Finding Comparable Patient Histories: A Temporal Categorical Similarity Measure with an Interactive Visualization

Krist Wongsuphasawat

Abstract—Finding similar patients within millions of Electronic Health Records (EHRs) is a challenging problem. A major challenge is how to define a similarity measure that capture the searchers intent. Many methods for computing a similarity measure between time series have been proposed, but patient history with temporal categorical data require fresh thinking. To address this problem, we propose a temporal categorical similarity (TCS) measure, which is based on the concept of aligning temporal data by sentinel events, then matching events between two records. Next, the TCS measure is calculated as a combination of the time differences between pairs of events and number of mismatches. To accommodate customizable parameters in the TCS measure and results interpretation, we implemented Similan, an interactive tool for database search and results visualization. A usability study with 8 participants demonstrates that Similans interface was easy to learn, but users had more difficulty understanding the TCS measure. Users had strong opinions that Similan could help them find similar records in many temporal categorical databases.

Index Terms—Information visualization, Similarity measures, Medical Information Systems, Time series analysis.

1 Introduction

Electronics Health Records (EHRs) are being collected by leading health organizations. These EHRs contains millions of records with patient histories. Challenges arise when clinicians want to find patients with similar symptoms to a target patient in order to guide the treatment of the target patient. A major challenge is defining similarity measures for temporal categorical data.

Many methods for computing a similarity measure between time series have been proposed. However, modifying them to suit temporal categorical data remains an open problem. This paper presents a temporal categorical similarity (TCS) measure data which is based on aligning temporal data by sentinel events [1], then matching events between two records. If the events are identical between records, then the TCS measure is the sum of the distances (time difference) between the matched pairs. A lower distance represents higher similarity.

The problem becomes more complex when the set of events in the target patient does not exactly match those in another patient. To accommodate unmatched events, we convert this into an assignment problem and use the Hungarian Algorithm [2], [3] to match events that produce the minimum distance. Consequently, the TCS measure is redefined as a combination of the number of mismatches and the distance.

• *Krist Wongsuphasawat is a Graduate Student in the Department of Computer Science and the Human-Computer Interaction Laboratory at the University of Maryland, College Park, MD, 20742.*
E-mail: kristw@cs.umd.edu

Furthermore, we believe that an interactive user interface will provide help in finding and understanding results. Since the TCS measure has many customizable parameters which need to be adjusted. We developed an interactive interface, Similan, that allows users to adjust them and see the results in real time. Similan adopts the alignment concept from LifeLines2 [1] and allows users to preprocess the dataset by aligning events by a sentinel event. Similan displays all events in a timeline for each record. Our extension to the rank-by-feature framework [4] allows users to select a target record and then adjust the ranking criteria to explore the impact of result order.

Records are simultaneously visualized on a coordinated scatterplot according to the number of mismatches and the distance function. The comparison panel provides more advanced exploration. When users select one record for a detailed comparison with the target record, they see links between events, enabling them to understand how close the relationship is.

This paper is organized as follows: Section 2, covers the relevant history of similarity searching and related areas. Section 3 discusses the TCS measure. Section 4 discusses the user interface design. Section 5 describes a usability study done to evaluate the interface. We describe future work in section 6, and conclude in section 7.

2 Related Work

A growing body of recent work is focused on finding similar patients. For example, the national health insurance system in Australia records details on medical services and claims provided to its population. Tsoi et al. [5] proposed a method based on clustering

and hidden Markov models to classify patients from medical claims data into various groups. Their aim is to detect similar temporal behavioral patterns among patients in the dataset. *PatientsLikeMe* [6] is an online community where patients with life-altering diseases share and discuss personal health experiences. Users enter their structured data on symptoms, treatments, and health outcomes into the site. This information is rendered as data visualizations on both an individual and an aggregate level. Users can also search for similar patients by specifying demographic information. Unlike PatientsLikeMe, the TCS measure focuses on a sequence of events (symptoms, treatments, and outcomes) in patient records, which is temporal categorical data.

Temporal categorical data is one type of time series. A traditional time series is a sequence of data values, measured at successive times, spaced at (often uniform) time intervals. One common notation is:

$$Y = \{Y_t \mid t \in T\}$$

Many methods for computing a similarity measure between numeric time series have been proposed. According to a survey of previous methods by Ding et al. [7] and Saeed and Mark [8] , similarity measures for time series can be grouped into various types.

The first type is *lock-step* measures, which compare the i-th point of one time series to the i-th point of another. The most straightforward similarity measure of this type is the *Euclidean distance* between two discrete time series X_t and Y_t where the distance between two series is defined as:

$$D(X, Y) = \sqrt{\sum_{t=0}^{M} (X_t - Y_t)^2}$$

However, since the mapping between the points of two time series is fixed, these distances measures are very sensitive to noise and misalignments in time, and are unable to handle local time shifting.

Second, *elastic* measures are distance measures that allow comparison of one-to-many points (e.g., DTW) and one-to-many / one-to-none points (e.g., LCSS). *Dynamic time warping (DTW)* [9], is an algorithm for measuring similarity between two sequences which may vary in time or speed with certain restrictions. The sequences are "stretched" or "compressed" non-linearly in the time dimension to provide a better match with another time series. Continuity is less important in DTW than in other pattern matching algorithms; DTW is an algorithm particularly suited to matching sequences with missing information. However, one important restriction imposed on sequence matching is on the monotonicity of the mapping in the time dimension. This may be violated in EHRs data since swapping of event order can occur.

Another group of *elastic* measures are developed based on the concept of the *edit distance* [10]. In information theory and computer science, the edit distance between two strings of characters is the number of operations required to transform one of them into the other. The lower the number is, the more similar they are. There are several different algorithms to define or calculate this measure. Hamming distance [11], Levenshtein distance [12] or Jaro-Winkler distance [13] are some examples. However, the best known such distance is the LCSS distance, utilizing the *longest common subsequence* model. [14], [15] If a temporal categorical database samples at uniform intervals, we may represent that data in string form ("AABBAC...") and use edit distance as a similarity measure. However, problems arise if more than one event occurs at the same time.

Pattern-based similarity measures, such as SpADe [16], finds out matching segments within the entire time series, called patterns, by allowing shifting and scaling in both the temporal and amplitude dimensions. The problem of computing similarity value between time series is then transformed to the one of finding the most similar set of matching patterns. The disadvantage of SpADe is that it requires tuning a number of parameters.

The transform-based techniques project time series onto a set of functions such as sinusoids or principal components. The data transformation reduces the dimensionality of the original times series and facilitates the use of machine learning techniques [17] or other methods [18] in matching time series. The TQuEST [19] threshold-based approach introduced an idea to transform time-series into a sequence called *threshold-crossing* time intervals. Each time interval is then treated as a point in two dimensional spaces, where the starting time and ending time constitute the two dimensions. The similarity measure is then computed from the two sequences of time interval points.

However, X_t and Y_t for all methods above are numerical values. But for temporal categorical data, they are not numerical values.

$$Y_t \in \{\text{“}Category1\text{”}, \text{“}Category2\text{”}, \ldots\}$$

The TCS measure can address all the issues. Nevertheless, the TCS approach needs to match events between two records. New challenges arise since there are many possible ways to match events but the TCS measure requires matching which will yield maximum similarity. This matching problem can be reduced to problem in graph theory, called the *assignment problem*. [2] In its most general form, the problem is as follows:

"There are a number of agents and a number of tasks. Any agent can be assigned to perform any task, incurring some cost that may vary depending on the agent-task assignment. It is required to perform all tasks by assigning exactly one agent to each task in such a way that the total cost of the assignment is minimized."

If the numbers of agents and tasks are equal and the total assignment cost for all tasks is equal to the sum of the costs for each agent (or the sum of the costs for each task, which is the same thing in this case), then the problem is called the Linear assignment problem. When

the assignment problem has no additional qualifications, the term linear assignment is used.

The *Hungarian algorithm* is a combinatorial optimization algorithm which solves assignment problems in polynomial time ($O(n^3)$). The first version, known as the Hungarian method, was invented and published by Kuhn [2] in 1955. This was revised by Munkres [3] in 1957, and has been known since as the Hungarian algorithm, the Munkres assignment algorithm, or the Kuhn-Munkres algorithm. Bertsekas [20] has proposed a new and more efficient algorithm for the assignment problem. The factor of improvement increases with the problem dimension N and reaches an order of magnitude for N equal to several hundreds.

Before applying the similarity measure to the data, Similan allows users to preprocess data by aligning them by sentinel events. Previous work has shown that using an absolute time scale alone does not address all of the tasks users face when comparing temporal categorical data. In particular, tasks that involve temporal comparisons relative to important events such as a heart attack are not supported. Wang et al. [1] propose a concept of aligning temporal data by sentinel events. Sentinel events are important events, e.g. heart attack, which are used as references. The time in each record is then recomputed, referenced from the time that sentinel event in each record occurs. Making time at which sentinel event, events before sentinel event and events after sentinel events occur become zero, negative and positive, respectively.

Seo and Shneiderman [4] presented a conceptual framework for interactive feature detection named *rank-by-feature framework*. In the rank-by-feature framework, users can select an interesting ranking criterion, and then all possible axis-parallel projections of a multidimensional data set are ranked by the selected ranking criterion. The ranking result is visually presented in a color-coded grid called *Score Overview*. Similan, inspired by this *rank-by-feature* idea, allows users to rank the dataset by many criteria derived from the TCS measure and display those scores in a color-coded grid.

In order to facilitate the result interpretation, the data records should be visualized in a meaningful way. Ma et al. proposed Event Miner, a tool that integrates data mining and visualization for analysis of temporal categorical data. [21] However, Event Miner was designed for analyzing only one record of temporal categorical data. There has been a number of published visualization works on temporal categorical data on timelines. A design using timelines for medical records was proposed by Powsner and Tufte [22], who developed a graphical summary using a table of individual plots of test results and treatment data. LifeLines [23] presented personal history record data organized in expandable facets and allowed both point event and internal event representations. Alonso et al. [24] conducted an experiment to compare a tabular format and the graphical presentation in LifeLines. Results suggest that overall the LifeLines representation led to much faster response times. A test showed that LifeLines can reduce some of the biases of the tabular record summary. However, their design is not suitable for displaying many records at the same time and does not assist comparison between records.

3 Temporal Categorical Similarity (TCS) Measure

We define a new similarity measure for temporal categorical (TC) data. The following notation is used to describe a temporal categorical record, which is a list of temporal categorical events.

$$X = \{x_t^c\}$$

$$c \in \{\text{“}Category1\text{”}, \text{“}Category2\text{”}, \ldots\} \text{ and } t \in Time$$

The TCS measure consists of two measures. One measure is for the matched events, events which occur both in target record and compared record. Another measure is for the missing or extra events, events which occur in a target record but do not occur in compared record, or vice versa. The first measure is defined in terms of distance function between two temporal categorical records. A lower distance means higher similarity. The distance function is later converted to a *match score*, ranging from 0.01 to 1.00. The second measure is a *mismatch score*. It is based on difference in number of events in each category between the two records. Both measures are combined into *total score*, ranging from 0.01 to 1.00. For all three scores, a higher score represents higher similarity. Only a perfect match, having no missing or extra events and zero distance, will yield a total score of 1.00.

3.1 TCS Distance Function

We first define a distance function between each pair of events, as follow:

$$d(x_t^c, y_u^d) = \begin{cases} |t-u| & \text{if } c = d \\ \infty & \text{if } c \neq d \end{cases} \quad (1)$$

The distance is computed from time difference if both events have the same category. Since we do not allow matching between different event categories, the distance when both events come from a different category is infinity.

Then, a distance function between a target record

$$X = \{x_{t_1}^{c_1}, x_{t_2}^{c_2}, \ldots, x_{t_m}^{c_m}\}$$

and a compared record

$$Y = \{y_{u_1}^{d_1}, y_{u_2}^{d_2}, \ldots, y_{t_n}^{d_n}\}$$

is described as follow:

$$D(X,Y) = min \sum_{i \in [1,m], j \in [1,n]} d(x_{t_i}^{c_i}, y_{u_j}^{d_j}) \quad (2)$$

each i and j is used exactly once.

A distance function between two records is calculated by matching events from the two records into event

pairs and summing the distance $d(x_t^c, y_u^d)$ between each pair. There can be many possible ways to match events into pairs. Therefore, the distance function will be calculated from the matching which produces the minimum distance. (How to match events to produce minimum distance will be explained in section 3.2.) However, this distance function works only when $m = n$ because it requires a perfect match between the two records. Also, even when $m = n$, this case can occur:

$$X = \{x_{t1}^{category1}, x_{t2}^{category1}, x_{t3}^{category2}\}$$

$$Y = \{y_{u1}^{category1}, y_{u2}^{category2}, y_{u3}^{category2}\}$$

This will force at least one pair of different category events to pair together, which is not preferred. Therefore, the distance function fills in some null events (x_{null}) to equalize numbers of events between the two records in each category. The two lists above become.

$$X = \{x_{t1}^{category1}, x_{t2}^{category1}, x_{t3}^{category2}, x_{null}^{category2}\}$$

$$Y = \{y_{u1}^{category1}, y_{null}^{category1}, y_{u2}^{category2}, y_{u3}^{category2}\}$$

The distance function between each pair of events is revised. Let X be a target record and Y be a compared record.

$$d'(x_t^c, y_u^d) = \begin{cases} dist(t,u) & \text{if } c = d \\ \infty & \text{if } c \neq d \end{cases} \quad (3)$$

$$dist(t,u) = \begin{cases} |t-u| & \text{if } t, u \neq null \\ missingPenalty & \text{if } t \neq null \text{ and } u = null \\ extraPenalty & \text{if } t = null \text{ and } u \neq null \\ \infty & \text{if } t, u = null \end{cases} \quad (4)$$

Currently, the missing event penalty ($missingPenalty$) and extra event penalty ($extraPenalty$) are set to zero by default. This is because the distance function is a measure for the matched events. It should not be affected by missing or extra events. Those events will be handled by another measure which will be described in section 3.4

Finally, a distance function between a target record X and a compared record Y becomes:

$$D'(X,Y) = min \sum_{i \in [1,m], j \in [1,n]} d'(x_{t_i}^{c_i}, y_{u_j}^{d_j}) \quad (5)$$

each i and j is used exactly once.

3.2 Minimum Distance Perfect Matching

The problem here is how to match each $x_{t_i}^{c_i}$ and $y_{u_j}^{d_j}$ in X and Y to yield minimum distance. Then the problem becomes an assignment problem (see section 2).

Let events $(x_{t_i}^{c_i})$ from X become agents and events $(y_{u_j}^{d_j})$ from Y become tasks. Cost of the assignment is $d'(x_t^c, y_u^d)$. Then the Hungarian Algorithm solves the problem. The distance matrix between X and Y is displayed below.

$$\begin{array}{c} \\ x_{t_1}^{c_1} \\ x_{t_2}^{c_2} \\ \vdots \\ x_{t_m}^{c_m} \end{array} \begin{array}{c} \begin{array}{cccc} y_{u_1}^{d_1} & y_{u_2}^{d_2} & \cdots & y_{u_n}^{d_n} \end{array} \\ \begin{pmatrix} d'(x_{t_1}^{c_1}, y_{u_1}^{d_1}) & d'(x_{t_1}^{c_1}, y_{u_2}^{d_2}) & \cdots & d'(x_{t_1}^{c_1}, y_{u_n}^{d_n}) \\ d'(x_{t_2}^{c_2}, y_{u_1}^{d_1}) & d'(x_{t_2}^{c_2}, y_{u_2}^{d_2}) & \cdots & d'(x_{t_1}^{c_1}, y_{u_n}^{d_n}) \\ \vdots & \vdots & \ddots & \vdots \\ d'(x_{t_m}^{c_m}, y_{u_1}^{d_1}) & d'(x_{t_m}^{c_m}, y_{u_2}^{d_2}) & \cdots & d'(x_{t_m}^{c_m}, y_{u_n}^{d_n}) \end{pmatrix} \end{array}$$

3.3 Match Score

The TCS distance function can be used to represent the similarity between the target record and a compared record. However, the distance can be a large number, which users find difficult to compare. Hence, we normalize the distance into a *match score*, ranging from 0.01 to 1.00. A higher score represents higher similarity. Only records with zero distance will yield a score of 1.00. Let n be total number of records in the dataset. X and Y are target and compared record, respectively. The match score ($M(X,Y)$) is calculated from the following equation:

$$D'_{max} = Max_{j \in [1,n]} D'(X, Y_j) \quad (6)$$

$$M(X,Y_i) = \begin{cases} 1.00 & \text{if } D'(X,Y_i) = 0 \\ \frac{D'_{max} - D'(X,Y_i)}{D'_{max}} * .98 + .01 & \text{otherwise} \end{cases} \quad (7)$$

3.4 Mismatch Score

When the number of events in two records are not equal, there are missing or extra events. A *missing* event is an event that occurs in a target record but does not occur in a compared record. An *extra* event is an event that does not occur in a target record but occurs in a compared record. For example, imagine a target record for a patient who has chest pains, followed by elevated pulse rate, followed by a heart attack diagnosis. If the compared record has only chest pains and heart attack diagnosis, it has one missing event.

We count a *number of mismatches* ($N(X,Y)$), a sum of number of missing or extra events in each category, and normalize it into a *mismatch score* ($MM(X,Y)$), ranging from 0.01 to 1.00. Only records with no mismatch events will yield a score of 1.00.

$$N_{max} = Max_{j \in [1,n]} N(X, Y_j) \quad (8)$$

$$MM(X,Y_i) = \begin{cases} 1.00 & \text{if } N(X,Y_i) = 0 \\ \frac{N_{max} - N(X,Y_i)}{N_{max}} * .98 + .01 & \text{otherwise} \end{cases} \quad (9)$$

3.5 Total Score

The *match score* and *mismatch score* are combined into *total score* ($T(X,Y_i)$) using weighted sum.

$$T(X,Y_i) = w * M(X,Y_i) + (1-w) * MM(X,Y_i); w \in [0,1] \quad (10)$$

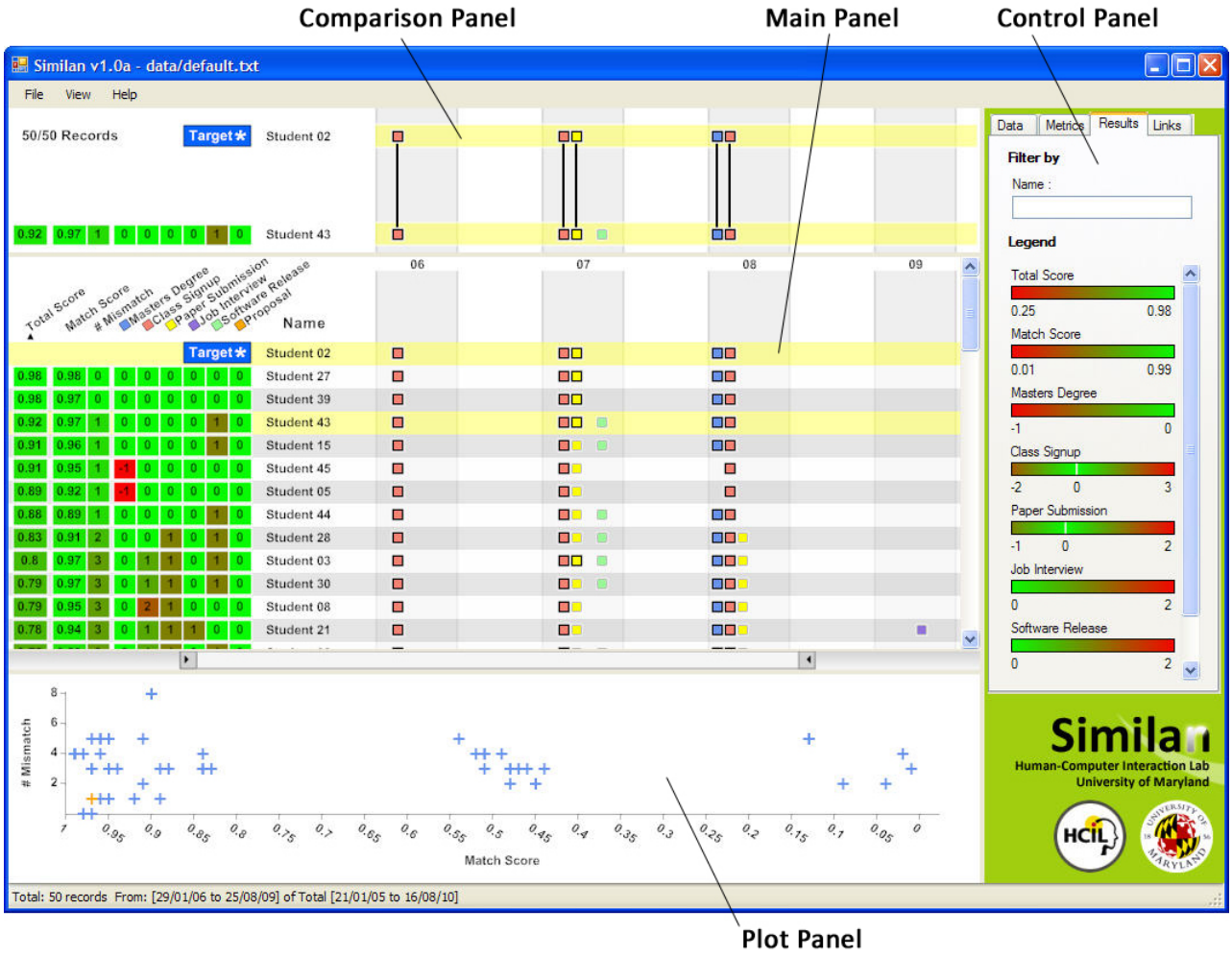

Fig. 1. Overview of Similan - Similan consists of 4 panels: main, comparison, plot and control. Users can start from selecting a target record from the main panel. After that, the main and plot panels give an overview of the similarity search result. Users can rank all records using many ranking criteria provided. By clicking on a particular record, the comparison panel will show relationships between that record and the target record.

The default value for weight (w) is 0.5. However, the Similan interface allows users to adjust this weight by themselves and see the results in real-time. (will be discussed in section 4.5)

4 Similan Interface Design

The final output of the TCS measure is a score which represents the similarity between a pair of records. However, the score alone does not help the users understand why records are similar or dissimilar. Also, the TCS measure can be adjusted by many parameters. Furthermore, one of the users' goals is to find the similar records from a database which contains multiple records. Hence, a tool to assist the users to understand the results, customize the parameters, and perform a similarity search in a database is needed. To address these issues, an interactive interface called Similan is developed. Similan is written in C# .NET using the Piccolo.NET [25] visualization toolkit. Similan provides a visualization of the search results to help the users understand the results, and an interface that facilitates the searching process and parameter customization. The key design concept of Similan follows the Information Visualization Mantra [26] : overview first, zoom and filter, details on demand.

4.1 Overview

Similan consists of 4 panels: main, comparison, plot and control, as shown in Figure 1. Users can start from selecting a target record from the main panel. After that, the main and plot panels give an overview of the similarity search result. Filtering and ranking mechanisms help users narrow down the search result. Users then can focus on fewer records. By clicking on a particular record, the comparison panel shows relationships between that record and the target record on demand. Moreover, mouse hovering actions on various objects also provide details on demand in the form of tooltips.

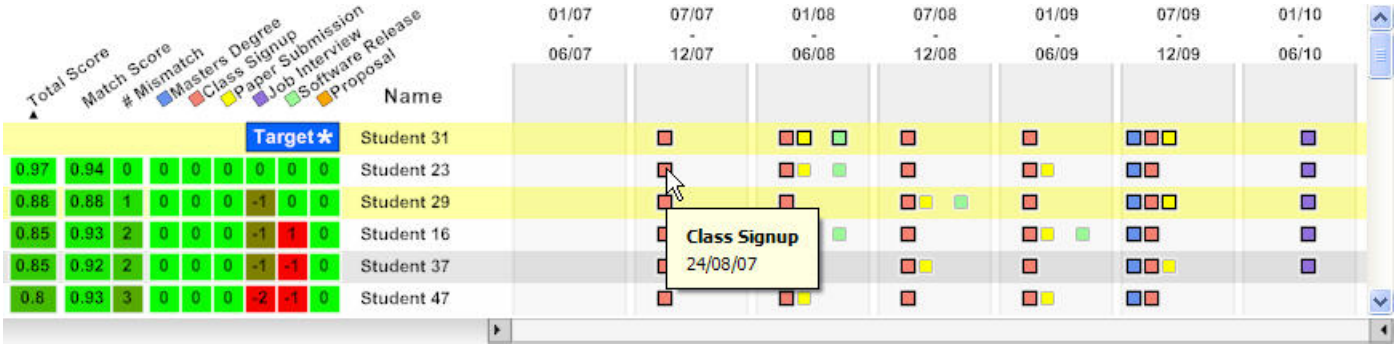

Fig. 3. Main Panel: Events are sorted by category and placed on range timeline, the time axis is binned into discrete ranges. Ranking score are shown on the left. Records can be sorted by clicking the headers on the top. Zooming and panning are possible using a range slider at the bottom.

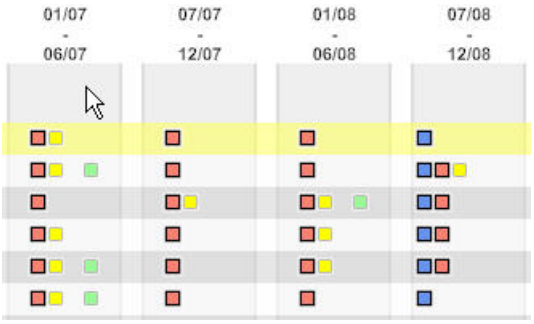

Fig. 2. Range Timeline: Events are grouped by time range. Range labels are placed on the top

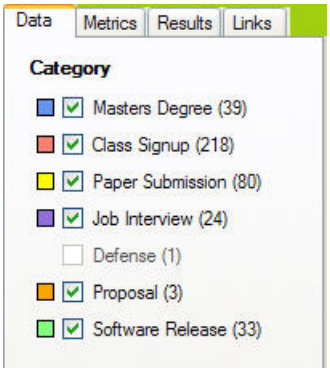

Fig. 4. Control Panel: Select up to six interested categories. Numbers of events in each category are display in the label. Click on the colored squares to customize color.

All panels are resizable which provides flexibility in utilizing space. Results of any parameter adjustment are displayed immediately.

4.2 Events and Timeline

Colored squares are used to represent events. Each color represents a category. Currently, Similan allows a maximum of 6 categories. Users can check the checkboxes in the control panel (Figure 4) to select up to six categories. The number of events in each category are displayed behind the category name to help users make their decisions. Users can also customize the colors using a control provided in the control panel.

Similan's timeline is not a continuous timeline (see Figure 2). This timeline is divided into time ranges. The range interval is automatically calculated by the size of application window and interval of the database. In each time range, events are grouped by category and categories are placed in the same order. Maintaining the same order allows visual comparison between two records (see main panel and comparison panel).

4.3 Main Panel

Each record is vertically stacked on alternating background colors and identified by its name on the left (see Figure 1). Ranking scores appear on the left hand side before the name (more detail about ranking score will be explained in section 4.5). Events appear as colored squares on the timeline as described in section 4.2. By default all records are presented using the same absolute time scale (with the corresponding years or month labels displayed at the top) and the display is sized so that the entire date range fits in the screen.

A **double-click** on any record sets that record to be the target record. A target mark will be placed in front of the target record instead of a ranking score. **Click** on any record to select a compared record. Both the target record and compared record will be highlighted. Users can move the cursor over colored squares to see details on demand in the form of tooltips. Also, zooming on the horizontal axis and panning are possible using a range slider provided at the bottom of the main panel.

4.4 Alignment

According to Wang et al. [1], *alignment* allows users to perform tasks that involve temporal comparisons relative to sentinel events. Sentinel events are important events, e.g. heart attack, which are used as references. The time in each record is recomputed, referenced from the time that the sentinel event in each record occurs. Similan adapts this idea by allowing users to align temporal categorical events by *sentinel category*. Since there

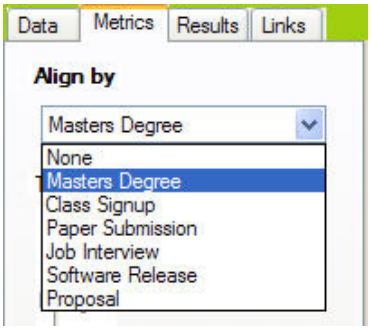

Fig. 5. Control Panel: Users can choose to align events by selecting sentinel category.

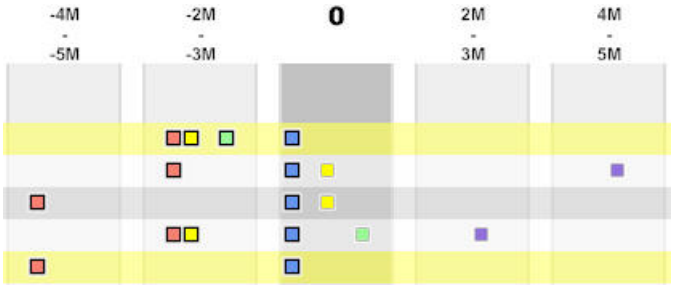

Fig. 6. Relative Timeline: Time scale is now relative to sentinel events (blue). Time zero is highlighted.

can be more than one candidate for the sentinel event in the sentinel category, a candidate event which produces the maximum score according to the TCS measure will be selected.

Users can select a sentinel category from a drop-down list as shown in Figure 5. By default, sentinel category is set to none. When the sentinel category is selected, the time scale will change from an absolute time, i.e. real time, into a relative time. The sentinel event becomes time zero. In the timeline, time zero will be highlighted, as shown in Figure 6.

4.5 Rank-by-feature

Similan is inspired by the idea of *rank-by-feature* from Hierarchical Clustering Explorer (HCE) [4]. Ranking criteria are derived from the TCS measure proposed earlier in section 3 of this paper. Whenever a target record has been selected, the similarity measure will be calculated for each record. The main panel then allows users to sort records according to these ranking criteria:

1) **Total Score**
   ranging from 0.01 to 1.00
   as described in section 3.5. Total score is a weighted sum of match and mismatch score. The weight can be adjusted and see the result in real-time using a slider and textboxes as shown in figure 7
2) **Match Score**
   ranging from 0.01 to 1.00
   as described in in section 3.3. We choose to display match score instead of distance because the distance is a large number, which can go up to 5-6 digits even for a small dataset. Since it is hard to tell the difference between two large numbers and to understand the distribution, Similan shows the match score.
3) **Number of Mismatches (#Mismatch)**
   ranging from 0 to n
   as described in section 3.4. This is the total number of missing and extra events. The number of mismatches is shown, instead of the mismatch score because number of mismatches is a small number and one unit of this number is more meaningful than one unit of distance. Furthermore, we break down the number of mismatches into categories. Positive value means number of extra events while negative value means number of missing events.

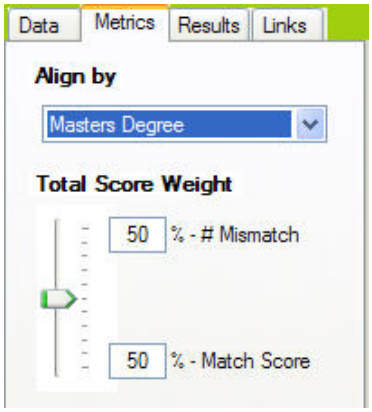

Fig. 7. Control Panel: Weight for calculating total score can be adjusted using slider and textboxes.

Users can click on the ranking criteria on the top to sort the records. By clicking again the order is then reversed. A triangle under the header shows current ranking criterion. Legends in the control panel show the range of each ranking score and how they are color-coded, see Figure 1.

4.6 Plot Panel

In addition to displaying results as a list in the main panel, Similan also visualizes the results as a scatterplot in the plot panel (Figure 8). Records are represented by + sign on a scatterplot. Horizontal axis is the match score while vertical axis is the number of mismatches (#mismatch). Records in the bottom-left area are records with high match score and low number of mismatches, which should be considered most similar according to the TCS measure.

Moving the cursor over a + sign will trigger a tooltip to be displayed. Clicking on a + will set that record to be the compared record and scroll the main panel to that record. Users can also draw a region on the scatterplot to filter records. The main panel will show only records in the region. Clicking on the plot panel again will clear the region and hence clear the filter.

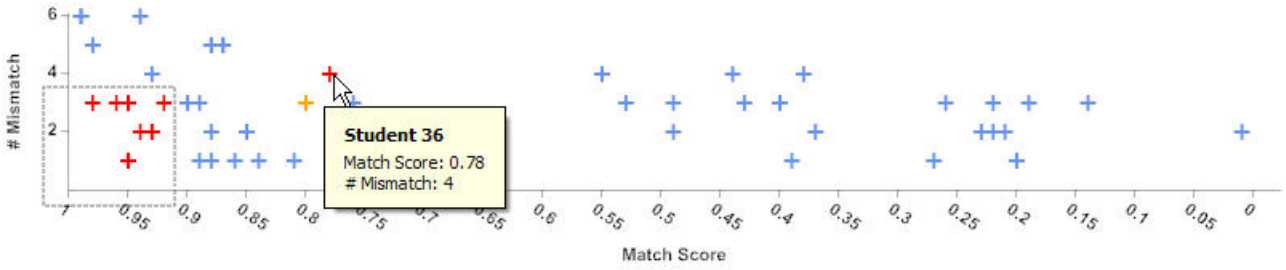

Fig. 8. Plot Panel: Records are placed on 2-dimensional space by their match score and number of mismatches.

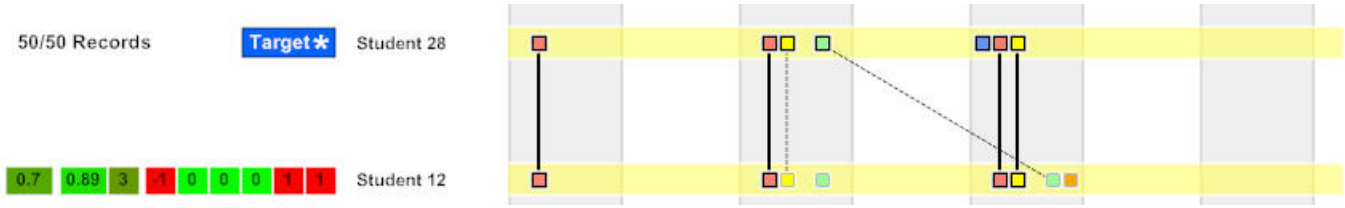

Fig. 9. Comparison Panel: Solid lines are links with low distance. Dashed lines are links with high distance. Move cursor over a link to see more detail.

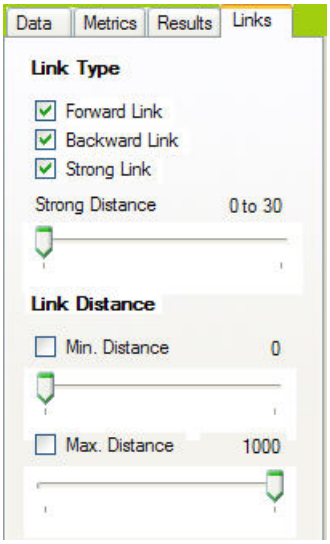

Fig. 10. Control Panel: Links in comparison panel can be filtered using this control.

4.7 Comparison Panel

The comparison panel is designed to show similarity and difference between the target record and the compared record. To maintain consistency, each record is displayed exactly the same as in main panel. Lines are drawn between pairs of events matched by the TCS measure to show similarity and difference. Line style is used to show the distance value. *Strong links*, or links with low distance, are shown as solid lines. *Weak links*, or links with high distance, are shown as dashed lines. Those events without any links connected to them are missing or extra events. Users can adjust the distance threshold for strong links in the control panel, see Figure 10. Moving the cursor over a link will display a tooltip showing the event category, time of both events and distance.

Furthermore, users can filter the links by using the filters in the control panel (Figure 10). Users can filter by setting the minimum and/or maximum distance. By selecting link types, only the selected type are displayed. *Strong links* are links with a distance in the range specified by the slider. *Forward Links* are links which are not strong links and the event in target record occurs before the event in compared record. *Backward Links* are the opposite, links which are not strong links but the event in the compared record occurs before the event in the target record.

5 Evaluation

A usability study for Similan was conducted with 8 participants. The goals in this study were to study the learnability of Similan, assess the benefits of a scatterplot, and learn how the number of events and categories affect user performance. We also observed what strategies the users chose and what problems they encountered while using the tool. Synthetic data based on graduate school academic events, such as admission, successful dissertation proposal, and graduation, are used instead of Electronics Health Records. This change was intended to make the tasks more comprehensible and meaningful to participants, who were technically-oriented graduate students.

5.1 Usability Study Procedure

Two versions of Similan were used in this usability study. One with full features (S-Full) and another without a scatterplot (S-NoPlot). All usability sessions were conducted on an Apple laptop (15 inch widescreen, 2.2 Ghz

CPU, 2GB RAM, Windows XP Professional) using an optical mouse.

5.2 Tasks

The study had two parts. In the first part, participants had an introduction to the TCS measure and training with the Similan interface without a scatterplot (S-NoPlot) for each participant. Then, the participants were asked to perform these tasks:

1) Given a target student and dataset of 50 students. Each student record has 2 categories of events and the total number of events is between 4 to 6 events. Find 5 students which are most similar to the target student using S-NoPlot.
2) Given a target student and dataset of 50 students. Each student record has 4 categories of events and the total number of events is between 6 to 10 events. Find 5 students which are most similar to the target student using S-NoPlot.
3) Given a target student and dataset of 50 students. Each student record has 6 categories of events and the total number of events is between 8 to 16 events. Find 5 students which are most similar to the target student using S-NoPlot.

After that, participants were introduced to the scatterplot and how to use it. Then, they were asked to perform these tasks.

4) Given a target student and dataset of 50 students. Each student record has 2 categories of events and the total number of events is between 4 to 6 events. Find 5 students which are most similar to the target student using S-Full.
5) Given a target student and dataset of 50 students. Each student record has 4 categories of events and the total number of events is between 6 to 10 events. Find 5 students which are most similar to the target student using S-Full.
6) Given a target student and dataset of 50 students. Each student record has 6 categories of events and the total number of events is between 8 to 16 events. Find 5 students which are most similar to the target student using S-Full.

The datasets used in task 1-3 and 4-6 are the same but the students are renamed and the initial orderings are different. Task 1 and 4 are used only for training purpose. The results will be collected from task 2, 3, 5 and 6.

In addition to observing the participants behavior and comments during the sessions, we provided them with a short questionnaire which asked specific questions about the Similan interface. Answers were recorded using a seven-option Likert scale and free response sections for criticisms or comments.

5.3 Results

For the first part of this 30-minute study, all participants were observed to use the following strategy: first select the target student, and then use the main panels ranking mechanisms to rank students by the total score. In their first usage, some participants also selected the student who had the highest total score to see more detail in the comparison panel. Afterwards, they just studied the visualization and reported that these students with high total score are the answers.

For the second part of the study, which focused on the scatterplot, most of the participants were observed to use the following strategy: first select the target student, draw a selection in the plot panel, and then use main panels ranking mechanisms to rank students by the total score. However, a few participants did not use the scatterplot to do the task at all. They used the same strategy as in the first part.

Users spent comparable time on tasks 2 and 3 and on tasks 5 and 6. There was no difference in performance times between tasks 2 and 3 or between tasks 5 and 6, even though there were more events in tasks 3 and 6. This is understandable since participants reported that they trusted the ranking criteria provided by the interface. However, users spent more time doing the tasks while using the scatterplot.

All of the participants trusted the *total score* ranking criterion and used it as the main source for their decisions. ”I am trusting the algorithm”, said one participant. ”The total score ranking function is very useful”, said another participant. They explained that the visualization in the main panel convinced them that the ranking criterion gave them the correct answers. Therefore, in the later tasks, after ranking by total score and having a quick look at the visualization, they simply answered that the top five are the most similar.

All of them agreed that the main panel is useful for its ranking features. But when asking about the distribution of the similarity search result set, the scatterplot became more valuable. However, they had different opinions about its usefulness in finding similar students. Some of the participants mentioned that it was useful when they wanted to find similar students. They explained that the similar students can easily be found at the bottom left of the scatterplot. One participant said that, by drawing a selection on the scatterplot, she had to choose two parameters (*number of mismatches* and *match score*) while using the main panel, she only had to choose one parameter (*total score*). Few of them even mentioned that it is not necessary to use the scatterplot if they just want to find some similar students. ”If I want to find similar, just sorting is enough”, said one participant.

Although they had different opinions about its usefulness in finding similar students, they all agreed that the scatterplot gives a good overview of the students’ distribution. It can show clusters of students which could not be discovered from other panels. Also, one participant pointed out that the result and comparison panels are helpful in showing how students are similar, while the plot is more helpful in explaining how students are dissimilar. The plot characterizes students with respect to

the target student. However, one participant complained that the student markers can overlap on the plot which can be misleading in some situations.

Every participant mentioned that the comparison panel is useful in showing the similarity between a target student and a compared student.

Participants had positive comments on Similan simple, intuitive and easy to learn interface. Most of the participants got started without assistance from the experimenter. Some participants clicked instead of double-clicked until they were given help. One participant said that she like the way the main panel, comparison panel and plot panel are connected (coordinated).

Nevertheless, some user interface design concerns were noted. One participant was not sure that the ranking header could be clicked on. Another participant noticed that the range timeline could be misleading in some situations. One participant suggested adding some visual effects to bring attention to the comparison panel when users select a student from the main panel. He also suggested drawing a guide box in the bottom left corner of the scatterplot to show that this is a high similarity area.

Overall, participants liked the simple but attractive Similan's interface and strongly believed that Similan can help them find students who are similar to the target student. Ranking in the main panel appears to be useful. By contrast participants had difficulties in learning the TCS measure, since it combines two kinds of scores. The scatterplot did not benefit the tasks in this study but we believe it may yet prove useful for more complex databases.

6 Future Work

6.1 Similarity Measure

The TCS measure does not allow matching of events between different categories. However, allowing matching between different categories can make the similarity measure become more flexible. This can be easily achieved by adjusting the different type penalty (currently set to ∞) in equation 1. User interface features can be added to help users specify penalty for matching between different categories.

Currently, the TCS measure takes all missing and extra events into account. In some situations, missing events are not considered important but extra events are, or vice versa. In a more complex situation, missing 1 of 2 events is not important but missing 2 of 2 events can be considered critical. These issues have to be addressed.

Also, EHRs contains millions of records with patient histories. Dealing with a large database is another challenging problem because of the polynomial nature of the Hungarian algorithm. An improved algorithm to calculate the similarity measure must be fast enough to support large databases efficiently.

6.2 User Interface Design

Some user interface design issues in Similan need to be addressed. Using range timeline can be misleading is some situations. A pair of events in the same range can have a longer distance than a pair of events in different ranges.

Representing events in the same range and category by one node provides a clean and simple interface. It also allows easy comparison between records. However, overlapping events need to be handled in a better way.

Similan allows users to sort by only one ranking criterion at a time. An interface that allows users to sort by many ranking criteria in a sequential order could be added to make the interface more flexible.

7 Conclusion

This paper proposes a TCS measure, a novel similarity measure for temporal categorical data. Briefly, the TCS measure is a combination of time differences between events, and number of missing and extra events.

We also introduces Similan, an interactive tool that facilitates similarity searching and search results visualization for temporal categorical data. The alignment feature allows users to pre-process the dataset by aligning events by a sentinel category. Users are allowed to rank the temporal categorical records by many ranking criteria derived from the TCS measure. The scatterplot provides an overall distribution of search results. The drill-down approach allows users to explore by selecting a region on the scatterplot. The comparison panel provides advanced exploration of relationships between records.

A usability study has been conducted to evaluate the interface. The Similan interface was comprehensible to users but they had a harder times understanding the TCS measure. Users expressed strong opinions that Similan can help them find similar records from temporal categorical data.

Acknowledgments

The authors would like to thank Dr. Samir Khuller for his guidance about the algorithms, Dr. Amol Deshpande for his guidance about the similarity measures, Taowei David Wang for his thoughtful comments, and our usability study participants for their time. We appreciate partial support from Washington Hospital Center through their grant on Discovering Patterns of Events in Patient Histories.

References

[1] T. D. Wang, C. Plaisant, A. J. Quinn, R. Stanchak, S. Murphy, and B. Shneiderman, "Aligning temporal data by sentinel events: discovering patterns in electronic health records," in *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2008, pp. 457–466. [Online]. Available: http://portal.acm.org/citation.cfm?id=1357054.1357129

[2] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. [Online]. Available: http://dx.doi.org/10.1002/nav.3800020109

[3] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957. [Online]. Available: http://www.jstor.org/stable/2098689

[4] J. Seo and B. Shneiderman, "A rank-by-feature framework for interactive exploration of multidimensional data," *Information Visualization*, vol. 4, pp. 96–113, 2005. [Online]. Available: http://portal.acm.org/citation.cfm?id=1103520.1103524

[5] A. C. Tsoi, S. Zhang, and M. Hagenbuchner, "Pattern discovery on australian medical claims data-a systematic approach," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 10, pp. 1420–1435, 2005.

[6] H. J. Frost and P. M. Massagli, "Social uses of personal health information within patientslikeme, an online patient community: What can happen when patients have access to one another's data," *J Med Internet Res*, vol. 10, no. 3, p. e15, May 2008. [Online]. Available: http://www.jmir.org/2008/3/e15/

[7] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," *VLDB '08*, August 2008.

[8] M. Saeed and R. Mark, "A novel method for the efficient retrieval of similar multiparameter physiologic time series using wavelet-based symbolic representations." *AMIA Annual Symposium Proceedings*, vol. 2006, 2006, pMC1839671.

[9] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *KDD Workshop*, pp. 359–370, 1994.

[10] E. Ristad and P. Yianilos, "Learning string-edit distance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 5, pp. 522–532, May 1998.

[11] R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[12] V. I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb. 1966.

[13] W. E. Winkler, "The state of record linkage and current research problems," Statistical Research Division, U.S. Census Bureau, Tech. Rep., 1999.

[14] H. André-Jönsson and D. Z. Badal, "Using signature files for querying time-series data," in *PKDD '97: Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery*. London, UK: Springer-Verlag, 1997, pp. 211–220.

[15] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. J. Keogh, "Discovering similar multidimensional trajectories," *ICDE*, 2002.

[16] Y. Chen, M. A. Nascimento, B. C. Ooi, and A. K. H. Tung, "Spade: On shape-based pattern detection in streaming time series," *icde*, vol. 0, pp. 786–795, 2007.

[17] H. Liu, Z. Ni, and J. Li, "Time series similar pattern matching based on empirical mode decomposition," *isda*, vol. 1, pp. 644–648, 2006.

[18] V. Tseng, L.-C. Chen, and J.-J. Liu, "Gene relation discovery by mining similar subsequences in time-series microarray data," *Computational Intelligence and Bioinformatics and Computational Biology, 2007. CIBCB '07. IEEE Symposium on*, pp. 106–112, April 2007.

[19] J. Assfalg, H.-P. Kriegel, P. Kroger, P. Kunath, A. Pryakhin, and M. Renz, "Similarity search on time series based on threshold queries," *EDBT*, 2006.

[20] D. P. Bertsekas, "A new algorithm for the assignment problem," *Mathematical Programming*, vol. 21, pp. 152–171, Dec. 1981. [Online]. Available: http://dx.doi.org/10.1007/BF01584237

[21] S. Ma, J. L. Hellerstein, C. shing Perng, and G. Grabarnik, "Progressive and interactive analysis of event data using event miner," *icdm*, vol. 00, p. 661, 2002.

[22] S. M. Powsner and E. R. Tufte, "Graphical summary of patient status," *The Lancet*, vol. 344, pp. 386–389, Aug. 1994. [Online]. Available: http://www.sciencedirect.com/science/article/B6T1B-49NR4SR-Y9/1/1177a0f51d95d831c6516a20c0e69410

[23] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman, "Lifelines: using visualization to enhance navigation and analysis of patient records." *Proc AMIA Symp*, pp. 76–80, 1998. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/9929185

[24] D. L. Alonso, A. Rose, and C. Plaisant, "Viewing personal history records: A comparison of tabular format and graphical presentation using lifelines," *Behaviour and Information Technology*, vol. 17, no. 5, pp. 249–262, September 1998.

[25] B. B. Bederson, J. Grosjean, and J. Meyer, "Toolkit design for interactive structured graphics," *IEEE Transactions on Software Engineering*, vol. 30, no. 8, pp. 535–546, 2004.

[26] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pp. 336–343, Sep 1996.