Published in final edited form as:

KDD. 2017 August ; 2017: 2111–2120. doi:10.1145/3097983.3098174.

A Data-driven Process Recommender Framework

Sen Yang[1], **Xin Dong**[1], **Leilei Sun**[2], **Yichen Zhou**[1], **Richard A. Farneth**[3], **Hui Xiong**[1], **Randall S. Burd**[3], and **Ivan Marsic**[1]

[1]Rutgers University, NJ

[2]Tsinghua University, PR China

[3]Children's Nat'l Medical Center, Washington, DC

Abstract

We present an approach for improving the performance of complex knowledge-based processes by providing data-driven step-by-step recommendations. Our framework uses the associations between similar historic process performances and contextual information to determine the prototypical way of enacting the process. We introduce a novel similarity metric for grouping traces into clusters that incorporates temporal information about activity performance and handles concurrent activities. Our data-driven recommender system selects the appropriate prototype performance of the process based on user-provided context attributes. Our approach for determining the prototypes discovers the commonly performed activities and their temporal relationships. We tested our system on data from three real-world medical processes and achieved recommendation accuracy up to an F1 score of 0.77 (compared to an F1 score of 0.37 using ZeroR) with 63.2% of recommended enactments being within the first five neighbors of the actual historic enactments in a set of 87 cases. Our framework works as an interactive visual analytic tool for process mining. This work shows the feasibility of data-driven decision support system for complex knowledge-based processes.

Keywords

Process Recommender System; Process Trace Clustering; Process Prototype Extraction; Emergency Medical Process Analysis

1 INTRODUCTION

Contemporary information systems, such as personal calendars and electronic health records (EHR), often record activity logs. Process mining techniques attempt to extract nontrivial knowledge and insights from activity logs and use them for further analyses [1]. Process mining techniques have been applied in practical problems, assisting in visualizing, interpreting and diagnosing processes [1]. Existing recommender systems have not been developed based on process mining. Our current work presents such a bridge. We are designing a data-driven process analysis and recommender system that can provide contemporaneous recommendations of process steps and help with retrospective analyses of the process. Our approach relies on mining historic data to uncover the potential association between the way of enacting a process and contextual attributes. If association tests are

significant, we train a recommender system to output a prototypical enactment for the given context attributes.

Unlike most recommender systems that propose one or few next steps at a time, our system initially recommends all steps at once. Although it may not be feasible for the performers to study and follow a long list of steps, this recommendation can be used at runtime to automatically verily the process compliance and detect omitted steps and other process errors. Our framework has two stages: process analysis and process recommendation (Fig. 1(a)). Process analysis includes: (1) clustering of historic traces based on similarity; (2) determining the cluster prototypes that represent the established process enactment for each cluster; (3) regression analysis to explore the correlation between cluster membership and context attributes; and (4) interactive visualization and statistical analysis of process traces. The recommendation stage includes: (1) predicting the cluster to which the given trace belongs based on the observed context attributes, and (2) displaying the prototype of the predicted cluster as the recommended enactment.

Key technical challenges for this system include measuring the similarity of process traces and determining the cluster prototypes. Similarity measurement strongly affects the results of trace clustering and plays a key role in our system. Several metrics of trace similarity exist but suffer from either inaccurate measurement because of timeline stretching needed to normalize the trace duration and compute the overlap between the traces, or information loss from forced sequencing of concurrent activities needed to apply edit distance or pattern-based distance [2]. Another challenge is determining a prototype that represents the recommended sequence of steps for each cluster. Our contributions include:

- A novel metric of pairwise similarity between process traces based on time warping. Unlike existing similarity metrics (edit distance, pattern-based distance, and Euclidean distance based on a normalized timeline), our approach incorporates the time information while correcting for temporal differences between the same activities in different process traces, such as different start times, idle times and duration of performance. Our approach also handles concurrent activities and parallel activities for which the order of performance is irrelevant.
- A novel approach for determining a prototype for a cluster of process traces. Our prototype captures the established enactment for a given context and considers the temporal relationships between activities. It achieves a higher average similarity to process traces in its cluster than the cluster medoid.
- A data-driven recommender system that selects a representative enactment based on user-provided context attributes. We tested our system on data from three real-world medical processes and achieved high recommendation accuracy.

2 RELATED WORK

Complex knowledge-based processes are usually performed based on domain knowledge and standard protocols. For example, for trauma resuscitation the Advanced Trauma Life Support (ATLS) protocol [3] suggests the workflow based on treatment priorities: Airway

→ Breathing → Circulation → (Neurological) Disability. Clarke et al. [4] and Fitzgerald et al. [5] developed computer-aided decision support that recommends next steps to reduce human errors. These systems rely on rules manually specified by domain experts, lack generalizability, and are subject to human bias. We present an automatic, data-driven, label-free framework for process analysis and recommendation.

Our framework incorporates three main techniques: similarity metrics for process traces, trace clustering algorithms, and cluster prototype extraction. These techniques have been well studied in analysis of time series [6], but are not applicable to process data. Unlike time series with numerical values, process data is typically categorical, representing different activity types and their properties. Different process datasets may have very different features and no rule exists to decide similarity between traces of process enactment. Common similarity metrics include edit or Levenshtein distance [7,8] and pattern-based similarity, e.g., n-gram [9,10]. Both metrics accept as input only process traces represented as sequences, which requires that concurrent activities are sequenced (e.g., by activity start time) and that temporal information on activity duration and idle times is ignored. Forestier et al. [11,12] proposed dynamic time warping (DTW) as a similarity metric for process traces. The DTW, however, cannot handle concurrent activities, does not consider idle time intervals, and has other issues when used for process traces [13]. In addition, Forestier et al. considered processes that are mostly sequential (non-concurrent), with no activities for which the order of performance is irrelevant. To address these challenges, we introduce a novel similarity metric based on time warping that incorporates temporal information, such as activity start time, performance duration, and idle intervals.

Hierarchical clustering has been commonly used for process trace clustering [11][14-16]. This algorithm does not need a predefined number of clusters and produces a visually intuitive dendrogram (tree diagram). Its main limitation is its computational complexity, generally $O(n^2\log(n))$ where n is the number of traces, which makes it too slow for large datasets. We implemented hierarchical clustering in our framework as well as two other state-of-the-art clustering algorithms.

Cluster prototype candidates can be determined using different techniques. A widely used cluster centroid represents the cluster center with a minimum distance to other points in the cluster, e.g., sum-squared distance [6]. For categorical and event-based data, however, the notion of a “center” may not apply [6]. For example, the centroid of categorical data {orange, apple, banana} cannot be determined. An alternative is the cluster medoid as the most representative data object in the cluster—an existing object that has a minimal average dissimilarity to all other objects in its cluster. The medoid, however, may not be adequate when no “suitable” representative exists in the cluster. Another kind of prototype is the consensus sequence, a sequence of commonly observed activities found by aligning many process traces [2,13]. The consensus sequence, however, represents only the order of performance without temporal information. We introduce a novel approach for cluster prototype extraction that incorporates temporal information.

3 TERMS AND DEFINITIONS

A performance of a process can be captured with activity codes and timestamps. We represent each **activity** by its type and performance time (Fig. 1(b)) denoted as $A = \{A^{\text{type}}, A^{\text{ts}}, A^{\text{te}}]$, where A^{type} is the activity type, A^{ts} is the start time, and A^{te} is the end time. A **process case** $c = \{\text{id}, \boldsymbol{x}, \boldsymbol{T}\}$ is an instance of process performance. It is indexed with a unique case *id* and consists of the *trace* $\boldsymbol{T}$ which is a vector of performed activities (internal information), and the vector $\boldsymbol{x}$ of *context attributes* (external information). An ith **process trace** is represented as $\mathbf{T_i} = [A_{i1},..., A_{ik}]$, where k is the trace length (number of performed activities). To make explicit concurrent activities, we use a matrix representation of traces as $\boldsymbol{T}_i = [\boldsymbol{p}^i_1, \ldots, \boldsymbol{p}^i_{k^i}]$, where the duration of ith trace is discretized into k^i time units and in each time unit m the vector $\boldsymbol{p}^i_m = [a^1, \ldots, a^\ell]$ represents the execution status of all ℓ activity types. If an activity of type a^j is being performed during time m, then $a^j = 1$ and $a^j = 0$ otherwise. The magnitude of each activity vector is $|\boldsymbol{p}^i_m| = \Sigma_j |a^j|$ **Context attributes** (or external attributes) record the contextual information of a process case, such as the patient demographics (Fig. 1(c)) in a vector $\boldsymbol{x} = [x_1, ..., x_d]^T$ of d observed attributes x_i. By associating context attributes with step-by-step activities based on historic data, we can recommend the best process enactment for given attributes.

A **process trace cluster** $\boldsymbol{C} = [\boldsymbol{T_1},..., \boldsymbol{T_c}]$ is a group of c traces that are similar in terms of type, activity performance order and times. The cluster membership is determined by information internal to process traces. A **prototype trace** of a cluster is the most representative or typical process enactment for this cluster. This representative enactment can be an actually observed trace (an exemplar) or derived from other traces in the cluster. Cluster prototypes summarize the cluster information and highlight the commonalities of the process traces, which can help visualize and compare the differences between different clusters.

A **recommended process trace** is determined using both internal and contextual information of historic traces to find a standardized process performance. This trace can be used to guide the process performance or verify the process compliance and detect omitted steps and other process errors.

4 METHOD

Our framework performance (i.e., recommendation accuracy) does not depend as much on the recommender model as on the ability to capture significant commonalities between process performances using a similarity metric and clustering, as well as on determining the proper cluster prototype. Therefore, we focus on the similarity metric, clustering, and prototype extraction for assessing the performance.

4.1 Trace Similarity based on Time Warping

The process traces we considered are not simple sequences just recording activity type and the order of their performance, but concurrent timelines showing the performance status of

each activity type over time. Pairwise comparison of these composite traces is challenging. An effective similarity metric should combine (*i*) intrinsic activity likeness, e.g., some activities are mutually substitutable, (*ii*) activity performance time, (*iii*) relative order of performance, and (*iv*) temporal variation between different performances. The temporal variation has several causes, such as activities initiated at different times relative to the process start, performed at different speeds, omitted or repeated. The same activities may have different temporal characteristics in different traces and traces may have different duration. Although several similarity metrics exist for temporal sequences [7,8] [11,12] [14-16], none satisfies the above requirements.

We introduce a novel similarity metric for complex process traces using timeline warping to determine the optimal pairwise alignment (Fig. 2). Our metric considers both the sequential order and temporal overlaps of activities during this optimization. We define the similarity between traces $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ as:

$$s(i,j) = \frac{\left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right|}{\left|\boldsymbol{T}_i \cup \boldsymbol{T}_j\right|} \quad (1)$$

where $|\boldsymbol{T}_i| = \Sigma m |\boldsymbol{p}^i{}_m|$ is the total performance time of activities in trace $\boldsymbol{T}_i$ and $\boldsymbol{p}^i{}_m$ is the vector of performance status of all activities in mth time unit. $|\boldsymbol{T}_i \cap \boldsymbol{T}_j|$ is the time when both traces had same activities performed and $|\boldsymbol{T}_i \cup \boldsymbol{T}_j|$ is the time when one or both traces had same activities performed. If we define $|\boldsymbol{T}_i \otimes \boldsymbol{T}_j|$ as the time when only one trace had activities performed, then the total active time in a pair of traces is:

$$\left|\boldsymbol{T}_i\right| + \left|T_j\right| = \left|\boldsymbol{T}_i \cup \boldsymbol{T}_j\right| + \left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right| \quad (2)$$

and

$$\left|\boldsymbol{T}_i\right| + \left|T_j\right| = \left|\boldsymbol{T}_i \otimes \boldsymbol{T}_j\right| + 2\left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right| \quad (3)$$

By combining these equations, the similarity of $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ is:

$$s(i,j) = \frac{\left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right|}{\left|\boldsymbol{T}_i\right| + \left|\boldsymbol{T}_j\right| - \left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right|} = \frac{\left|\boldsymbol{T}_i\right| + \left|\boldsymbol{T}_j\right|}{\left|\boldsymbol{T}_i\right| + \left|\boldsymbol{T}_j\right| - \left|\boldsymbol{T}_i \cap \boldsymbol{T}_j\right|} - 1 \quad (4)$$
$$= \frac{2\left|\boldsymbol{T}_i\right| + 2\left|\boldsymbol{T}_j\right|}{\left|\boldsymbol{T}_i\right| + \left|\boldsymbol{T}_j\right| + \left|\boldsymbol{T}_i \otimes \boldsymbol{T}_J\right|} - 1$$

The only variable term in this equation during warping alignment of two traces is $|\boldsymbol{T}_i \otimes \boldsymbol{T}_j|$. The optimal warping path between $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ is $\boldsymbol{P}^{ij} = \left\{\boldsymbol{p}^{ij}_{mn}\right\} = \left\{\left(\boldsymbol{p}^i_m, \boldsymbol{p}^j_n\right)\right\}$, which is the solution to this optimization problem:

$$\underset{p^{ij}}{\operatorname{argmin}} \left| \boldsymbol{T}_i \otimes \boldsymbol{T}_j \right| = \sum_{m,n} \left| (\boldsymbol{p}^i_m - \boldsymbol{p}^j_n)(\boldsymbol{J}_\ell - \boldsymbol{S}^a)\boldsymbol{w} \right| \quad s.t. \bigcup_m \boldsymbol{p}^i_m \in \boldsymbol{T}_i \quad \text{and} \quad \bigcup_n \boldsymbol{p}^j_n \in \boldsymbol{T}_j \tag{5}$$

where $\boldsymbol{w} = [w_1, \ldots, w_l]^T$ is a vector of weights indicating that some activities are more important than others. The weight can be any positive real number and the default is 1. When the weights are included, the trace magnitude is redefined as $|\boldsymbol{T}_i| = \Sigma_m |\boldsymbol{p}^i_m \boldsymbol{w}|$. The ℓ-by-ℓ matrix $\boldsymbol{S}^a(i,j) \in [0,1]$ represents the degree to which any pair of ℓ activity types are substitutable and $\boldsymbol{S}^a(i,j) = 1$ when activity types a^i and a^j are identical. An ℓ-by-ℓ matrix $\boldsymbol{J}_l$ of all ones is used to determine the distance between pairwise activity types as $\boldsymbol{J}_l - \boldsymbol{S}^a$. The weights and substitutability information are optional and may be given by domain experts when appropriate. Otherwise, they will default to a vector of ones and an identity matrix, respectively. Examples illustrate the influence of activity weight (Fig. 3(a),(c)) and substitutability (Fig. 3(a),(d)). Eq. (5) can be solved similarly as Levenshtein distance [7] using dynamic programming with a novel *score function*:

$$t^{ij}(g,h) = \begin{cases} -\sum_{m=0}^{g} \left| \boldsymbol{p}^i_m \boldsymbol{w} \right| - \sum_{n=0}^{h} \left| \boldsymbol{p}^j_n \boldsymbol{w} \right| - \epsilon, & \text{if } \min(g,h) = 0 \\ \max \begin{cases} t^{ij}(g-1,h-1) - |(\boldsymbol{p}^i_g - \boldsymbol{p}^j_h)(\boldsymbol{J}_\ell - \boldsymbol{S}^a)\boldsymbol{w}| \\ t^{ij}(g-1,h) - |\boldsymbol{p}^i_g \boldsymbol{w}| - \epsilon \\ t^{ij}(g,h-1) - |\boldsymbol{p}^j_h \boldsymbol{w}| - \epsilon \end{cases} \end{cases} \tag{6}$$

The score function $t^{ij}(g,h)$ is defined for alignment costs of two time units $\boldsymbol{p}^i_g$ and $\boldsymbol{p}^j_h$. For aligning traces $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$, we define the (k^i+1)-by-(k^j+1) score matrix $\boldsymbol{t}^{ij}$. The time-penalty vector $\boldsymbol{\epsilon} = [\epsilon, \epsilon, \ldots, \epsilon]^T \in \mathbb{R}^{1 \times k}$ is designed to penalize excessive warping of the timeline (grayed out bottom rows of traces in Fig. 3(a)). When $\epsilon = 0$, the timeline can be warped without cost, which may declare a short trace similar to a long trace. Constant ϵ can be heuristically set to the reciprocal of the standard deviation of case duration. When time penalty $\boldsymbol{\epsilon}$ is applied, the trace magnitude is redefined as $|\boldsymbol{T}_i| = \Sigma_g |\boldsymbol{p}^i_g \boldsymbol{w}| + \epsilon$. The above problem is a combinatorial optimization of inteival data. We first discretize the time axis and then use a time warping algorithm to find the optimal warping path $\boldsymbol{P}^{ij}$ and similarity s_{ij}. Algorithm 1 (TwS-PT) shows our approach for calculating the similarity of process traces.

4.2 Clustering Process Traces

To determine the recommended enactments from a large number of process traces, we clustered the traces. Exemplar-based clustering (EC) is an important category of clustering algorithms. These algorithms first select exemplars (representative points) from the whole dataset and then assign the remaining objects to their nearest exemplar. EC includes classic

clustering algorithms, like *K*-means and *K*-medoids, and recent methods, like Affinity Propagation (AP) [17] and Density Peaks based Clustering (DPC) [18]. We used Hierarchical Clustering, AP, and DPC.

Selecting the number of clusters is a difficult and well-known problem. Our method for setting this number is motivated by an intuition about cluster perception. A set of data points projected onto a similarity space observed from distance would appear as having fewer clusters than when observed up close. We propose that the number of clusters that remains stable over the greatest range of observation granularities represents the most probable structure of the dataset. We used AP clustering to analyze how the number of clusters varies with perception granularity. In methods like *K*-means, *K*-medoids, and spectral clustering, the number of clusters K is specified by the user. Although a similar parameter (preference p) is specified in AP clustering, the selection of p is more robust than that of K, as p linearly controls the perception granularity. The number of clusters increases with p and depends on the number of input objects [17]. We used p^c (p coefficient) to avoid the dependence on the number of objects:

$$p = \text{mean}(\boldsymbol{S}) - p^c \cdot N \quad (7)$$

where $\boldsymbol{S}$ is the similarity matrix of traces and N is the number of traces. Algorithm 2 summarizes our approach for selecting the number of clusters using the AP clustering algorithm (NumC-AP).

In Algorithm 2, γ is the increment of p^c and N^c is the number of clusters. We used synthetic data to show how NumC-AP works (Fig. 4). In the first example, points are distributed into four groups (Fig. 4(1.a)). The NumC-AP results show how the number of clusters changes with p^c from N to 1 (Fig. 4(1.b)). The proper number of clusters determined by NumC-AP is 4 and the second best choice is 2 clusters (Fig. 4(1.c)) as they best reflect the actual distribution of data points (Fig. 4(1.a)). Changing the distribution of the synthetic data causes the optimal number of cluster to change accordingly (Fig. 4(2)).

4.3 Determining the Cluster Prototype

After trace clusters are determined, a step-by-step prototype trace representing the recommended enactment is identified for each cluster. In the past, the medoid or a consensus sequence have been used as process prototypes. Because our traces contain concurrent activities that vary in the order of performance and temporal characteristics, existing methods cannot provide representative prototypes for our application. We developed an approach for determining cluster prototypes in three steps: (1) discovering the time-warped prototype using time warping paired with a divide-and-conquer strategy (a method of dividing the problem into recursively conquerable subproblems used, for example, in Quicksort); (2) unwarping the timeline to find the prototype; and (3) filtering and repairing the prototype for easier interpretation. Given a cluster $\boldsymbol{C}$ of traces, we first build a guide tree t (a dendrogram) using hierarchical clustering with the Ward's method linkage criterion [19]. The time-warped cluster prototype $\boldsymbol{q}$ is then solved recursively from the leaves to the root of

the guide tree (Fig. 5(b)). At each step, $\boldsymbol{q}$ is calculated pairwise from process traces by summing up their aligned results (Fig. 5(a)).

$$\boldsymbol{q} = \boldsymbol{T}^{i,j} = \overline{\boldsymbol{T}}_i + \overline{\boldsymbol{T}}_j = [\boldsymbol{p}_1^i + p_1^j\, \boldsymbol{p}_2^i + \boldsymbol{p}_2^j \cdots \boldsymbol{p}_k^i + \boldsymbol{p}_k^j] \quad (8)$$

where $\overline{\boldsymbol{T}}_i$ and $\overline{\boldsymbol{T}}_j$ denote traces aligned using Algorithm 1 and k is the length of the warped timeline. The time penalty vector $\boldsymbol{e}$ is set to $[0.2, ..., 0.2]^T$ (bottom rows in Fig. 5(a)). The penalties start as equal for the original traces so during alignment $\boldsymbol{e}$ can capture whether a warped time unit was frequently aligned or only existed in few cases. The summed $\boldsymbol{e}$ in $\boldsymbol{q}$ in the root of guide tree t can guide the time unwarping by its values in each time unit (Fig. 5(c)). For example, the long yellow bar in the bottom row of Fig. 5(a), between time 10 and 25 in $\boldsymbol{T}^{(1,3,2),4}$ comes from trace $\boldsymbol{T}_3$ which has a long idle period in the middle. For easier interpretation, we simplify $\boldsymbol{q}$by thresholding out the rare activities (Fig. 5(d)). To this aim, we define the *support* of a time cell a_{ij}^q as:

$$sup = a_{ij}^{\boldsymbol{q}} \Big/ c \quad (9)$$

where c is the number of traces in the cluster; i is the ith row (also ith activity type) of $\boldsymbol{q}$; j is the jth column (also jth time unit) of $\boldsymbol{q}$. The time unit is set to 1 when its support is greater than a threshold α and 0 otherwise, where α is by default set to 0.5. A potential drawback of this thresholding strategy is that it cannot capture frequent but sparsely distributed activities. To address this problem, we estimated the activity's frequency and included frequent activities (unique freq ≥ 0.5 in the clusters' cases) back into the prototype at the most likely position. This adjustment was done because the sparsely distributed rare activities may be aligned to several different positions during the prototyping. The thresholding removed them from consideration during warping, and left them to reincorporate more appropriately later. Another problem is, as time units are independent and discrete, activitytime cells of an activity may be fragmented after alignment and filtering (e.g., activity C in Fig. 5(d)). This fragmentation occurs because the time axis is discretized, a continuous activity is sliced into discrete slices and each slice is aligned independently with the corresponding time slice in other traces. When the slices of a continuous activity are independently aligned with other traces, the alignment may introduce gaps between the slices (e.g., activity C in in Fig. 5(d)) because in another trace the same activity was performed with an interruption or because a concurrent activity forced this fragmentation to achieve higher similarity score. We apply a repair to mitigate this problem by moving the smaller fragment to merge with the large one and close the gap if the gap is smaller than a time threshold β, which can be set as the mean value of all activity durations. We move the smaller fragment to the larger one since this repair has a smaller cost. Our procedure for extracting cluster prototype is summarized as Algorithm 3.

4.4 The Recommender Model

We chose to use regression model for our recommender system rather than a complex model (e.g., SVM or neural networks), as the statistical analysis (e.g., significance test) in regression model can help us easier interpret the correlations between data cluster membership and context attributes.

The goal of our logistic regression model is to leverage a set of n process cases to design a classifier that can distinguish between $m \geq 2$ clusters given context attributes $\boldsymbol{x}$. The cluster label of a process trace is encoded as $\boldsymbol{y} = [y^{(1)}, y^{(2)}, \ldots, y^{(m)}]^T$ where $y^{(i)} = 1$ if $\boldsymbol{x}$ is the context information of a trace that belongs to cluster i and $y^{(i)} = 0$ otherwise. The n process cases can then be represented as $\mathcal{S} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$. By default, we define the last class (the mth cluster) as the reference categoiy, against which logits of the first $m - 1$ categories are compared. Our logistic regression was trained with L2 regularizer:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \left[\sum_{j=1}^{n} \log P(\boldsymbol{y}_j \mid \boldsymbol{x}_j, \boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|^2 \right] \quad (10)$$

where $\boldsymbol{\beta}$ are regression coefficients for context attributes and λ is the ridge estimator of L2 regularizer. To find which attributes are associated with cluster membership, we used the Wald test [20] for logistic regression and a significance level at <0.05.

To generate recommendations, our system works by taking a new context attribute set $\boldsymbol{x}'$ (given by the user) and outputs a recommended enactment. The trained regression model selects the cluster class label $\boldsymbol{y}$ that maximizes the likelihood function:

$$\boldsymbol{y} = \arg\max_{\boldsymbol{y}} P(\boldsymbol{y} \mid \boldsymbol{x}', \hat{\boldsymbol{\beta}}) \quad (11)$$

Our system then returns the prototype of the most probable cluster as the recommended enactment. Because not all contextual attributes are good predictors, we used only statistically significant attributes to improve the recommendation accuracy. If no attribute was found as significant, all attributes are considered. Our framework was implemented as a web app (VIT-PLA, Fig. 6) using D3.js, Bootstrap, JSP, Java, and includes interactive visual functions.

5 EXPERIMENTAL RESULTS

We demonstrated the use of our framework with three real-world logs and evaluated the performance of different techniques.

5.1 Real World Medical Process Datasets

Datasets from three medical processes, collected in the emergency department of Children's National Medical Center, a level 1 pediatric trauma center in Washington, DC, were used for evaluating our framework (Table 1):

Tracheal Intubation Data: Ten context attributes are of three types: (a) patient demographics: age (<24 months, 24-96, >96), gender, height, weight, body mass index (BMI); (b) provider attributes: intubator's medical role (emergency medicine attending, anesthesia resident, etc.); and (c) event attributes: night/day, emergency/pre-arrival, direct-laryngoscopy/video-laryngoscopy and reason for intubation (seizure, respiratory distress, altered mental status—AMS).

Trauma Resuscitation Data: The trauma resuscitation is performed by a trauma team comprised of several physicians, nurses and ancillary medical staff, all working concurrently. Each case was coded with 17 context attributes of two types: (a) patient demographics: age, race, gender, injury type, injury severity score, pre-arrival intubation, mental status, body region injured (e.g., head, face, chest, etc.); and (b) treatment attributes: paged response (stat, transfer), day/night, weekend/weekday.

Emergency Department (ED) Data: This dataset contained a very diverse set of patient procedures. The attribute types are the same as for the trauma resuscitation data. Unlike tracheal intubation and trauma resuscitation, which are standardized processes, the ED process is not. ED data is quite different from case to case and the activities are temporally sparse.

5.2 Similarity Metric Evaluation

To evaluate our similarity metric, we performed experiments using 65 randomly selected sets of three traces from the Intubation dataset $\{\boldsymbol{T}_i, \boldsymbol{T}_j, \boldsymbol{T}_k\}$ (Fig. 7(a)). Three medical experts were asked to decide the most similar among three trace pairs, $(\boldsymbol{T}_i, \boldsymbol{T}_j)$, $(\boldsymbol{T}_i, \boldsymbol{T}_k)$ and $(\boldsymbol{T}_j, \boldsymbol{T}_k)$ based on their domain knowledge. Our visualizations (Fig. 7(a)) of traces helped them to quickly detect the differences between traces in a set. They used their domain knowledge to judge how important these differences are, and decide which trace pair is more similar than others. We used these labeled results to evaluate our similarity metric. Our baselines included edit distance (ED), sequential-pattern based distance (SP based on algorithm CM-SPADE [22]), normalized Euclidean-distance (NE), and dynamic time warping distance (DTW). We also evaluated these similarity metrics using a majority voting strategy that determines whether the most similar pair selected by each metric matched the majority decision.

The results (Fig. 7(b)) showed that our time-warping-based similarity metric achieved the highest accuracy on both medical expert labels (0.69) and voting-based results (0.80). Edit distance, the simplest metric considered, also performed well because the intubation data was mostly sequential so the activity type and order of performance were the key for comparing the traces. Normalized Euclidean distance and DTW performed worse because they failed in cases where a long intubation trace (e.g., 40 mins) was compared with a brief trace (e.g., 10 mins). The normalized Euclidean distance failed because it could only capture few similarities after normalizing long and short timelines. The DTW failed because it did not penalize long idle times and activity duration differences between traces. In addition to individual metrics, we also computed the accuracy of the majority. The majority of our

similarly metrics correctly identified 39 sets (≥3 votes) and 5 sets as unsure (with two tied majorities).

In 12 of the 65 sets, all metrics and the medical experts agreed on the most similar trace pairs. In another 5 cases, all metrics made wrong choices. We reanalyzed these 5 cases and found that the ground truth was incorrectly labelled in two, and in the other three cases the experts used medical knowledge that was not explicitly considered by the similarity metrics: (1) time-to-task for "decision to intubate," and (2) the type of oxygen mask (BVM vs. NRB). Even without additional domain knowledge, we found that in 62 of 65 cases (95.4%), at least one data-driven similarity metric made the same decision as the experts did. Our TwS-PT (Algorithm 1) independently achieved 69% decision accuracy. These two findings show the feasibility of using purely data-driven similarity metrics for comparing complex process traces.

5.3 Prototype Analysis

We evaluated our prototype extraction method (TwCP, Algorithm 3) quantitatively and by qualitative feedback from domain experts. We used mediod as the benchmark prototype since it is often used as cluster exemplar. For this comparison, we extracted the prototypes and medoids from the whole datasets without clustering, to avoid potential bias from clustering algorithms (Fig. 8). We omitted the ED dataset from this comparison because its prototype and medoid had only two activities. Our results show TwCP prototype had higher average similarity to other traces than the medoid (Fig. 9(a)). This difference was greater for the trauma dataset than for the intubation dataset because the medoid depends on dataset size (number of traces) and trace complexity. A large dataset is more likely to contain a trace close to the centroid. In a small dataset, the medoid may be far from the centroid. Process complexity also affects the medoid because more activities and greater variability makes it less likely that an existing trace will well represent the characteristics of the process. Our intubation data is much simpler than trauma data that had more than 100 activity types and average trace length of 109 activities.

The medoids may not fully capture deviations from the standard protocol due to the variable injuries of different patients. Our TwCP prototype better captured standard practices and included more tasks applicable to a diverse range of injuries, but it may capture idiosyncratic details that would not be expected by a domain expert. For the Trauma data (Fig. 8(b)), the medoid omitted inspection of the eyes, nose and pelvis while the TwCP suggested an acceptable but uncommon sequence for the extremity exam. For the Intubation data (Fig. 8(a)), TwCP included the performance of airway assessment and the use of the non-rebreather (NRB), which the medoid omitted. The medoid more accurately represented oxygen delivery during intubation because one cannot use a bag valve mask (BVM) and NRB simultaneously. TwCP, however, showed that both mechanisms of oxygen delivery were acceptable before intubation and included airway assessment, making the prototype more complete. The human factors literature suggests to study work-as-done rather than work-as-imagined when designing computerized support systems. TwCP prototype is useful since it captures actual work. By comparing a given trace to the prototype, one can detect and analyze the process deviations.

5.4 Recommendation System Evaluation

Our recommendation system was evaluated using two approaches: (E1) whether the actual process trace (denoted as $\boldsymbol{T}_a$) belonged to the most probable cluster decided based on context attributes by the trained regression model; and (E2) whether the recommended trace (denoted as $\boldsymbol{T}_r$) was close to the actual trace. Because trace clusters may be of very different sizes (multi-class imbalance learning problem), we adopted the F-measure (F_1-score) and geometric mean (G-mean) [21] to properly evaluate the performance using the first approach (E1). We did not choose the commonly used accuracy measure as it is ineffective at evaluating imbalanced learning scenarios, where the accuracy of the majority class may dominate. F-measure and G-mean can balance the classification performances of all majority and minority classes. The second approach (E2) evaluated our system by checking if the recommended trace $\boldsymbol{T}_r$ was among the k nearest neighbors of $\boldsymbol{T}_a$, where k ranged from 1 to n and for k=1 the recommended prototype was the closest neighbor. This metric is not symmetrical, i.e., $\boldsymbol{T}_r$ being within k neighbors of $\boldsymbol{T}_a$ does not imply that $\boldsymbol{T}_a$ is within k neighbors of $\boldsymbol{T}_r$. Therefore, a recommended trace that is among a few neighbors of most traces is very representative for the given cluster.

We implemented several similarity metrics: edit distance (ED), sequential pattern (SP), and TwS-TP, and several clustering algorithms: hierarchical clustering (HC), density-peak clustering (DPC) and affinity propagation clustering (APC). We clustered the process traces using different combinations of similarity and clustering algorithms. We used tenfold cross-validation to reduce the variance of the recommendation accuracy. We selected ZeroR as the baseline, which always takes the largest cluster as the prediction result. Our experimental results (Table 2) show that the combination of time-warping distance and APC algorithm achieved the highest F_1 score for both the Intubation and Trauma data. Edit distance with APC algorithm achieved the highest F_1 score for the ED procedure data. From the perspective of the clustering algorithm, APC performed better than HC and DPC in most cases regardless of the similarity metric. From the perspective of the similarity metric, our TwS-PT performed best for both the Intubation and Trauma data. Edit distance performed best for ED data (Table 2), because ED procedures are sparse with only few activities and temporal information is not essential. Temporal information is informative and important for some but not all processes. The selection of similarity metric is best decided by the nature of dataset with the help of visualization tools. Medical procedures depend on other factors that were not recorded in our data, such as the environment, patient condition, and medical team status. This fact explains why we could not achieve very high recommendation accuracy for these complex datasets. An alternative is making recommendations only for a subset of cases when regression model has a high confidence. For example, when we made prediction only for patients whose intubation reason was altered mental status (AMS) and type of call was “now,” we achieved 87.5% recommendation accuracy using the TwS-PT + APC combination.

In 55 of 87 cases (63.2%) in the Trauma dataset, our recommended prototype was among the 5 nearest neighbors of the actual trace (Fig. 9(b)). In the remaining 32 cases, the recommended prototype was not among the 5 nearest neighbors of the actual trace because our regression model incorrectly predicted the cluster membership from trace’s context. For

example, TwS-PT+APC had 0.767 F_1 score for finding cluster membership using context attributes for trauma data (Table 2). A wrong cluster, in turn, results in recommending a wrong prototype.

5.5 A Case Study with Intubation Process

We used the Intubation dataset as a case study to further illustrate the performance of our framework. The recommended number of clusters given by NumC-AP (Algorithm 2) was 2 (Fig. 10(a)(b)). The process traces were clustered using algorithms HC ((Fig. 10(c)), DPC, and APC. In the trained regression model, several context attributes, e.g., intubator role, night shift, intubation reasons, were statistically significant for trace clusters (Table 3). Using the APC result as an example, the reason for intubation and intubator role were significantly correlated with the two clusters. The two prototypes ($\boldsymbol{q}^{c1}$ and $\boldsymbol{q}^{c2}$) (Fig. 11) extracted from two clusters showed many differences: (1) $\boldsymbol{q}^{c1}$ (Fig. 11(a)) had the activities "airway assessment" and "NRB," while in $\boldsymbol{q}^{c2}$ (Fig. 11(b)) these activities were missing; (2) $\boldsymbol{q}^{c1}$ (~19.5 mins) was shorter than $\boldsymbol{q}^{c2}$ (~22 mins); (3) activities "pre-oxy breathing verb." and "decision to intubate" occurred later in $\boldsymbol{q}^{c2}$. In addition to these differences, $\boldsymbol{q}^{c1}$ and $\boldsymbol{q}^{c2}$ had many commonalities, e.g., performance time and sequential order of activities "chest auscultation," "critical window," "RSIs" and "laryngoscopy." Our medical experts explained that in cluster-1 clinicians used a passive non-rebreather (NRB) instead of an active bag-valve mask (BVM) for initial oxygen delivery. The ATLS protocol requires that providers secure the patient's airway before moving onto other survey items. Our results showed that patients in cluster-1 more frequently underwent intubation for respiratory distress. If patients in cluster-1 originally presented with a secured airway, it would make sense that the onset of respiratory distress would necessitate intubation to secure the airway. Patients in cluster-2 were already experiencing some degree of respiratory distress or they would not have needed a BVM. It is plausible, then, that other clinical indicators prompted intubation in cluster-2.

6 CONCLUSION

We presented a process analysis and recommendation framework. Our framework starts by clustering traces of process enactment and extracting a prototypical enactment for each cluster. A regression model was then trained based on the associations between trace clusters and context attributes. A recommended enactment of the process is generated when a new set of context attributes is input into the trained regression model. We introduced novel approaches for measuring trace similarity and extracting prototypes. Although our framework was tested only with medical processes, it can be used for analyzing other real-world processes.

ACKNOWLEDGMENT

This paper is based on research supported by National Institutes of Health under grant number 1R01LM011834-01A1.

REFERENCES

[1]. Van Der Aalst Wil. Process mining: discovery, conformance and enhancement of business processes. Springer Science & Business Media, (2011).

[2]. Yang Sen, Xin Dong, Moliang Zhou, Xinyu Li, Shuhong Chen, Rachel Webman, Aleksandra Sarcevic, Ivan Marsic, and Burd Randall S.. "VIT-PLA: Visual Interactive Tool for Process Log Analysis." KDD Workshop on Interactive Data Exploration and Analytics (2016).

[3]. American College of Surgeons. Committee on Trauma. Advanced trauma life support: ATLS: student course manual. American College of Surgeons, 2012.

[4]. Clarke John R., Hayward Catherine Z., Santora Thomas A., Wagner David K., and Webber Bonnie L.. "Computer-generated trauma management plans: comparison with actual care." World journal of surgery 26, no. 5 (2002): 536–538. [PubMed: 12098040]

[5]. Fitzgerald Mark, Peter Cameron, Cohn Mackenzie, Nathan Farrow, Pamela Scicluna, Robert Gocentas, Adam Bystrzycki et al. "Trauma resuscitation errors and computer-assisted decision support." Archives of Surgery 146, no. 2 (2011): 21S–225.

[6]. Aghabozorgi Saeed, Ah Seyed Shirkhorshidi, and Teh Ying Wah. "Time- series clustering-A decade review." Information Systems 53 (2015): 16–38.

[7]. Levenshtein Vladimir I. "Binary codes capable of correcting deletions, insertions, and reversals." In Soviet physics doklady, vol. 10, no. 8, pp. 707–710. 1966.

[8]. Bose RP Jagadeesh Chandra, and Wil MP van der Aalst. "Process diagnostics using trace alignment: opportunities, issues, and challenges." Information Systems 37, no. 2 (2012): 117–141.

[9]. H Huaulme Arnaud, Voros Sandrine, Riffaud Laurent, Forestier Germain, Alexandre Moreau-Gaudry, and Pierre Jannin. "Distinguishing surgical behavior by sequential pattern discovery." Journal of Biomedical Informatics 67 (2017): 34–41. [PubMed: 28179119]

[10]. Liu Chuanren, Fei Wang, Jianying Hu, and Hui Xiong. "Temporal phenotyping from longitudinal electronic health records: A graph based framework." In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 705–714. ACM, 2015.

[11]. Forestier Germain, Florent Lalys, Laurent Riffaud, Brivael Trelhu, and Pierre Jannin. "Classification of surgical processes using dynamic time warping." Journal of biomedical informatics 45, no. 2 (2012): 255–264. [PubMed: 22120773]

[12]. Forestier Germain, Francois Petitjean, Laurent Riffaud, and Pierre Jannin. "Non-linear temporal scaling of surgical processes." Artificial intelligence in medicine 62, no. 3 (2014): 143–152. [PubMed: 25466153]

[13]. Yang Sen, Moliang Zhou, Rachel Webman, JaeWon Yang, Aleksandra Sarcevic, Ivan Marsic, and Burd Randall S.. "Duration-Aware Alignment of Process Traces" In Industrial Conference on Data Mining, pp. 379–393. Springer International Publishing, 2016.

[14]. Jung Jae-Yoon, Joonsoo Bae, and Ling Liu. "Hierarchical clustering of business process models." International Journal of Innovative Computing, Information and Control 5 12 (2009): 1349–4198.

[15]. Liu Chuanren, Kai Zhang, Hui Xiong, Guofei Jiang, and Qiang Yang. "Temporal skeletonization on sequential data: patterns, categorization, and visualization." IEEE Transactions on Knowledge and Data Engineering 28, no. 1 (2016): 211–223.

[16]. Bose RP Jagadeesh Chandra, and Wil MP van der Aalst. "Context aware trace clustering: Towards improving process mining results." In Proceedings of the 2009 SIAM International Conference on Data Mining, pp. 401–412. Society for Industrial and Applied Mathematics, 2009.

[17]. Frey Brendan J., and Delbert Dueck. "Clustering by passing messages between data points." science 315, no. 5814 (2007): 972–976. [PubMed: 17218491]

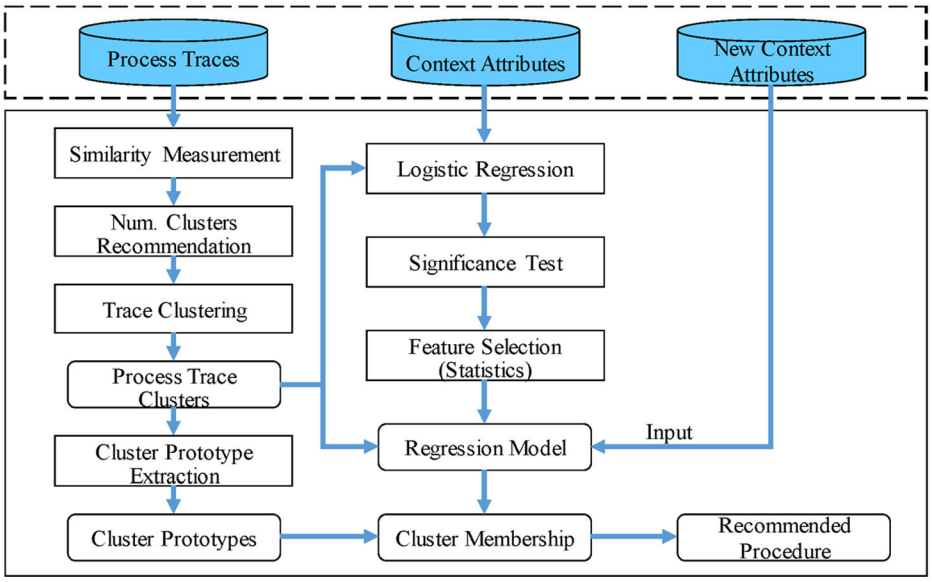
[18]. Rodriguez Alex, and Alessandro Laio. "Clustering by fast search and find of density peaks." Science 344, no. 6191 (2014): 1492–1496. [PubMed: 24970081]

[19]. Murtagh Fionn, and Pierre Legendre. "Ward's hierarchical clustering method: clustering criterion and agglomerative algorithm." arXiv preprint arXiv:l 111.6285 (2011).

[20]. Wasserman Larry. All of statistics: a concise course in statistical inference. Springer Science & Business Media, 2013.

[21]. He Haibo, and Garcia Edwardo A.. “Learning from imbalanced data.” IEEE Transactions on knowledge and data engineering 21, no. 9 (2009): 1263–1284.

[22]. Fournier-Viger Philippe, Antonio Gomariz, Manuel Campos, and Rincy Thomas. “Fast vertical mining of sequential patterns using co-occurrence information” In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 40–52. Springer International Publishing, 2014.



(a) An overview of framework

Case ID	Activity	Start Time	End Time
xx1	Patient Arrival	0:00:00	0:00:01
xx1	NRB	0:00:00	0:00:01
xx1	Pre-Oxy Chest Ausc	0:01:08	0:01:23
xx1	Pre-Oxy Breath Verb	0:01:48	0:01:49
xx1	Airway Assessment	0:05:59	0:06:08
xx1	BVM	0:06:43	0:06:44
xx1	Critical Window	0:07:19	0:07:20
xx1	RSI Sedative Meds	0:07:50	0:08:02
xx1	RSI Paralytic Meds	0:08:16	0:08:32
xx1	BVM	0:09:52	0:09:53
xx1	Laryngoscopy	0:10:19	0:10:51

(b) Medical process trace

Case ID	xxx1	xxx2
Age category	24-96	24-96
Sex	Male	Female
Intubator	PEM Attending	PEM/ED Resident
Direct laryngoscopy	1	1
Night Shift	1	0
Reason	Seizure	Respiratory Distress
Type of Call	ED Patient	Now
Height (cm)	86	90
Weight (kg)	13	16.4
BMI	17.6	20.2
Num. Intubation Attempts	3	3

(c) Process case context attributes

Figure 1.
Data sample and our framework structure.

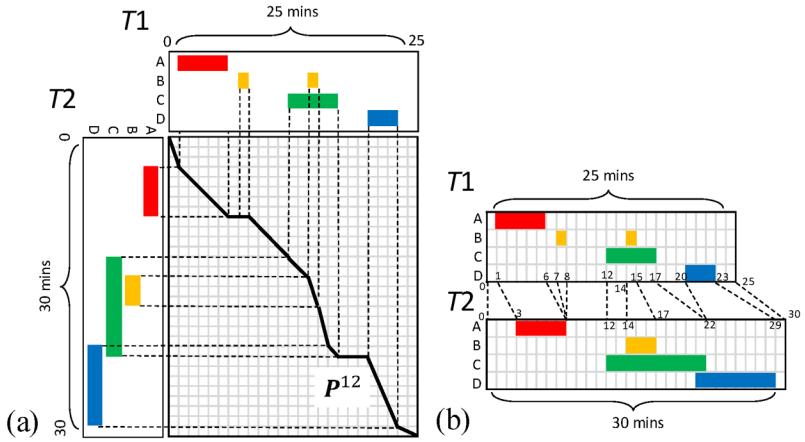


Figure 2.

Our time warping approach to find the minimum warping distance between two process traces $T1$ and $T2$. (a) Illustration of the warping path calculated between $T1$ and $T2$ (Eq. (4)). (b) Alignment of the warped timelines.

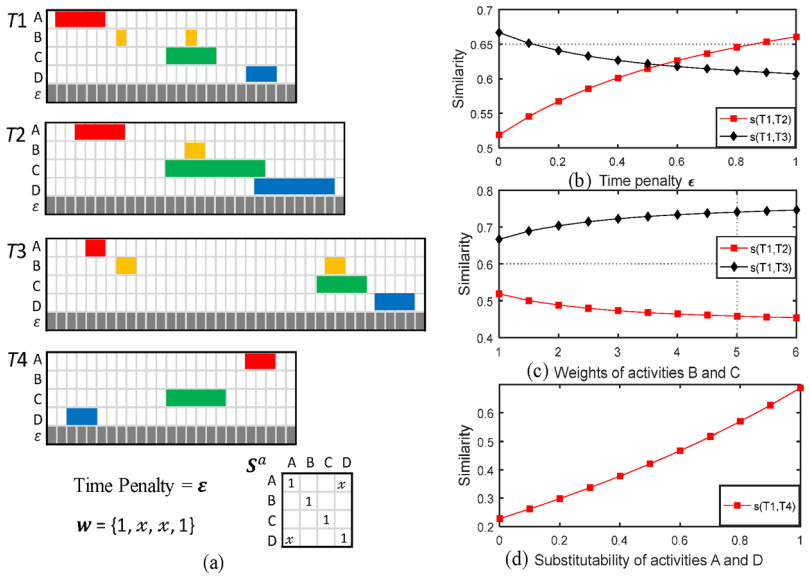


Figure 3.

(a) Example traces $T_1 - T_4$ showing how the similarity results are affected by (b) the time penalty ε, (c) activity weights w, and (d) activity substitutability S^a.

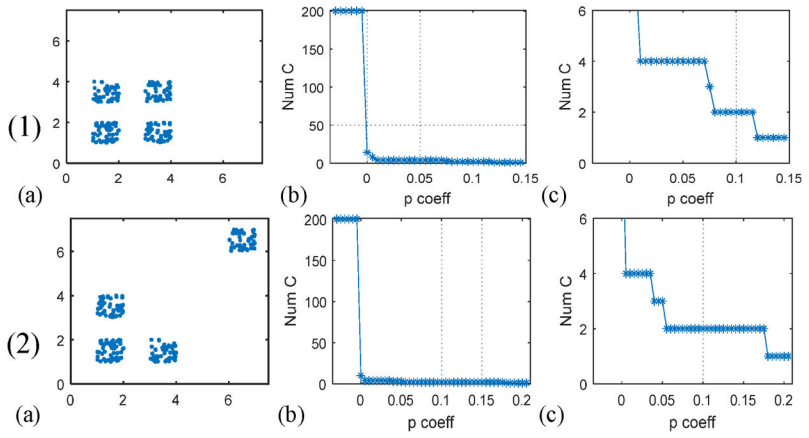


Figure 4.

Two examples of synthetic data in rows (1) and (2) showing how NumC-AP (Algo. 2) decides the number of clusters, (a) The data distribution in a plane, (b) p^c vs. the number of clusters, (c) Zoomed-in view of (b).

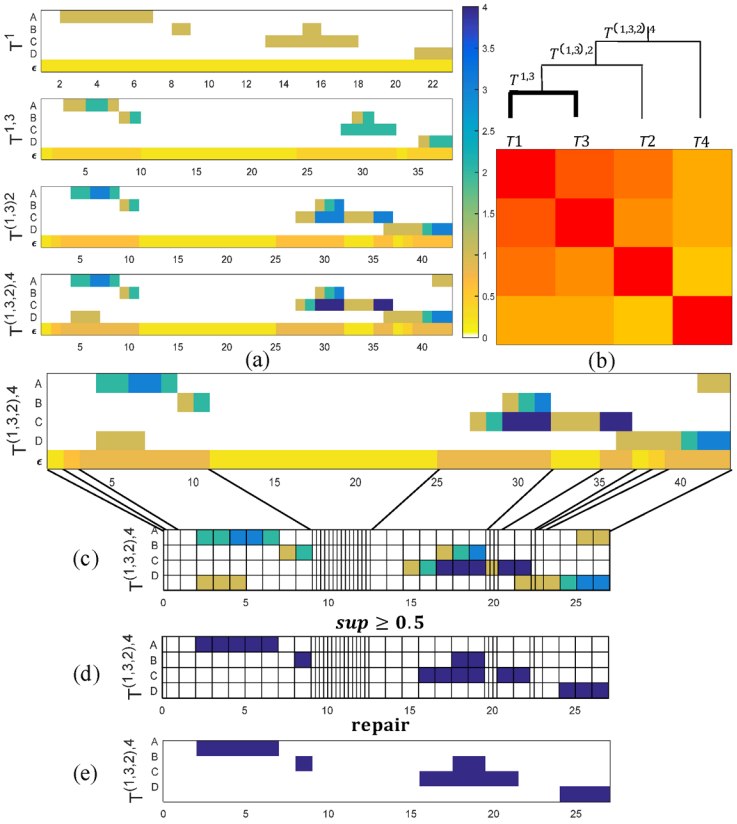


Figure 5.

Steps for calculating a cluster prototype, (a) Calculating prototype *q* pairwise recursively from a set of process traces. Trace activities are shown in rows. After traces are aligned and activities summed up, the summed value is visualized using the color-bar from 1 to *n*, where *n* is the number of traces, (b) A guide tree for directing the prototype calculation for a cluster of traces, (c) Unwarping the warped timeline to restore the timeline and find the prototype, (d) Filtering the prototype using α. (e) Repairing activity C by merging smaller fragment to the larger one.

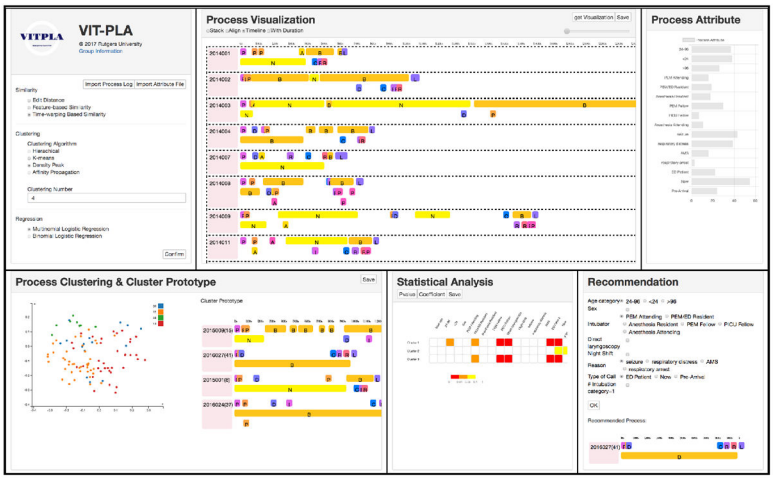


Figure 6.

Graphical user interface of the implementation of our framework for process mining and recommendation.

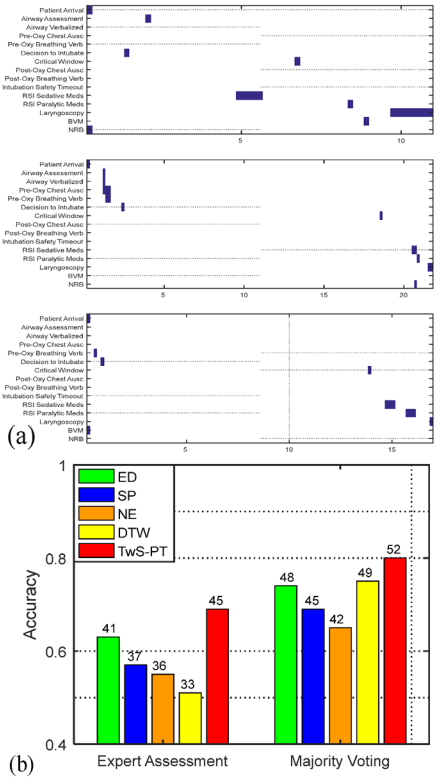


Figure 7.

(a) A sample set of Intubation procedure given to medical experts to evaluate. The horizontal-axis denotes timestamp in minutes and vertical-axis denotes activity types. The blue blocks represent the performance time and duration of activities, (b) Performance of different similarity metrics compared to expert opinion.

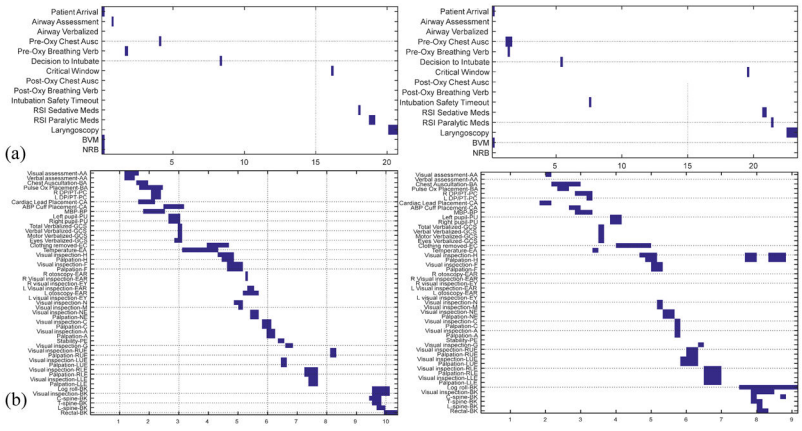


Figure 8.

(a) TwCP prototype (left) and medoid (right) for the whole Intubation dataset. (b) TwCP and medoid for Trauma dataset showing the 52 commonly performed activities. For easier comparison, the vertical axis labels (activity names) were ordered based on a rough temporal order of activities. The horizontal axis denotes the real (not warped) timeline in minutes.

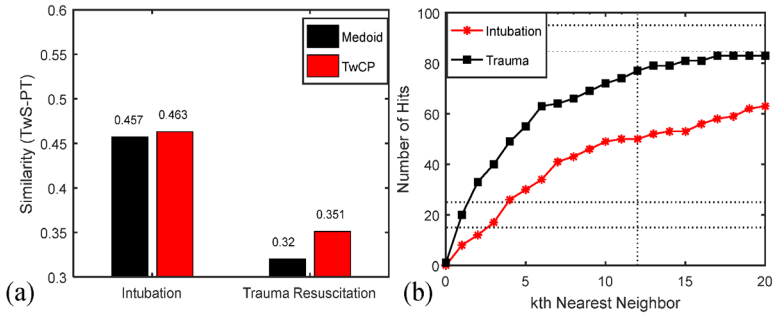


Figure 9.

(a) Avg. similarity between prototypes and other process traces, (b) Number of hits of recommended process enactment within *k* nearest neighbors of the actual enactment.

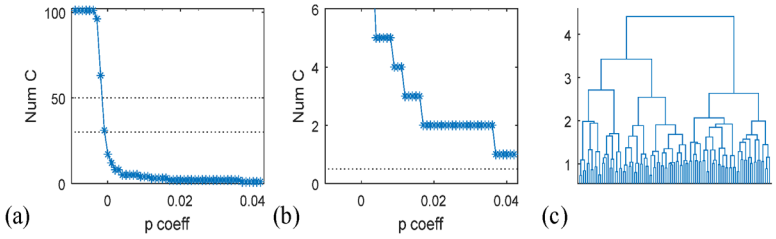


Figure 10.

(a)(b) NumC-AP (Algorithm 2) on Intubation data and (c) hierarchical clustering (based on Ward's method).

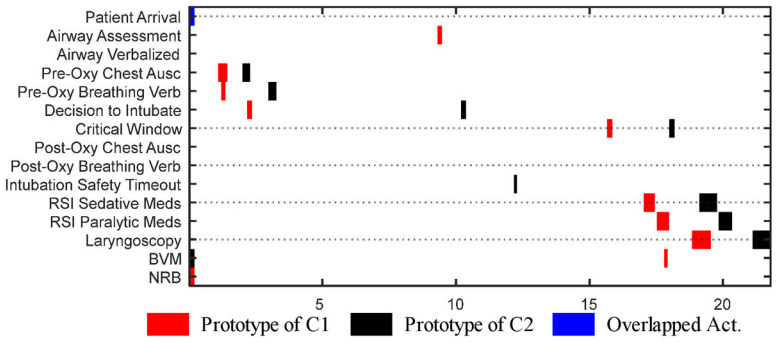


Figure 11.
Prototypes of cluster-1 (q^{c1}) and cluster-2 (q^{c2}).

Table 1.

Properties of our three medical process dataset.

Stats \ Dataset	Intubation	Trauma	ED
Num. Patient Records	101	87	644
Num. Total Acts	1244	9477	2290
Num. Act Types	15	128	65
Longest Trace (Num. Acts)	20	196	12
Shortest Trace (Num. Acts)	S	60	1
Num. External Attributes	10	11	11

Table 2.

Recommendation evaluation on three medical process datasets. The format α (τ) represents the regression model result α and the baseline (ZeroR) result (τ). Rec NC stands for recommended number of clusters.

	Intubation Data		Trauma Resuscitation Data		ED Procedure Data	
Rec NC	**ED (2), SP (3), Time-Warping (2)**		**ED (2), SP (2), Time-warping (2)**		**ED (2), SP (3), Time-warping (2)**	
Metrics	**F-Score**	**G-means**	**F-Score**	**G-means**	**F-Score**	**G-means**
ED + HC	0.505 (0.504)	0.445 (0.479)	0.634 (0.654)	0.448 (0.428)	0.615 (0.615)	0.445 (0.445)
ED + DPC	0.719 (0.755)	0.383 (0.374)	0.692 (0.686)	0.436 (0.413)	0.860 (0.860)	0.293 (0.293)
ED + APC	0.415 (0.339)	0.416 (0.500)	0.346 (0.353)	0.392 (0.500)	**0.595 (0.447)**	**0.571 (0.491)**
SP + HC	0.286 (0.275)	0.412 (0.497)	0.637 (0.533)	0.603 (0.471)	0.395 (0.292)	0.531 (0.499)
SP + DPC	0.446 (0.264)	0.566 (0.496)	0.637 (0.533)	0.603 (0.471)	0.516 (0.516)	0.476 (0.476)
SP + APC	0.487 (0.277)	0.593 (0.471)	0.645 (0.519)	0.591 (0.475)	0.485 (0.477)	0.485 (0.494)
TwS-PT + HC	0.596 (0.419)	0.590 (0.495)	0.526 (0.392)	0.520 (0.497)	0.502 (0.395)	0.554 (0.497)
TwS-PT + DPC	0.605 (0.567)	0.494 (0.461)	0.713 (0.670)	0.556 (0.421)	0.531 (0.387)	0.538 (0.498)
TwS-PT + APC	**0.700 (0.384)**	**0.695 (0.498)**	**0.767 (0.366)**	**0.683 (0.499)**	0.581 (0.471)	0.549 (0.486)

Table 3.

p-values from regression model.

Attributes \ Clustering		HC	DPC	APC
(Intercept)		0.43	0.73	**0.03**
Age	**<24 months**	0.43	**0.07**	0.11
	24–96 months	0.75	0.39	0.94
Gender		0.76	**0.1**	0.34
Intubator Role	**Anesthesia Resident**	0.41	0.31	0.2
	PEM Attending	0.58	0.85	**0.03**
	PEM Fellow	0.17	0.2	0.25
	PEM/ED Resident	0.64	**0.09**	0.79
	PICU Fellow	0.43	0.74	0.36
Direct Laryngoscopy		0.77	0.11	0.4
Night Shift		0.18	**0.03**	0.51
Reason	**Respiratory Distress**	0.15	0.87	**0.02**
	Seizure	0.74	0.94	0.56
Type of Call	**ED Patient**	0.85	0.53	0.14
	Now	0.79	0.34	0.57

Algorithm 1. Time-warping Similarity of Process Traces (TwS-PT)

Input: $\boldsymbol{T}_i$, $\boldsymbol{T}_j$

Output: $s(i,j)$

Step1. Initialize $\boldsymbol{T}_i = [\boldsymbol{p}_1^i, \ldots, \boldsymbol{p}_{k^i}^i]$, $\boldsymbol{T}_j = [\boldsymbol{p}_1^j, \ldots, \boldsymbol{p}_{k^j}^j]$, $\boldsymbol{P}^{ij} = \{\varnothing\}$, $|\boldsymbol{T}_i| = \sum_g |\boldsymbol{p}_g^i|$, $|\boldsymbol{T}_j| = \sum_h |\boldsymbol{p}_h^j|$, $\boldsymbol{t}^{ij} = \{\varnothing\}$.

Step2. Fill score matrix $\boldsymbol{t}^{ij}$ progressively using Eq. (6);

Step3. Deduce $\boldsymbol{P}^{ij}$ by tracing back $\boldsymbol{t}^{ij}$ from $t^{ij}(k^i, k^j)$ to $t^{ij}(0,0)$ and at each step choosing the neighboring cell that yields the maximum score (Eq. (6)).

Step 4: $|\boldsymbol{T}_i \otimes \boldsymbol{T}_j| = (-1) * \boldsymbol{t}^{ij}(k^i, k^j)$;

Step5. **return** $s(i,j)$ computed using Eq. (4)

Algorithm 2. Number of Clusters using AP (NumC-AP)

Input: $\boldsymbol{S} = \{s(i,j)\}, p^c_{min}, \gamma$

Output: N^{c*}

Step1. Initialize $u = 1$, $p^c(u) = p^c_{min}$;

Step2. Run AP clustering with $\boldsymbol{S}$ and p^c(u). The output is the number of clusters $N^c(u)$.

Step3. If $N^c(u) > 1$, $u = u + 1$, $p^c(u) = p^c(u-1) + \gamma$, go to Step2.

Step4. **return** the most stable number of clusters $N^{c*} = mode(N^c)$.

Algorithm 3. Time-warping based Cluster Prototype (TwCP)

Input: $\boldsymbol{C}$, α, β

Output: $\boldsymbol{q}$

Step1. Calculate similarity matrix $\boldsymbol{S}$ of $\boldsymbol{C}$ using Algorithm 1 (TwS-PT);

Step2. Build the guide tree t with HC algorithm and $\boldsymbol{S}$;

Step3. Traverse t bottom up, from leaves to the root;

Step4. $\boldsymbol{T}_i$= node.get(left), $\boldsymbol{T}_j$= node.get(right), align $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$;

$$\boldsymbol{q} = \boldsymbol{T}^{i,j} = \overline{T_i} + \overline{T_j};$$

Step5. Go to Step 3 until current node equals root;

Step6. Unwarp $\boldsymbol{q}$ to recover the timeline;

Step7. Filter $\boldsymbol{q}$with a predefined α and repair $\boldsymbol{q}$ with β;

Step8. **return** $\boldsymbol{q}$