparisons. We assess the effectiveness of ET^3's analytical model and visualization design through a quantitative evaluation, three case studies, and qualitative feedback from domain experts.

6.1 Quantitative Evaluation

We compare the performance of our VAE-based anomaly detection algorithm (denoted as VA) with two baseline methods using an intrusion detection dataset, snd-cert [19]. The dataset consists of sequences of operating system calls that are labeled in terms of the system state (i.e., normal or hacked) when running these operations.

Baseline Methods and Evaluation Metrics. We select two representative baseline methods under the categories of kernel-based and Markovian anomaly detection techniques: Nearest Neighbor (kNN) [32] and Hidden Markov Model (HMM) [45]. Both methods have been shown efficient for detecting anomalies in event sequence data in previous research [7,39,47]. More specifically, the longest common subsequence (LCS) was used as the distance metric in kNN. We use standard information retrieval metrics (precision, recall, and ROC) to evaluate the performance of our approach and these two baseline methods. Because the number of positive and negative instances are imbalanced in the dataset, we use the precision-recall curves and ROC curves to comprehensively illustrate the performance of the algorithms.

Evaluation Results. Our algorithm outperforms the baseline methods as shown in Fig 5. The ROC plot (Fig 5(a)) illustrates that VA achieves higher true positive rates when the false positive rates remain

low (below 0.25) compared to the other two baseline methods. The precision-recall plot (Fig 5(b)) shows that VA had overall higher precision than the baseline methods. The results indicate that our approach can produce a higher quality set of suspicious sequences when compared to the baseline algorithms. Using the designed visualization, the system can further support the interpretation of detected anomalies.

6.2 Case Studies

In this section, we report the results of three real-world cases to demonstrate the capabilities of ET^3 in finding interpretable anomalies in general event sequence datasets.

6.2.1 Misuse of Prescription Drugs

We applied ET^3 to MIMIC [28], a publicly accessible critical care database with de-identified electronic health records for 46,520 patients with 12,487 event types in total. Due to the diversity of sequence progression for patients with different diseases, training with the entire database could introduce noise and produce inaccurate anomaly results. With this consideration, we selected a subgroup of 7,537 patients who were diagnosed with cardiovascular diseases to produce a more homogeneous set of sequence progressions for the training set.

Four cardiologists (**E1–E4**, 5–8 years of domain experience each) were invited to participate in our study. Prior to the study, the doctors were asked about expected patterns of anomaly and they expressed interests in exploring anomalous medical usage within the follow-up lab test results, based on which we extracted 87 event types under the category of *prescriptions* and *lab events*. The sequences were sent to the analysis module for training the model. 404 anomalous sequences were detected for subsequent analysis. The study session lasted approximately 1 hour, starting with a 10-minute introduction to the dataset and the system design, followed by a 5-minute demonstration of an example use case. After some practice, the doctors were asked to explore the analysis results with our system, demonstrate domain-relevant insights, and provide feedback on the system's usability. The system was projected to a large screen for all experts to inspect simultaneously, and one of the experts **(E1)** was responsible for operating the system. The experts were asked to think out loud and make comments at any time during the session. The entire study procedure was recorded and their comments are discussed in Section 6.3.

After a brief inspection of the *overview*, the experts chose a patient who was far away from the main cluster with a relatively high anomaly score (as shown in Fig. 4(1)), and then retrieved 105 similar patients with the distance to the mean sequence under 0.2 for subsequent analysis (as shown in Fig. 4(2)). After the *flow overview* was loaded, the experts quickly scrolled the view back and forth to get a big picture view of the major sequence progression paths. They found that the treatment plans for the patients were very similar, and speculated that all of these patients were suffering from epilepsy and diabetes as there was regular use of Phenytoin and Insulin along the clinical paths (as shown in Fig. 6(a)). They then took a careful look at the flow of the anomalous patient, and noticed that most events were in agreement with the major trend. However, the patient showed several exceptional events in the second and the third time slots (Fig. 4(g)). They decided to focus their analysis around these time slots. By splitting the sequence of the anomalous patient from others, the comparison glyphs uncovered suspicious events at each time slot. The experts briefly browsed the events displayed in the glyphs and noticed an abnormal lab event with a 0.59 anomaly score and 100% support rate, CK-MB (Fig. 6(i)). "This is a critical indicator for myocardial infarction", E1 said. The experts also found an event that was continuously missing in several time slots throughout the entire progression, Hydralazine (Fig. 4(b)). "This drug is mainly applied to patients with chronic heart failure" explained E2. "This may imply different causes of epilepsy. Both heart diseases can potentially cause epileptic seizure." The experts then decided to explore the redundant medicines highlighted in the comparison glyphs to investigate the differences in the treatment plan. Apart from several anesthetics used for pain relief, the doctors surprisingly found no medicines aimed directly at curing myocardial infarction. "This is unusual," E1 said, "It seems the patient was treated as a regular epileptic patient." They also found a type of medicine, Pheonobarbital(Fig. 6(ii)),

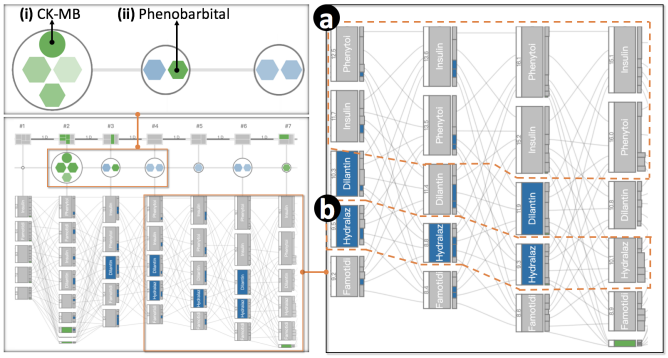

Fig. 6. The anomaly detection result of MIMIC dataset. The system identified major progression paths (a-b), from which the sequence anomaly deviates in (i) an anomalous lab test result and the (ii) misuse of a prescription drug.

having an anomaly score of 0.61 and 100% support rate, used only by the anomalous patient. "To my knowledge, Pheonobarbital is mainly used for neonatal and childhood seizures according to guidelines," said E3. "It is rare to see this drug prescribed for a 69-year-old man." E4 found this finding especially useful, as he commented: "It is a potential drug of abuse. Long-time usage can result in physical dependence, thus should be strictly controlled...I feel this system has great potential to be applied to monitor drug misuse."

6.2.2 Application Log Diagnosis

In our second case study, we applied ET^3 to a public application log data, Agavue [17]. It traces the application function calls invoked by user interactions in a suite of visualization tools embedded in Excel. The dataset contains 2,212 user sessions with 34 unique event types. The model detected 143 anomalous sequences in total. A software developer **(E5)** familiar with graphing in Excel was invited to participate in this study. We observed her actions and took notes on her comments and findings. A document illustrating the meaning of each event in the dataset was provided before the study to help the participant better understand her observations during the analysis.

After loading the dataset, the participant observed the *overview* and selected a sequence with a high anomaly score for subsequent analysis. She then brushed session sequences with a distance to the mean sequence under 0.2 to retrieve a group of 103 similar sequences and started the comparative analysis. By switching to the *flow comparison view*, the system revealed anomalous events in the *comparison glyphs* in time steps 3–5 and 9 (Fig. 1). The participant started the exploration from the highlighted anomalies in time steps 3–5. She decided to neglect the redundant *resize* (Fig. 1(a)) event as it is a common user action and has low support rate. She turned instead to time steps 4 and 5, which both showed redundant events for *toolBarToogle* (Fig. 1(b)). This indicates a user action of showing or hiding the toolbar. As she hovered on the event, a path in the flow of the similar records was highlighted, suggesting another sequence also had the *toolBarToogle* event at step 4 and 5 but was identified as normal. She thus decided to compare this sequence and the anomalous sequence. The participant clicked on the *toolBarToogle* event in the comparison view to select the normal sequence having this event (Fig. 1(1)), and the comparison glyphs alerted the participant to missing *toolTip* events (Fig. 1(c)). By observing the highlighted flow underneath, she realized that *toolTip* co-occurred with *toolBarToogle* in the normal sequence. Thus, she surmised that a tooltip should normally appear when a user hovers their mouse on toolbars. However, this did not show up in the sequence of the anomalous user. She then switched to the *sequence comparison view* to inspect the normal sequences individually and treemaps compounding the *toolBarToogle* and *toolTip* confirm her speculation.

The participant then focused on the missing events time steps 3–5, which were a series of *error* events (Fig. 1(d)). She noticed that error messages were frequently encountered in sequences of "normal" users but not the anomalous user. To find reasons, she clicked on the *error* event to narrow the comparison scope. As shown in Fig. 1(2), two

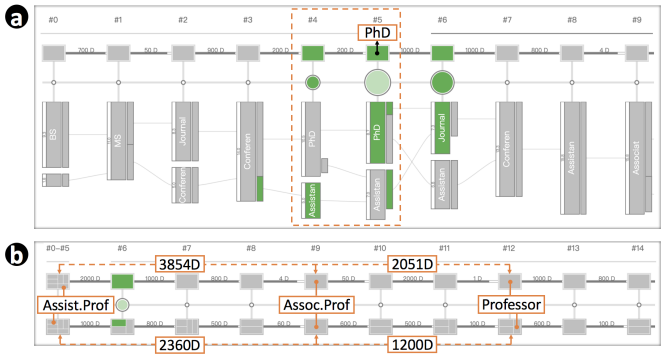

Fig. 7. Anomalies detected when analyzing the career paths of a group of scholars. ET^3 identified two anomalous patterns, which includes (a) a distinctive order of acquiring PhD degree and Assistant Professor, and (b) a slower promotion progress than usual.

missing events, *btnSetData* (Fig. 1(e)) and *bindFromPrompt* (Fig. 1(f)) simultaneously appear in the first stage and the following time steps, indicating that the lack of the two events prevented the anomalous sequence from having errors. The participant looked up these events in the documentation and found that they represent continuous application calls in response to the user requesting a customized data binding by pressing the *set data* button. She suspected that users might have attempted to bind data with an inappropriate format, which would lead to an unsuccessful graph. This also explained the redundant event, *readBoundData*, in the last time slot, indicating the anomalous user eventually bound data successfully. In general, this finding suggested that Agavue should provide more guidance when users try to arbitrarily bind data to visualizations.

6.2.3 Rare Career Path Detection

We also tested the capability of ET^3 in identifying anomalies within a small event sequence collection using a career path dataset [1]. The dataset contains 10 types of milestone events of 40 university professors, such as receiving degrees, publishing papers, and changing academic positions. We trained the anomaly detection model and received two anomalous career paths. A graduate student (**E6**) was invited to analyze the anomalous sequences and provide feedback. The participant selected one anomaly and a group of similar scholars identified as normal to start the comparison. He first looked into the stage analysis results using the *reconstruction view*. In combination with the *flow overview*, he identified the first stage from time step 0 to 5 as the degree obtainment period through a signature *PhD* event. Time steps afterward were designated as stage two, representing the career path after graduation. He then split the anomalous sequence from the flow. As shown in Fig. 7(a), the comparison glyphs from 4–5 highlighted two redundant events before the scholar's career path stage began. He inspected on the anomalous events at time step 4 and 5, which were obtaining PhD degree and acquiring assistant professor. Surprisingly, he found the anomalous scholar experienced these two events in an unexpected order which suggested the scholar became an assistant professor before obtaining a PhD degree. In contrast, the major flow in normal sequences goes from the *PhD* to *assistant professor*. Moreover, no link was found from *assistant professor* to *PhD* in the flow of normal sequences.

The participant then merged stage one and turned to analyze the career path in stage two. He showed great interest in the promotion of the assistant professor. He switched to the *summarization view* to compare the promotion time of the focal scholar with the rest of the scholars. As shown in Fig. 7(b), the *associate professor* events were all aligned at time step 9 and the *professor* event at time step 12. The participant noticed that the career path of the focal scholar was not as smooth as expected, as it took longer for the scholar to get granted associate professorship (3,854 days compared to 2,360 days on average) and get promoted from associate professor to professor (2,051 days compared to 1,200 days on average).

6.3 Feedback

This section reports subjective feedback collected from post-study interviews. We summarize the participants' comments around three themes: usefulness, system usability, and visualization design.

Usefulness. All participants respond positively when asked if the anomaly detection technique in ET^3 was useful for the analytical tasks in their area of expertise. The medical experts believed that ET^3 helps detect unusual treatment plans. **E1** also mentioned that "we typically check if the clinical decisions are in accordance with guidelines with statistical methods, which is adequate in identifying numerical anomalies such as overdose of antibiotics or anesthetic, however, fail to discover complex anomalies." **E4** added: "anomalies in medical data can be a single event or a combination of events." He felt involving sequential context in the anomaly analysis can help doctors "discover more complex anomalous patterns and find the reasons for the anomalies." **E2** expressed a desire for this system to support predictive analysis: "if the system can detect anomalous trends ahead, we can take actions to put the anomalous patient back on track." **E3** found comparing the anomaly with similar patients especially useful as a way to "provide evidence for the detection results derived from the machine." **E5** felt that "diagnose errors with context makes the result easier to understand," and suggested that we extend the system's capability to support analysis of streaming data for tasks such as "network behavior monitoring".

System Usability. According to the participants' feedback, ET^3 is generally easy to use. All three experts (**E1,5,6**) that navigated the system expressed their appreciation for the system's workflow and felt it "easy to follow." **E5** said: "Following the workflow, we can have the overview first and drill down to specific anomalous events step by step." **E6** also commented on the comprehensiveness of the system: "the system covers information from various aspects. Each view displays the data from a different angle, but altogether they manage to integrate nicely." The medical expert (**E1**) felt that the *flow comparison view* was most informative, as "it gives a clear summarization on patients' clinical pathways." He felt the *sequence comparison view* less useful because the raw medical records are generally complex and hard to explore. However, he agreed that "it could help validate findings when the exploration space is narrowed to a small sub-population after selection." E2 also commented that "..when dealing with a large group of sequences, analyzing anomalous sequences one after another may be time-consuming." This suggests a future direction of supporting the analysis of multiple anomalous sequences simultaneously.

Visualization and Interaction Design. All participants agreed that all visualization in ET^3 were easy to understand. Specifically, **E3** felt the design of the *comparison glyph* made sense: "It covers two critical points of how anomalous the event is, and if the analysis result is confident." He also suggested that "a measure can be defined to combine these two attributes as they seem correlated". **E2** also mentioned that the dynamic selection of subgroups is useful, as he explained: "The analysis result of anomaly can be different in different sub-populations. This interaction can help us explore the reasons and the effect of anomalies among subgroups of patients efficiently."

7 Conclusion

We have presented ET^3, a visual analysis technique designed to support visual anomaly detection in event sequence data. ET^3 incorporates an unsupervised VAE-based anomaly detection model to identify anomalous sequences and events in an interpretable manner. Based on the anomaly detection result, a visualization system with multiple coordinated views and rich interactions is provided to facilitate interpretation via one-to-many sequence comparison. We evaluate the effectiveness and usefulness of ET^3 through a quantitative comparison of the performance of our proposed algorithm, three case studies with real-world datasets, and domain expert interviews. The study results illustrate the strengths of ET^3 and shed light on several directions for future work. These include enabling predictive anomaly analysis, integrating an associative measurement for event abnormality that considers both anomaly score and support rate, and supporting the analysis of multiple anomalous sequences simultaneously.

REFERENCES

[1] Professors dataset, 2016. Human-Computer Interaction Lab, University of Maryland. Retrieved from `https://eventevent.github.io/`.

[2] J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2:1–18, 2015.

[3] D. Antonelli, G. Bruno, and S. Chiusano. Anomaly detection in medical treatment to discover unusual patient management. *IIE Transactions on Healthcare Systems Engineering*, 3(2):69–77, 2013.

[4] A. Bock, A. Pembroke, M. L. Mays, L. Rastaetter, T. Ropinski, and A. Ynnerman. Visual verification of space weather ensemble simulations. In *IEEE Scientific Visualization Conference*, pp. 17–24. IEEE, 2015.

[5] R. J. Bolton, D. J. Hand, et al. Unsupervised profiling methods for fraud detection. *Credit Scoring and Credit Control VII*, pp. 235–255, 2001.

[6] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *ACM SIGMOD Record*, vol. 29, pp. 93–104. ACM, 2000.

[7] S. Budalakoti, A. N. Srivastava, R. Akella, and E. Turkov. Anomaly detection in large sets of high-dimensional symbol sequences. 2006.

[8] N. Cao, C. Lin, Q. Zhu, Y.-R. Lin, X. Teng, and X. Wen. Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):23–33, 2018.

[9] J. Chae, D. Thom, H. Bosch, Y. Jang, R. Maciejewski, D. S. Ebert, and T. Ertl. Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition. In *IEEE Visual Analytics Science and Technology*, pp. 143–152. IEEE, 2012.

[10] R. Chalapathy and S. Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

[11] R. Chalapathy, A. K. Menon, and S. Chawla. Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*, 2018.

[12] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing surveys*, 41(3):15, 2009.

[13] H. Chen, S. Zhang, W. Chen, H. Mei, J. Zhang, A. Mercer, R. Liang, and H. Qu. Uncertainty-aware multidimensional ensemble data visualization and exploration. *IEEE Transactions on Visualization and Computer Graphics*, 21(9):1072–1086, 2015.

[14] F. Du, C. Plaisant, N. Spring, and B. Shneiderman. Eventaction: Visual analytics for temporal event sequence recommendation. *Proceedings of the IEEE Visual Analytics Science and Technology*, 2016.

[15] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[16] H. Fanaee-T and J. Gama. Tensor-based anomaly detection: An interdisciplinary survey. *Knowledge-Based Systems*, 98:130–147, 2016.

[17] D. Fisher. Agavue event data sample, 2016. Version of August 10, 2016, Microsoft Research. Retrieved from `https://eventevent.github.io/`.

[18] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda. Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of the International Conference*, p. 8. ACM, 2010.

[19] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for unix processes. In *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 120–128. IEEE, 1996.

[20] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011.

[21] J. Guerra-Gómez, M. L. Pack, C. Plaisant, and B. Shneiderman. Visualizing change over time using dynamic hierarchies: Treeversity2 and the stemview. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2566–2575, 2013.

[22] 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(1):417–426, 2019.

[23] 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.

[24] M. Hauskrecht, I. Batal, M. Valko, S. Visweswaran, G. F. Cooper, and G. Clermont. Outlier detection for patient monitoring and alerting. *Journal of Biomedical Informatics*, 46(1):47–55, 2013.

[25] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[26] T. Idé and H. Kashima. Eigenspace-based anomaly detection in computer systems. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 440–449. ACM, 2004.

[27] Z. Jin, J. Yang, S. Cui, D. Gotz, J. Sun, and N. Cao. Carepre: An intelligent clinical decision assistance system. *arXiv preprint arXiv:1811.02218*, 2018.

[28] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3:160035, 2016.

[29] J. Kehrer and H. Hauser. Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, 2013.

[30] J. Kehrer, H. Piringer, W. Berger, and M. E. Gröller. A model for structure-based comparison of many categories in small-multiple displays. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2287–2296, 2013.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] Y. Liao and V. R. Vemuri. Use of k-nearest neighbor classifier for intrusion detection. *Computers & Security*, 21(5):439–448, 2002.

[33] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *IEEE International Conference on Data Mining*, pp. 413–422. IEEE, 2008.

[34] M. Müller. Dynamic time warping. *Information Retrieval for Music and Motion*, pp. 69–84, 2007.

[35] G. Münz, S. Li, and G. Carle. Traffic anomaly detection using k-means clustering. In *GI/ITG Workshop MMBnet*, pp. 13–14, 2007.

[36] D. Park, Y. Hoshi, and C. C. Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, 2018.

[37] A. Patcha and J.-M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, 2007.

[38] A. Qayyum, M. Islam, and M. Jamil. Taxonomy of statistical based anomaly detection techniques for intrusion detection. In *Proceedings of the IEEE Symposium on Emerging Technologies*, pp. 270–276. IEEE, 2005.

[39] Y. Qiao, X. Xin, Y. Bin, and S. Ge. Anomaly intrusion detection method based on hmm. *Electronics Letters*, 38(13):663–664, 2002.

[40] L. Ravindranath, J. Padhye, S. Agarwal, R. Mahajan, I. Obermiller, and S. Shayandeh. Appinsight: mobile app performance monitoring in the wild. In *Presented as part of the 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*, pp. 107–120, 2012.

[41] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection*, vol. 589. John wiley & sons, 2005.

[42] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, pp. 130–144. IEEE, 2000.

[43] D. Thom, H. Bosch, S. Koch, M. Wörner, and T. Ertl. Spatiotemporal anomaly detection through visual analysis of geolocated twitter messages. In *IEEE Pacific Visualization Symposium*, pp. 41–48. IEEE, 2012.

[44] Y. Tu and H.-W. Shen. Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1286–1293, 2007.

[45] Y. Xie and S.-Z. Yu. A large-scale hidden semi-markov model for anomaly detection on user browsing behaviors. *IEEE/ACM Transactions on Networking*, 17(1):54–65, 2009.

[46] G. Xiong, J. Cheng, X. Wu, Y.-L. Chen, Y. Ou, and Y. Xu. An energy model approach to people counting for abnormal crowd behavior detection. *Neurocomputing*, 83:121–135, 2012.

[47] X. Zhang, P. Fan, and Z. Zhu. A new anomaly detection method based on hierarchical hmm. In *Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 249–252. IEEE, 2003.

[48] J. Zhao, N. Cao, Z. Wen, Y. Song, Y.-R. Lin, and C. Collins. # fluxflow: Visual analysis of anomalous information spreading on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1773–1782, 2014.

[49] J. Zhao, Z. Liu, M. Dontcheva, A. Hertzmann, and A. Wilson. Matrixwave: Visual comparison of event sequence data. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pp. 259–268. ACM, 2015.