The Visual Computer (2019) 35:1013–1026
https://doi.org/10.1007/s00371-019-01673-y

ORIGINAL ARTICLE

# TimeCluster: dimension reduction applied to temporal data for visual analytics

Mohammed Ali[1,2] · Mark W. Jones[1] · Xianghua Xie[1] · Mark Williams[3]

Published online: 9 May 2019
© The Author(s) 2019

**Abstract**
There is a need for solutions which assist users to understand long time-series data by observing its changes over time, finding repeated patterns, detecting outliers, and effectively labeling data instances. Although these tasks are quite distinct and are usually tackled separately, we present an interactive visual analytics system and approach that can address these issues in a single system. It enables users to visualize, understand and explore univariate or multivariate long time-series data in one image using a connected scatter plot. It supports interactive analysis and exploration for pattern discovery and outlier detection. Different dimensionality reduction techniques are used and compared in our system. Because of its power of extracting features, deep learning is used for multivariate time-series along with 2D reduction techniques for rapid and easy interpretation and interaction with large amount of time-series data. We deploy our system with different time-series datasets and report two real-world case studies that are used to evaluate our system.

**Keywords** Time-series data · Visual analytics · Sliding window · Dimension reduction · Time-series graph · 2D projection · Repeated patterns · Outliers · Labeling

## 1 Introduction

Due to the growing amount of collected time-series data and the increase in the complexities involved in its understanding in practice, processing and analyzing such data have become more substantial procedures to understand the characteristics of the data and obtain meaningful insights and knowledge from it. Different approaches have been developed to extract useful information from raw time-series data including data-mining. In many situations, however, automated techniques do not achieve satisfactory results, so experts rely on visual analytics tools to perform their tasks [17]. Visual analytics [23] combines the strengths of machine capabilities with human capabilities to facilitate exploration, analysis, understanding, and providing insights. The visual analytics process aims to tightly couple automatic analysis methods and interactive visualization to gain knowledge from raw data and present a possible chance for analysts, through interaction tasks, to analyze, explore and understand data.

The time-series data are commonly represented as a time-series graph. When dealing with a small data space, time-series graphs are effective, but performing common tasks such as anomaly detection, extracting frequently occurring patterns, classifying time-series subsequences into clusters of similar patterns, or getting an overview of an uncompressed or compressed time-series graph for large time-series data become more challenging.

There is a considerable amount of works in information visualization which examine alternative visual encodings, such as color-fields [3,16,47] and horizon graphs [22,38]. They have focused on elementary visual tasks that evaluate estimation, such as, point comparison and discrimination

**Electronic supplementary material** The online version of this article (https://doi.org/10.1007/s00371-019-01673-y) contains supplementary material, which is available to authorized users.

✉ Mohammed Ali
884715@swansea.ac.uk

Mark W. Jones
m.w.jones@swansea.ac.uk

Xianghua Xie
x.xie@swansea.ac.uk

Mark Williams
mark.williams@southwales.ac.uk

1 Swansea University, Swansea, UK

2 King Khalid University, Abha, Saudi Arabia

3 University of South Wales, Pontypridd, UK

tasks, or estimation of averages. Thus, the results say very little about how the users assess the similarity of two or more time-series when utilizing various time-series visualizations [17]. Such tasks usually involve the notion of similarity between time-series which is sometimes inefficient [31]. Dimensionality reduction also is used to enhance the efficiency of finding repeated patterns by extracting features which usually require a discrete representation of the time-series [31]. Thus, locating such patterns is not easy and requires the user to have a better understanding of where repeated patterns (clusters) or outliers (anomalies) occur especially for the data that have long periods, and how relationships between data change over time.

In this paper, we investigate a methodology for visualization and interaction with large time-series. The tasks of anomaly discovery and discovery of frequent patterns are quite distinct and are usually tackled separately. Our approach addresses these issues in a single system. The proposed approach uses a sliding window and dimensionality reduction techniques which aim to depict a large time-series data as points into a 2D connected scatter plot. The sliding window moves along the time axis and relies on two main factors: stride between the existing window and next window; and the window size. Each vector derived from the sliding window will be considered as a point in high-dimensional space representing the phenomenon under consideration. To enable analysis and exploration, we apply dimensionality reduction techniques to project the points to two dimensions. The two-dimensional projections are used to simplify navigation techniques and prevent clutter. Ultimately, the whole time-series data are presented in one image. From the resulting projections, selecting any points should allow the user to know why they are similar or different, where outliers (anomalies) occur, where clusters (repeated patterns) occur, and how the relations between points evolve (connected lines between points). The methodology applies to univariate and multivariate time-series data and demonstrates how it aids the user to label patterns in time-series dataset.

The proposed visual analytics system and approach assists users to understand and visualize a large time-series data using both connected scatter plots which represent the entire dataset after the projection to the new space simultaneously with time-series graph. It provides novel interactive solutions to many pattern discovery issues such as anomalous or frequent patterns. It also assists to display how the form of data develops over time helping researchers to see, understand, and compare the phenomena under consideration over time. Dense clusters allow the rapid labeling of similar patterns. Also, selecting subsets of data in the original time series view allows them to be located in the connected scatter view.

Overall, our contributions in this work are that we:

1. Demonstrate a visual analytics system that aids identification of patterns, repeated patterns (clusters), outliers (anomalies), and transitions between states in large time-series data.
2. Use a deep convolutional auto-encoder (DCAE) to apply our approach to multivariate time-series data. Thus, our approach will be suitable for both univariate and multivariate time-series data.
3. Provide visual comparisons of the different approaches to dimension reduction in our accompanying video.
4. Evaluate our approach and system with two case studies utilizing two different time-series datasets.

In the following sections, we present a review of the challenges, approaches, and systems that are relevant to our work, as well as present and evaluate our approach.

## 2 Background and related work

In this section, we discuss the prior works that are pertinent to our work. We divide the related work into three categories: (1) pattern discovery, (2) labeling time-series data, (3) dimensionality reduction techniques. A brief description of each category and some works that are related to it will be discussed.

### 2.1 Pattern discovery

Pattern discovery is utilized to detect interesting patterns in the data. The presence of interesting patterns is discovered without any prior assumptions. Under this group, there are two main sub-tasks which are: (a) Identifying outliers (anomalies) in time-series which aims to extract data that deviates from other data and does not conform to an expected pattern in the data. (b) Identifying common patterns (motifs) in time-series that aims to find frequently occurring patterns in a large dataset.

VizTree [30] uses symbolic aggregate approximation (SAX) to discretize time-series data into a sequence of symbols. A suffix tree encapsulates the global and local structures of time-series data. Patterns are generated by moving a sliding window along the time-series data which are represented by a horizontal tree visualization. Performing different pattern discovery tasks is available in VizTree such as finding frequently occurring patterns (motif discovery) by selecting the thickest branches across the tree and surprising patterns (anomaly detection) by selecting the thinnest branches across the tree.

The Viztree algorithm is fast and effective but it assumes prior knowledge of the length of the motif to be found. Therefore, motifs with lengths other than the pre-defined length would remain undetected. However, if the algorithm re-runs

multiple times using different motif lengths, motifs could be detected, but would reduce its efficiency [56,57].

To overcome fixed pattern length, Li et al. [29] introduced a system for detecting variable length motifs by grammar induction on symbolic representations. Senin et al. [42] extend GrammarViz [29] to incorporate the parameterless discovery of anomalies in time-series data. However, processing multi-dimensional data [34] is unavailable in GrammarViz.

Ordonez et al. [37] add radial representations to their line graphs to simplify the motif analysis process; however, this could create an overlapping problem because multiple lines are drawn along the circular axes [10]. TimeSeer [36] uses scagnostics to identify scatter plots of data attributes at each time index. Using the statistical summaries lets the user to explore pairs of variables which helps detecting outliers in the time-series data. The interface of TimeSeer has lots of details which may require user training for data exploration [45]. Legg et al. [26] employ a sketch-based system for query-by-example search for similar patterns.

TimeSearcher2 [11] allows pattern discovery through query by example. Filtering is utilized to decrease the size of the search and allow users to explore multi-dimensional data using graphs and coordinated tables. The rubberband selection is also applied allowing users to perform a pattern search utilizing Euclidean distance. At least, one pattern must be provided to start the matching process. Similarly, TimeClassifier [53] requires the user to select one behavioral instance in order to perform the matching process. Therefore, both systems demand the user to have an overall notion of what constitutes intriguing or repeating patterns to be selected.

### 2.2 Labeling time-series data

Labeling is the task of providing labels $y$ to given input instances $x$; thus, labels can be utilized to find functions $f$ that map instances to labels, for example, $f(x_1, x_2) = y$ where $x_1$ and $x_2$ are instances and $y$ is the label [8]. Bernard et al. [7] conduct a study to compare and assess the performance of various labeling strategies using machine learning and visual analytics. Both fields have individual strengths and weaknesses. Machine learning follows a model-centered approach while visual analytics employs user-centered approaches. They conclude that visual analytics (visual-interactive labeling) can perform better than machine learning (active learning) provided that dimension reduction successfully separates the class distributions. Alsallakh et al. [5] introduce a visual analytics approach which supports the user with automated segmentation results and assists domain experts to inspect the results, to identify segmentation problems, and correct mislabeled segments accordingly. Rohlig et al. [39] propose a visual analytics system to help the user to comprehend the influence of parameters on the resulting segmentation and labeling. Thus, it supports subsequent decision making and enhances higher accuracy as well as confidence in the results. For the exploration of time-series data, Walker et al. [53] introduce TimeClassifier a visual analytics system for the classification of time-series to facilitate in labeling smart sensor data. They also introduce TimeNotes [52] which supports interactive selection, hierarchical navigation, exploration, and comparison of time-series data. Similar to our use case, sequences are labeled with overlaid colored regions illustrating labeled animal behavior.

### 2.3 Dimensionality reduction techniques

An efficient motif discovery algorithm for time-series would be beneficial to summarize and visualize large datasets. Dimensionality reduction is a way to enhance the efficiency of extracting patterns in data [31]. Utilizing dimension reduction in combination with further visual encodings that reveal the internal state of the learning model enhances the performance of visual-interactive labeling [7].

Principal component analysis (PCA), as a feature extraction method, is applied to time-series data [27,46,59,60]. It is used to decrease the dimensions of a $d$-dimensional dataset by decreasing it to a $k$-dimensional subspace (where $k < d$). t-Distributed Stochastic Neighbor Embedding (t-SNE) is used [13,28,48,58] which helps to visualize high-dimensional data by giving each datapoint a location in a two- or three-dimensional map. Huang et al. [21] use deep convolutional auto-encoder (DCAE), based on deep convolutional neural network (CNN), to hierarchically model tfMRI time-series data in an unsupervised manner. DCAE is a powerful method for learning high-level and mid-level abstractions from low-level raw data. It has the ability to extract features from complex and large time-series in an unsupervised manner.

In this work, dimension reduction visually clusters similar patterns removing the need for length matching computation. Previous works [4,48] also use sliding window approach and PCA, but here we introduce the option to switch between different dimensionality reduction techniques (t-SNE, UMAP and PCA) and also deep convolutional auto-encoder (DCAE). We develop the methodology further to incorporate DCAE work with multivariate time-series, provide a thorough consideration of parameters in the accompanying video, and discuss the results.

## 3 Overview of the methodology

Our method is designed for detecting, exploring and interpreting outlier patterns (anomalous) and repeated patterns (clusters) in large time-series data. In this section, we intro-

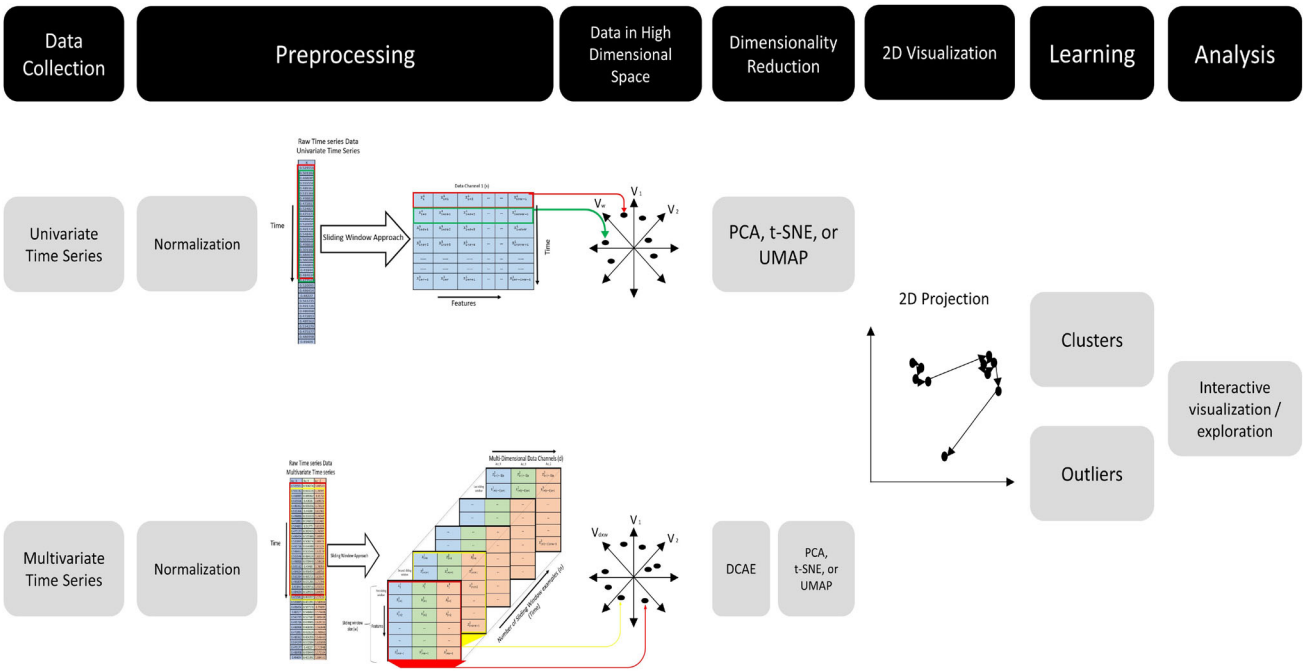



**Fig. 1** An overview of our proposed visual analytics approach. It starts from raw time-series data and ends allowing users to interact with the system and changing the parameters which help to improve and fit the users tasks. Transforming time-series data into a 2D space (points) passing through a multi-step process including preprocessing (the time-series data are in high-dimensional space after this step), dimensionality reduction techniques are used to project each sliding window into a 2D space (if the number of time-dependent variables is univariate PCA, t-SNE, or UMAP are applied directly on the sliding window matrix, but if it is multivariate, DCAE is applied to extract important features which are then projected into a 2D space using PCA, t-SNE, or UMAP), visualizing the data into a 2D space, assisting users to detect outliers and frequent patterns in large time-series data, and allowing users to interact with the system and customize views

duce our pipeline (Fig. 1) which helps users to visualize, understand, explore, and validate large time-series data.

## 3.1 Preprocessing

Data preprocessing transforms the raw data. In this step, we, respectively, apply normalization and sliding window approach.

### 3.1.1 Normalization

In our case, we use unity-based normalization Eq. (1) to set all values into the range [0,1].

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

### 3.1.2 Sliding window approach

Define a continuous multivariate time-series data D of dimension $d$ with $n$ time-steps, $D = X_1, X_2, \ldots, X_n$, where each $X_i = \{x_i^1, \ldots, x_i^d\}$. Let $w$ be the window width, $s$ the stride, and $t$ the start time of a sliding window in the data.

Define a new matrix $Z_k$ where each row is a vector of size $w$ of data extracted from the $k^{\text{th}}$ dimension.

$$Z_k(w, s, t) = \begin{bmatrix} x_t^k & x_{t+1}^k & \cdots & x_{t+w-1}^k \\ x_{t+s}^k & x_{t+s+1}^k & \cdots & x_{t+s+w-1}^k \\ \vdots & \vdots & \ddots & \vdots \\ x_{t+(r-1)s}^k & x_{t+(r-1)s+1}^k & \cdots & x_{t+(r-1)s+w-1}^k \end{bmatrix}$$

where $r$ is the number of desired rows, and $t + (r-1)s + w - 1 \leq n$

When more than one dimension of the multivariate data is used, the data are interleaved as depicted in (Fig. 1). As a default setting, the values of the window size (width) $w$ and stride (offset) $s$ have their default values where $w = 60$ and $s = 1$. However, they can be interactively changed using a slider in the system interface which gives the user control over the parameters $w$ and $s$ and helps to get insight into behaviors at different resolutions. These values are explored in the accompanying video. The overlapping between windows is very useful for avoiding lost data and facilitating the smooth transition between time-steps after reducing the

dimensionality of the features. Also, it helps to capture local temporal patterns in the datasets.

### 3.2 Dimensionality reduction (DR)

The resultant matrix from the sliding window approach is treated as points in high-dimensional space. Each such point represents the phenomena that occur at a different time-interval. We use DR techniques to provide an alternative view for users to visually analyze and explore the time-series data. The aim is to reduce the feature space to two dimensions using DR techniques. A higher-level abstraction is also generated which represents the data while preserving the shape characteristics of the original data during the reduction process. In general, choosing a particular DR technique is important in our approach because the visualization phase is dependent on it.

There are several linear and nonlinear DR techniques have been proposed which aim at decreasing the number of variables that describe the data [50]. The data attributes of the features in the lower-dimensional subspace are therefore approximated to the geometric attributes of the data in the original high-dimensional space. In our work, different linear and nonlinear DR techniques are applied such as Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE) [49], Uniform Manifold Approximation and Projection (UMAP) [33], and deep convolutional auto-encoder (DCAE). The target of using these techniques is to differentiate and visualize high-dimensional data by giving each data point a location in a two-dimensional map, thus, different perceptions of the phenomenon under consideration will be presented which help to visualize, analyze, and facilitate exploration of large time-series datasets. To overcome the complexity of multivariate time-series, DCAE is used to reduce the features to a certain value, then PCA, t-SNE, or UMAP is applied to the reduced features to obtain a 2D visualization while univariate time-series is straightway reduced to a 2D using PCA, t-SNE, or UMAP (Fig. 1).

We choose PCA as an initial DR technique. As nonlinear techniques, t-SNE and UMAP are available in the system using source code provided by the authors [33,49]. Nonlinear DR techniques could help to avoid overcrowding issues [6]. Both t-SNE and UMAP use as default the standard Euclidean distance between data points.

While t-SNE is currently the most commonly used technique, the new algorithm UMAP shows its high competitiveness compared to t-SNE [6]. t-SNE suffers from some limitations such as loss of large-scale information (the inter-cluster relationships). UMAP has a faster runtime and provides better scaling which helps to gain a meaningful organization of clusters, outliers and the preservation of continuums compared to t-SNE [6,33,51] (Fig. 2 and discussion in the case studies).

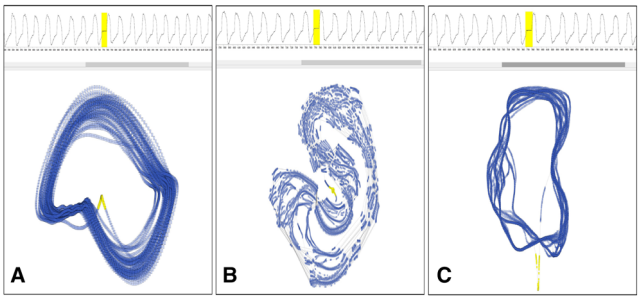



**Fig. 2** Top: (all) a selected part of the time-series graph which contains 12000 flow data points of breathing (exhalation and inhalation for one person). Bottom: connected scatter plot of the data after applying our methodology, **a** PCA, **b** t-SNE, and **c** UMAP. The yellow highlight in the time-series graph with the corresponding yellow points are shown in the connected scatter plot indicates the breathing at that moment was completely distinct which is clearly obvious once it is labeled in time-series graph

### 3.3 Deep convolutional auto-encoder (DCAE)

One of the practical applications of auto-encoders is dimensionality reduction for data visualization. It can learn data projections that are more interesting than other basic techniques [15]. Deep convolutional auto-encoder (DCAE) is a strong nonlinear dimensionality reduction method [14]. Compared to the conventional auto-encoder, DCAE has fewer parameters than the conventional auto-encoder which means less training time. Also, DCAE uses local information to reconstruct the signal while conventional auto-encoders utilize fully connected layers to globally do the reconstruction. DCAE is an unsupervised model for representation learning which maps inputs into a new representation space. It has two main parts which are the encoding part that is used to project the data into a set of feature spaces and the decoding part that reconstructs the original data. The latent space representation is the space where the data lie in the bottleneck layers.

Reducing the dimensionality is achieved by unsupervised training of an encoder and a decoder neural network, minimizing the reconstruction error [19,32]. The latent features resulting from the encoder are flattened, and one of PCA, UMAP, or t-SNE is then used to reduce them to 2D for visualization.

#### 3.3.1 Architecture

The loss function of the DCAE is defined as the error between the input and the output. DCAE aims to find a code for each input by minimizing the mean squared error (MSE) between its input (original data) and output (reconstructed data). The MSE is used which assists to minimize the loss; thus, the network is forced to learn a low-dimensional representation of the input [14,19].

**Table 1** Architecture of the deep convolutional auto-encoder with the dense layer highlighted in bold

| Layers | Shape | Filter size | Number of kernels | Number of units | Activation |
|---|---|---|---|---|---|
| Input | 60 × 3 | | | | |
| Convolution | 60 × 64 | 10 | 64 | | ReLu |
| MaxPool | 30 × 64 | 2 | | | |
| Convolution | 30 × 32 | 5 | 32 | | ReLu |
| MaxPool | 15 × 32 | 2 | | | |
| Convolution | 15 × 12 | 5 | 12 | | ReLu |
| MaxPool | 5 × 12 | 3 | | | |
| Flatten | | | | | |
| Dense | | | | **60** | **Linear** |
| reshape | 5 × 12 | | | | |
| Convolution | 5 × 12 | 5 | 12 | | ReLu |
| Upsample | 15 × 12 | 3 | | | |
| Convolution | 15 × 32 | 5 | 32 | | ReLu |
| Upsample | 30 × 32 | 2 | | | |
| Convolution | 30 × 64 | 10 | 64 | | ReLu |
| Upsample | 60 × 64 | 2 | | | |
| Output | 60 × 3 | 10 | 3 | | Linear |

For convenience, all layers input and output shape, filters size, number of kernels, number of units, and activation functions of the DCAE are summarized in Table 1 and can be explained as detailed below:

The network architecture consists of three main parts which are encoding part, encoded representation or bottleneck (compressed representation), and decoding part. The shape of the input and output layers are 60 × 3. In the encoding part, there are three convolutional layers, and each layer is followed by pooling layer. The max pooling is used which is a down-sampling operation on feature maps. Using max pooling has two main benefits which are: it obtains translation-invariant features [40]. Second, it ultimately reduces the computational cost for the upper layer [21]. It is followed by fully connected layers, which take the output of the last convolution layer and flattens it to 60 neurons. In the decoding part, it has three convolutional layers, and each layer is followed by the upsampling layer which is a process that is mainly used to increase the size of the input data. It works by repeating each temporal step $n$ times along the time axis. In our case $n = 3$ after the first convolutional layer, and $n = 2$ after the second and third convolutional layers in the decoding part. The upsampling process does not apply any particular function, just iterates the contents of the input. The last convolutional layer has output shape which is of the same shape as the input. As activation function, a Rectified Linear Unit activation function (Relu) [35], defined as $\mathrm{ReLU}(x) = \max(0, x)$, is used in all of the convolutional layers except the hidden layer and the final layer of the decoder part where linear activation function is used.

Using the Relu activation function has some advantages which have been discussed in previous studies [25,35] for example, it reduces the probability of vanishing gradient which often occurred when the model is deep. Another example is that it adds nonlinearity and guarantees the robustness of the system against noise in the input signals [2]. The output of the last layer in the decoding part is the reconstructed data of the original input where linear activation is used. Also, linear activation is used on the latent space layer (fully connected layer or hidden layer) to preserve the extracted features from the last conventional layer in the encoding part which will be used as input to the decoding part. It should be noted here that the features from the hidden layer are the features that we are looking for, so we use linear activation to ensure that they are not modified to be ready for the next process (2D visualization). The number of feature maps, size of filter and depth of the model are set based on the reconstruction error on validation set.

### 3.3.2 Training

The proposed model was implemented using the libraries TensorFlow [1] and Keras [15] for building, training, and processing the DCAE. Using all queries as a preprocessing stage, the model is trained end-to-end in an unsupervised manner before the visualization starts. Adam optimizer [24] is used which is computationally efficient, requires little memory, and appropriate for problems with noisy data. Each batch contains 100 random shuffled windows from the time-series data. The DCAE is trained to transform the time-series data into latent representation and then reconstruct the original

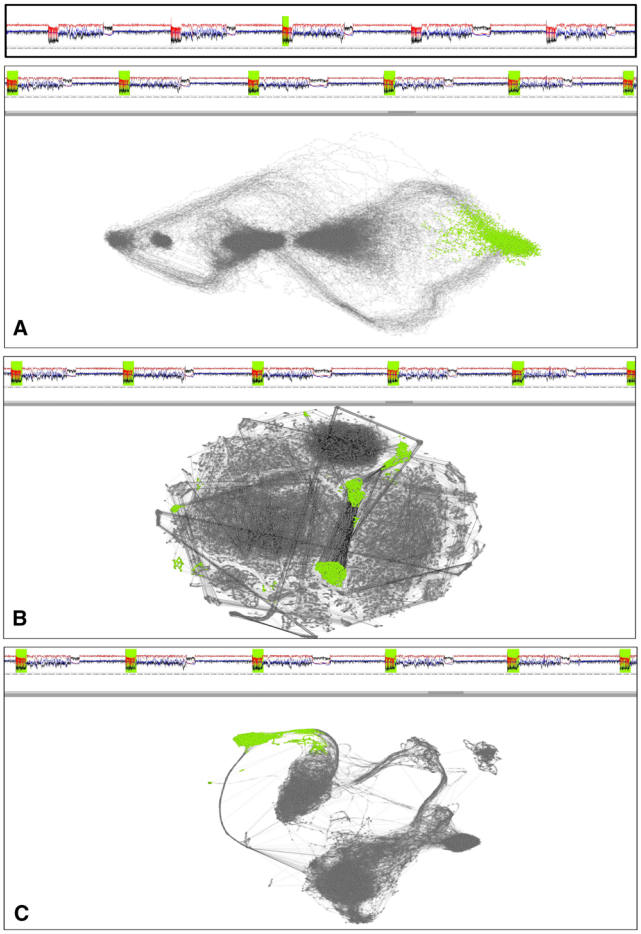



**Fig. 3** DCAE followed by **a** PCA, **b** t-SNE, and **c** UMAP to create the 2D visualization. This figure also shows the Auto-Labeling process. From the time-series graph (top), the user chooses a portion of data. Based on Euclidean distance, all windows that match below a threshold are labeled with the same color in both graphs. In this example, the selected cluster is corresponding to the Descent Phase of Dive

input or get an optimal approximation of the implicit data representation by minimizing the reconstruction error. DCAE is trained to perform the feature extraction process. After that, the features in the latent space (bottleneck) are projected into a 2D space using the previous DR techniques (Fig. 3).

## 4 Visualization and interaction techniques

The widely referenced mantra "overview first, zoom and filter, and then details-on-demand" by Shneiderman [44] is employed. As we show in this paper, our system fits neatly into these principles. In one image, the overview of the large dataset is obtained after applying the proposed approach using 2D connected scatter plot (Fig. 4). The user can zoom in a particular area, and the detail on demand will be provided for identified patterns. Sedlmair et al. [41] suggest using 2D scatter plots, as the most promising approach, to explore the output of different dimensionality reduction techniques. They also advocate avoiding interactive 3D scatter plots for dimension reduction data, especially for cluster verification tasks.

The time-series graph (Fig. 4A) displays the original data rendered on the time axis. To simplify navigation techniques and prevent clutter, the connected scatter plot (Fig. 4B) is used which displays the transformed points after applying dimension reduction using any of the described techniques. While the connected scatter plot is a simple visualization technique, it has very specific functions in our approach. Every sliding window is represented as a dot in the plot after the projection process (Fig. 4C, D). Before labeling, all points have the same color and transparency, and when they are concentrated in one area, the densities are accumulated. Lines are used to connect consecutive points preserving the temporal ordering of the data and allowing the user to see temporal connections (Fig. 4B). Thus, the point is linked to the previous point (inner) and to the posterior point (outer) as an indication of the flow of time. Lines can be omitted as one of the options provided in the system. Another option that is available is path extractions (Fig. 4E). It helps the user to track the transition between points or clusters. The size and stride of the sliding window can be also modified. If the stride has a bigger value than the window size, there will be some data uncovered, so the system limits the stride option to be less than the window size (see accompanying video).

For navigating large information spaces, filtering and zooming are important tasks which support panning or scrolling through the data. Selecting and zooming could be utilized to facilitate fast and interactive exploration of large datasets which help to define the level of detail the user requires (Fig. 4C). In the time-series graph, the width of the graph is expanded as the zoom is increased, and the scroll bar allows the user to scroll smoothly through the expanded time-series. In the connected scatter plot, scrolling and zooming display the visualization at different levels of abstraction. The user can zoom in on regions of interest to emphasize interesting data for instance, clusters, outliers, etc. That will give the user direct control over the mapped data and aid for quickly locating and a close-up visual displaying of clusters. Smooth zooming is applied to assist the user to maintain their sense of context and position. In the connected scatter plot, the smooth zooming is achieved through three levels: zooming into the whole image, zooming into a specific cluster, and zooming inside the specific cluster.

For details-on-demand, the idea of linking and brushing is implemented to connect the two visualization techniques, so the change to the representation in one view affects the representation in the other. Linking and brushing techniques are beneficial for instance, assisting to overcome the shortcomings of a single visualization technique, combining different visualization techniques, providing more information, etc.

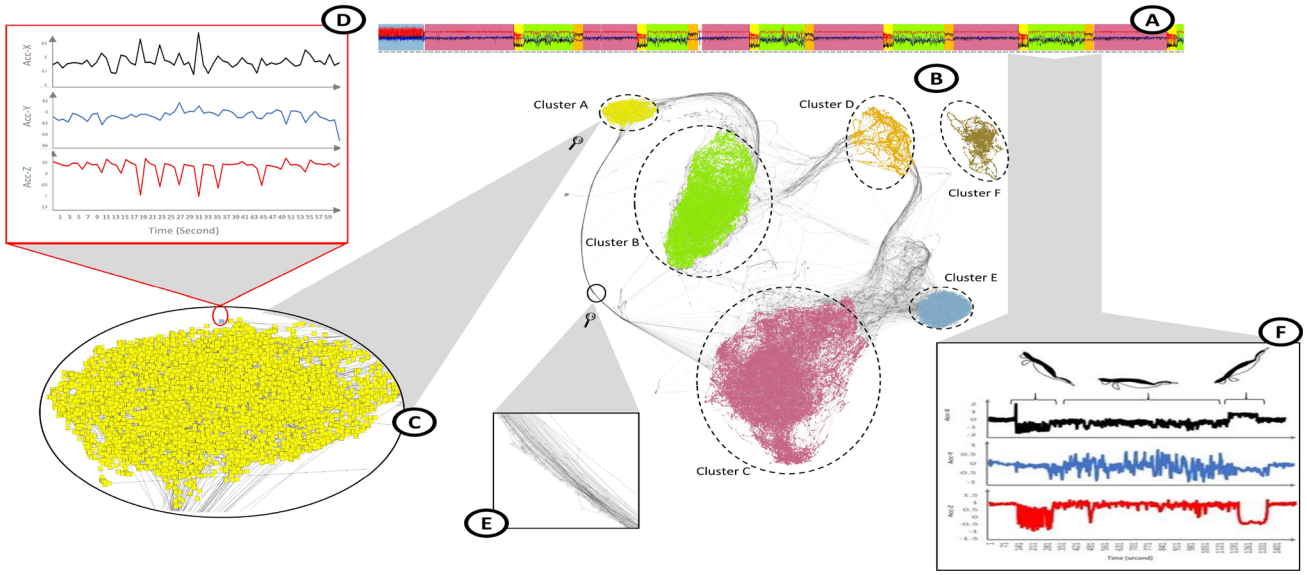



**Fig. 4** The overview of the system (also see video) (A) Time-series graph for raw data (multivariate time-series). (B) Connected scatter plot for the time-series data after applying the proposed approach which reveals six clusters and transitions between them. The DCAE following by a dimension reduction technique (UMAP) is applied on the data that is collected from a Cormorant bird using a sensor. Various colors reveal various clusters were: cluster A, B, C, D, E, and F are, respectively, Descent Phase of Dive, Bottom Phase of Dive, Surface Swimming, Ascent Phase of Dive, Flight, and the beginning and the end of the dataset. (C) Zooming in an area of interest (cluster A), (D) Drawing time-series graph for the selected point in connected scatter plot, (E) Zooming of the transitions (connected lines) from cluster C to cluster A, and (F) *X*, *Y*, and *Z* acceleration during a single cormorant dive where the changing in posture during descent, swimming, and ascent are obvious as shifts in the time-series graph

In our system, both time-series graph and connected scatter plot are linked. The desired data can be chosen in either view, and the highlighted color is automatically reflected on both graphs to distinguish selected data, hence, patterns, relationships, clusters, or outliers could be easily visualized, inspected, and differentiated. The selecting and highlighting could be performed in both graphs, and the selected data will be colored in the graph concurrently with the corresponding items in the other graph which is helpful to demonstrate a labeling task for repeated patterns, outliers, etc (Figs. 4, 6).

Query by example is also provided by the system to achieve automatic labeling, where the user selects the interesting data by applying rubber band brushing. Thus, a timebox as a rectangular region will highlight the interesting pattern (Fig. 3) (top time series graph). The matching process will be executed to find similar occurrences in the data using Euclidean distance (other similarity measures can be introduced). The threshold is set by the user, where 0 means the patterns are completely identical. Euclidean distance is calculated between the selected instance and the remaining of the series data; therefore, all windows that match below a threshold are labeled with the same color in both graphs (Fig. 3).

## 5 Case study

The capabilities of the system are demonstrated by analyzing real-world data from two domains: medicine and biology. We collaborated with experts who provided us with datasets and offered several suggestions and opinions to improve the system performance giving the user more control over exploration and analysis.

Two time-series datasets are presented. One of the datasets is univariate time-series (breathing patterns), and the second is multivariate time-series (triaxial accelerometers recording animal activity).

### 5.1 Case study 1: breathing patterns

The respiratory rate is an important vital sign to the health status of the human. There are various kinds of normal and abnormal respiration. Evaluating breathing patterns is important and helps the clinician in understanding the patient's current status [55]. Inspection of the pattern of breathing will yield clues of the disease process, independent of the rate measurement. Abnormal patterns of breathing suggest the possibility of diseases [55,61]. In preparation for analysis anomalies in the dataset should be removed. These are: a participant may sigh on inspiration or swallow or cough on expiration which interrupts the normal breathing patterns.

Looking for patterns, such as repeated patterns or abnormal tidal breathing patterns, is important, but when using time-series graph, revealing such patterns becomes a complex process. Each of the 48 participants have about 12,000

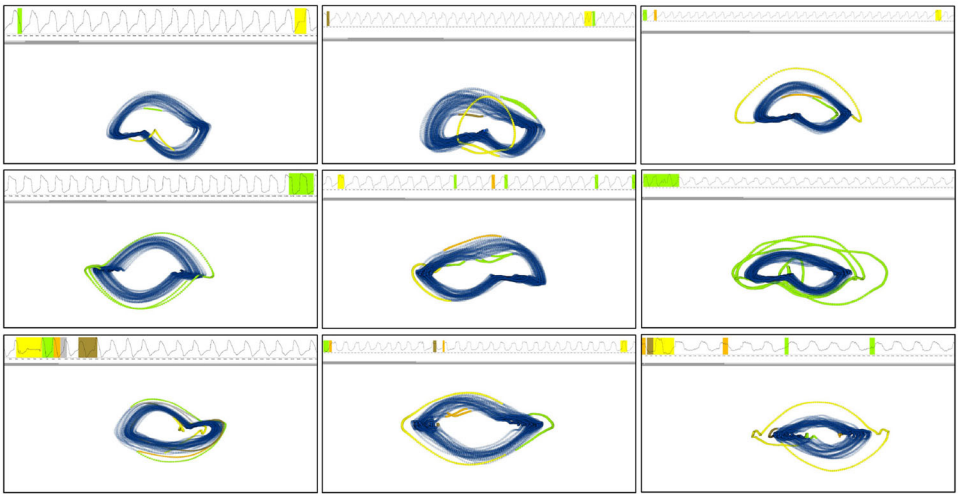


**Fig. 5** Nine breathing patterns for nine different participants where the abnormal patterns could be easily evidenced through the connected scatter plot after applying the proposed approach (PCA)

inspiratory and expiratory flow readings. The two main issues arising are: the comparing of all the individual breaths against each other is difficult while looking for abnormalities because of a long time-series data. Second, comparing one person with several others is complicated using time-series graphs.

After applying our approach, every inspiratory and expiratory breath are represented as one loop by applying PCA and UMAP (Fig. 2a, c). Using our approach allows users to see all breaths in one view which facilities finding irregular patterns (Fig. 5). Visual outliers correspond to problematical breaths (Fig. 6), which can be confirmed by brushing the outlier points in the connected scatter plot, e.g., the outlier in (Fig. 6) is highlighted in yellow and is found to correspond to an interrupted breath (see time-series graph). It is beneficial to eliminate abnormal patterns so they do not impact in any of the further statistical analysis. As (Fig. 2) shows, we found that outliers corresponding to breathing anomalies were visually obvious when using PCA or UMAP, but were not easy to detect when using t-SNE. Including the option to switch between dimension reduction techniques within a visual analytics system can lead to improved interaction with the data.

Another functional requirement is to be able to compare patterns far apart within the time-series. Identifying repeated patterns is a hard task specially with long time-series. After applying the proposed approach, similar patterns are clustered in the same area in the connected scatter plot. Thus, identifying such patterns become more simple. Repeated patterns can be confirmed by brushing the dense area (points), e.g., the repeated patterns in Fig. 6 are highlighted in green and orange are found to correspond to similar patterns (see time-series graph). As Fig. 2 shows, when using PCA or UMAP, repeated patterns are obvious, but they are hard to be located when using t-SNE.

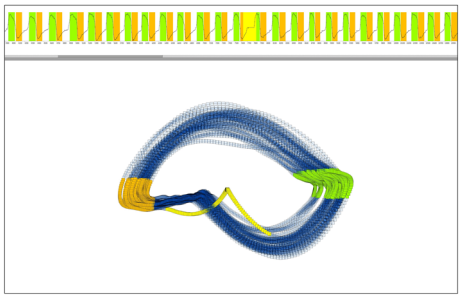


**Fig. 6** Top: time-series graph with overlaid colored regions indicating to the selected clusters in the connected scatter plot. Bottom: connected scatter plot where frequent patterns (orange and green) and outlier patterns (yellow) can be allocated with a distinct color that supports the identification and comparison of the data

## 5.2 Case study 2: imperial Cormorant bird

One of the attractive solutions to measure behavior in wild animals is using accelerometers [54]. The attachment of triaxial accelerometer provides quantitative data which assists biologists to monitor and determine animals behavior in their natural environment over long periods of time [9]. The three axes are corresponding to the dorsoventral ($Y$), anterior-posterior ($Z$) and lateral axes ($X$) [43]. The biologists sometimes use directly measured attributes such as pressure or temperature to be compared with derived attributes helping them in the validation of the animal activities.

Accelerometer data are presented on three separated time-series graphs, and each graph component of the signal describes the behavior over the time (Fig. 4F). Using this type of visualization is not easy to look into the triaxial nature of the data, and the correlation among axes is hard to be followed which is important to be considered during the searching process[18]. Simple visual inspection of three-line graphs in acceleration to locate behaviors is difficult because

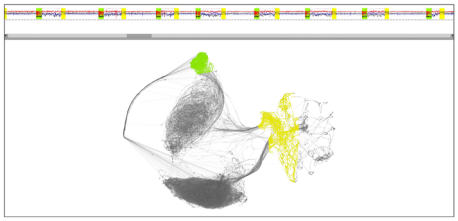


**Fig. 7** This figure shows the main user interface of our system operating on an Imperial Cormorant dataset. Top: a time series graph of the whole dataset where accelerator [respectively, *X*, *Y*, and *Z* (black, blue, and red)] and pressure (purple). The pressure is only plotted for the validation purpose and is not included in any process. Bottom: connected scatter plot where descent phase of dive (green color), ascent phase of dive (yellow color), and the rest of the behaviors (gray color). The high correlation between pressure and channels of the triaxial sensors are evident for example, during descent and ascent phases the changing of the pressure can be observed in the time-series graph supports the decision-making process

patterns may occur in different variables over a long period of time, such as repeated patterns (Fig. 4A).

Biologists from our university provided us with datasets for a device attached to an Imperial Cormorant bird. It records parameters such as triaxial acceleration, triaxial local magnetic field intensity, pressure, and temperature. Animal behavior can be derived and quantified from the triaxial accelerometer data because particular behaviors can be identified via animal posture and changes in body velocity which are extracted from accelerometers [53]. The dataset contains 173,256 multivariate measurements for the three axes of the accelerometer. Another dataset for the Imperial Cormorant bird is provided with the pressure measurements which is used for validation (Fig. 7)

The manual labeling of such data can take many days. As the video demonstrates, this methodology enables interactive labeling of the data in minutes. A major component of this is the clustering of similar features such that the user can select similar but temporally disparate features easily in the interface (see video). After applying the proposed approach, the dataset is converted to a connected scatter plot which clearly reveals six main clusters and the transitions between them. Each point in the plot represents the animal behavior for a particular duration.

The expert informs us that the dataset has five main behaviors which are Descent Phase of Dive, Bottom Phase of Dive, Ascent Phase of Dive, Surface Swimming, and Flight. After applying the proposed approach to the dataset, six clusters have been appeared. Five clusters correspond with five behaviors that are reported by the expert, and one cluster represents the beginning and the end of the dataset when the sensor was attached and detached (Fig. 4). Instead of looking for the behavior in the time-series graph which may take a long time, our approach increases the ability to detect animal postures (connected scatter plot) and behaviors (overlays—repeated patterns). Comparing between PCA, t-SNE, and UMAP which are applied after DCAE (Fig. 3), the clusters in PCA and UMAP are clear while t-SNE is more outspread. Also, the transitions between clusters are different where they are harmonious in UMAP and follow the same or near paths while in PCA they follow near paths and twisted which cause some dispersion. In t-SNE, the transitions are less clear than PCA and UMAP.

*Interaction* The selection and zooming tasks are available in either view to determine or zoom areas of interest (Fig. 4C, D). The brushing and rubberband selection tools are efficiently used in the system where the user can select an interesting area which is automatically reflected with the same color in both graphs for example, if the user selects a particular cluster in the scatter plot, the selected data will be highlighted by the same color, and all data that are associated with that cluster in time-series graph will be also highlighted with the same color (Fig. 4). For the remaining clusters, each cluster is selected in turn, is colored in the connected scatter plot where the highlighted color is automatically reflected on the line view. Thus, the expert can confirm that each cluster represents one of the behaviors in the raw data.

Edges between clusters can be selected (Fig. 4E). The user can employ region growing on a selection. Because a source point is not part of the already highlighted clusters, it can be selected which will be grown until it reaches a point that is part of an existing cluster. The source selection may have multiple points. A whole bundle of edges can be selected using this approach which is represented the transitional paths between two clusters. For example, the transitions (Fig. 4E) are selected which represent the dominant change between cluster C (Surface Swimming) to cluster A (Descent Phase of Dive). Also, dominant transitions can be obviously observed of cluster A (Descent Phase of Dive) to cluster B (Bottom Feeding), cluster B (Bottom Feeding) to cluster D (Ascent), and cluster D (Ascent) to cluster C (Surface). Cluster F (when the sensor was attached and detached) only occurs in the start and the end of the dataset which is obvious by looking to the connected lines to the cluster. Cluster E (flight) is dominating in the outset and end of the dataset. It also happens at several shorter intervals throughout the data, so some activity between those clusters can be seen. The well-defined edges between clusters can be explicated as repeated behavior which moves through those statuses with a high frequency while weaker edges indicate less frequent behavior. Our interface helps to quickly observe these types of transitions which can be labeled for further analysis.

For validation purpose, we also compare an automatic clustering approach. A hierarchical clustering method (HDBSCAN) [12,20] is used to generate the most significant clusters as a density-based clustering algorithm. It requires only one parameter which represents the minimum size of the cluster. We use the sklearn package, and we use the hdbscan package as available on PyPi in order to determine the

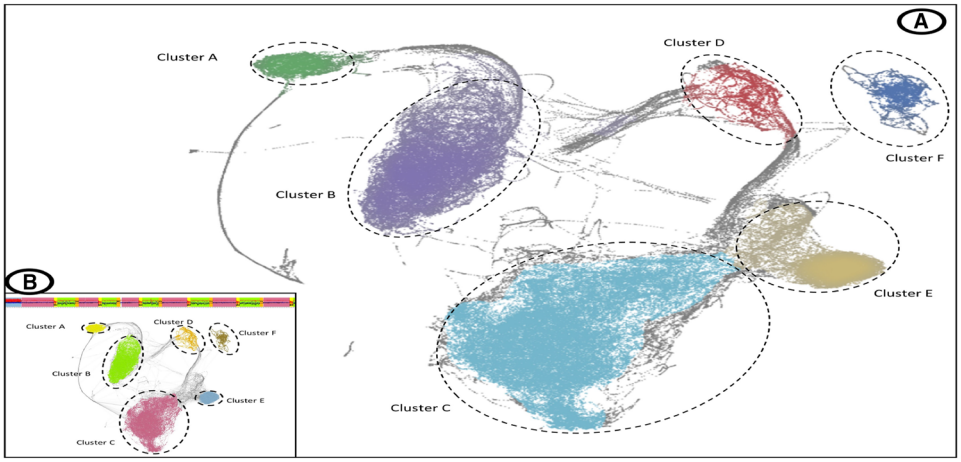



**Fig. 8** (A) HDBSCAN is used to color clusters after applying UMAP. Six clusters are clearly shown which are compatible with our system view (B)

number of clusters. It is able to correctly identify the separate clusters (six clusters) in the cormorant bird dataset after applying the UMAP (Fig. 8). Other methods, such as K-means clustering algorithm requires the users the number of clusters which are difficult to be known in advance especially in large datasets.

## 6 Conclusion

For time-series analysis, the sliding window approach together with dimension reduction techniques including auto-encoders are becoming popular. TimeCluster combines these approaches with user interaction to achieve a fast pattern identification, labeling and outlier detection. The user may vary the pipeline by choosing between different dimension reduction techniques, window and step size, and using 1D deep convolutional auto-encoder. For multivariate data, 1D deep convolutional auto-encoder has the ability to learn appropriate features resulting in less information loss. This transforms the points for 2D visualization allowing TimeCluster to summarize the whole dataset in one image and allowing interaction through multiply linked visualizations. For time-series data, we find that t-SNE over-clusters the data and presents a rather disjointed view that makes it difficult to locate outliers, or to brush similar features. Using deep learning to combine information from all channels using appropriate feature representation at the latent layer is a very effective method to find repetitive patterns or interesting anomalies that were previously unknown.

**Funding** We would like to acknowledge that this work was supported by EPSRC (Grant Number EP/N028139/1).

### Compliance with ethical standards

**Conflict of Interest** The authors declare that they have no conflict of interest.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283 (2016)
2. Abdelhameed, A.M., Daoud, H.G., Bayoumi, M.: Epileptic seizure detection using deep convolutional autoencoder. In: 2018 IEEE International Workshop on Signal Processing Systems (SiPS), pp. 223–228 (2018)
3. Albers, D., Correll, M., Gleicher, M.: Task-driven evaluation of aggregation in time series visualization. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI Conference, pp. 551–560 (2014)
4. Ali, M., Jones, M., Xie, X., Williams, M.: Towards visual exploration of large temporal datasets. In: 2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA), pp. 1–9 (2018)
5. Alsallakh, B., Bögl, M., Gschwandtner, T., Miksch, S., Esmael, B., Arnaout, A., Thonhauser, G., Zöllner, P.: A visual analytics approach to segmenting and labeling multivariate time series data. In: EuroVis Workshop on Visual Analytics, pp. 31–35. The Eurographics Association (2014)
6. Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I.W.H., Ng, L.G., Ginhoux, F., Newell, E.W.: Dimensionality reduction for visualizing single-cell data using umap. Nat. Biotechnol. **37**, 38–44 (2019)
7. Bernard, J., Hutter, M., Zeppelzauer, M., Fellner, D.W., Sedlmair, M.: Comparing visual-interactive labeling with active learning: an experimental study. IEEE Trans. Vis. Comput. Graph. **24**, 298–308 (2018)
8. Bernard, J., Zeppelzauer, M., Sedlmair, M., Aigner, W.: Vial: a unified process for visual interactive labeling. Vis. Comput. **34**, 1189–1207 (2018)

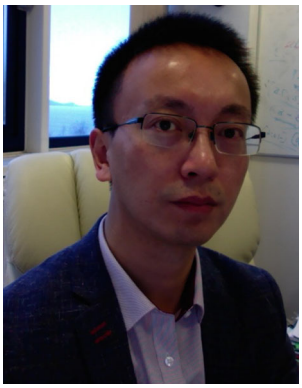
9. Bidder, O.R., Walker, J.S., Jones, M.W., Holton, M.D., Urge, P., Scantlebury, D.M., Marks, N.J., Magowan, E.A., Maguire, I.E., Wilson, R.P.: Step by step: reconstruction of terrestrial animal movement paths by dead-reckoning. Mov. Ecol. **3**, 23 (2015)
10. Brunker, A.S., Nguyen, Q.V., Maeder, A.J., Tague, R., Kolt, G.S., Savage, T.N., Vandelanotte, C., Duncan, M.J., Caperchione, C.M.. Rosenkranz, R.R., Van Itallie, A., Mummery, W.K.: A time-based visualization for web user classification in social networks. In: Proceedings of the 7th International Symposium on Visual Information Communication and Interaction, pp. 98:98–98:105 (2014)
11. Buono, P., Aris, A., Plaisant, C., Khella, A., Shneiderman, B.: Interactive pattern search in time series. In: Proceedings of SPIE, vol. 5669 (2005)
12. Campello, R.J.G.B., Moulavi, D., Sander, J.: Density-based clustering based on hierarchical density estimates. In: Pei J, Tseng VS, Cao L, Motoda H, Xu G (eds) Advances in Knowledge Discovery and Data Mining, . Springer, Berlin, Heidelberg pp. 160–172 (2013). https://doi.org/10.1007/978-3-642-37456-2_14
13. Cavallo, M., Demiralp, Ç.: Clustrophile 2: guided visual clustering analysis. IEEE Trans. Vis. Comput. Graph. **25**(1), 267–276 (2019)
14. Cheung, C.M., Goyal, P., Prasanna, V.K., Tehrani, A.S.: Oreonet: Deep convolutional network for oil reservoir optimization. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 1277–1282 (2017)
15. Chollet, F., et al.: Keras: The python deep learning library (2015). https://keras.io. Accessed 9 Feb 2019
16. Correll, M., Albers, D., Franconeri, S., Gleicher, M.: Comparing averages in time series data. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1095–1104 (2012)
17. Gogolou, A., Tsandilas, T., Palpanas, T., Bezerianos, A.: Comparing similarity perception in time series visualizations. IEEE Trans. Vis. Comput. Graph. **25**, 523–533 (2019)
18. Grundy, E., Jones, M.W., Laramee, R.S., Wilson, R.P., Shepard, E.L.: Visualisation of sensor data from animal movement. Comput. Graph. Forum **28**(3), 815–822 (2009)
19. Guo, X., Liu, X., Zhu, E., Yin, J.: Deep clustering with convolutional autoencoders. In: Liu D, Xie S, Li Y, Zhao D, El-Alfy EM (eds) Neural Information Processing. Springer, Cham, pp. 373–382 (2017). https://doi.org/10.1007/978-3-319-70096-0_39
20. Hensman, J., Lawrence, N.D., Rattray, M.: Hierarchical bayesian modelling of gene expression time series across irregularly sampled replicates and clusters. BMC Bioinform. **14**, 252 (2013)
21. Huang, H., Hu, X., Zhao, Y., Makkie, M., Dong, Q., Zhao, S., Guo, L., Liu, T.: Modeling task fmri data via deep convolutional autoencoder. IEEE Trans. Med. Imaging **37**(7), 1551–1561 (2018)
22. Javed, W., McDonnel, B., Elmqvist, N.: Graphical perception of multiple time series. IEEE Trans. Vis. Comput. Graph. **16**(6), 927–934 (2010)
23. Keim, D., Kohlhammer, J., Ellis, G., Mansmann, F.: Mastering the information age: solving problems with visual analytics. Eurographics Association (2010)
24. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). CoRR arXiv:1412.6980
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. **25**, 1097–1105 (2012)
26. Legg, P.A., Chung, D.H.S., Parry, M.L., Bown, R., Jones, M.W., Griffiths, I.W., Chen, M.: Transformation of an uncertain video search pipeline to a sketch-based visual analytics loop. IEEE Trans. Vis. Comput. Graph. **19**(12), 2109–2118 (2013)
27. Lesch, R.H., Caillé, Y., Lowe, D.: Component analysis in financial time series. In: Computational Intelligence for Financial Engineering, 1999. In: (CIFEr) Proceedings of the IEEE/IAFE 1999 Conference on, pp. 183–190 (1999)
28. Li, J., Chen, S., Zhang, K., Andrienko, G., Andrienko, N.: Cope: Interactive exploration of co-occurrence patterns in spatial time series. IEEE Trans. Vis. Comput. Graph. 1–14 (2018). https://doi.org/10.1109/TVCG.2018.2851227
29. Li, Y., Lin, J., Oates, T.: Visualizing variable-length time series motifs. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 895–906 (2012)
30. Lin, J., Keogh, E.J., Lonardi, S.: Visualizing and discovering non-trivial patterns in large time series databases. Inf. Vis. **4**(2), 61–82 (2005)
31. Lonardi, J., Patel, P.: Finding motifs in time series. In: Proceedings of the 2nd Workshop on Temporal Data Mining, pp. 53–68 (2002)
32. Martinez-Murcia, F.J., Ortiz, A., Gorriz, J.M., Ramirez, J., Castillo-Barnes, D., Salas-Gonzalez, D., Segovia, F.: Deep convolutional autoencoders vs PCA in a highly-unbalanced Parkinson's disease dataset: a datscan study. In: International Joint Conference SOCO'18-CISIS'18-ICEUTE'18, pp. 47–56 (2019)
33. McInnes, L., Healy, J., Melville, J.: UMAP: Uniform manifold approximation and projection for dimension reduction (2018). arXiv e-prints, page arXiv:1802.03426
34. Mohseni-Kabir, A., Wu, V., Chernova, S., Rich, C.: What's in a primitive? Identifying reusable motion trajectories in narrated demonstrations. In: 2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), pp. 267–272 (2016)
35. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)
36. Nhon, D.T., Anand, A., Wilkinson, L.: Timeseer: scagnostics for high-dimensional time series. IEEE Trans. Vis. Comput. Graph. **19**(3), 470–483 (2013)
37. Ordóñez, P., DesJardins, M., Feltes, C., Lehmann, C.U., Fackler, J.C.: Visualizing multivariate time series data to detect specific medical conditions. AMIA, pp. 530–534 (2008)
38. Perin, C., Vernier, F., Fekete, J.-D.; Interactive horizon graphs: Improving the compact visualization of multiple time series. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 3217–3226 (2013)
39. Rohlig, M., Luboschik, M., Schumann, H., Bögl, M., Alsallakh, B., Miksch, S.: Analyzing parameter influence on time-series segmentation and labeling. In: 2014 IEEE Conference on Visual Analytics Science and Technology (VAST), pp. 269–270 (2014)
40. Scherer, D., Müller, A., Behnke, S.: Evaluation of pooling operations in convolutional architectures for object recognition. In: Artificial Neural Networks—ICANN 2010, pp. 92–101 (2010)
41. Sedlmair, M., Munzner, T., Tory, M.: Empirical guidance on scatterplot and dimension reduction technique choices. IEEE Trans. Vis. Comput. Graph. **19**(12), 2634–2643 (2013)
42. Senin, P., Lin, J., Wang, X., Oates, T., Gandhi, S., Boedihardjo, A.P., Chen, C., Frankenstein, S., Lerner, M.: Grammarviz 2.0: a tool for grammar-based pattern discovery in time series. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 468–472 (2014)
43. Shepard, E.L., Wilson, R.P., Quintana, F., Laich, A.G., Liebsch, N., Albareda, D.A., Halsey, L.G., Gleiss, A., Morgan, D.T., Myers, A.E., et al.: Identification of animal movement patterns using tri-axial accelerometry. Endanger. Species Res. **10**, 47–60 (2008)
44. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: Proceedings 1996 IEEE Symposium on Visual Languages, pp. 336–343 (1996)
45. Singh, S., Zhang, S., Pruett, W.A., Hester, R.: Ensemble traces: interactive visualization of ensemble multivariate time series data. Electron. Imaging 1–9 (2016). https://doi.org/10.2352/ISSN.2470-1173.2016.1.VDA-505
46. Singhal, A., Seborg, D.E.: Clustering multivariate time-series data. J. Chemom. **19**(8), 427–438 (2005)

47. Swihart, B.J., Caffo, B., James, B.D., Strand, M., Schwartz, B.S., Punjabi, N.M.: Lasagna plots: a saucy alternative to spaghetti plots. Epidemiology **21**(5), 621–5 (2010)
48. van den Elzen, S., Holten, D., Blaas, J., van Wijk, J.J.: Reducing snapshots to points: a visual analytics approach to dynamic network exploration. IEEE Trans. Vis. Comput. Graph. **22**(1), 1–10 (2016)
49. van der Maaten, L., Hinton, G.E.: Visualizing data using t-sne. J. Mach. Learn. Res. **9**, 2579–2605 (2008)
50. van der Maaten, L., Postma, E.O., van den Herik, H.J.: Dimensionality reduction: a comparative review. J. Mach. Learn. Res. **10**(1–41), 66–71 (2009)
51. van Unen, V., Li, N., Molendijk, I., Temurhan, M., Höllt, T., van der Meulen-de Jong, A.E., Verspaget, H.W., Mearin, M.L., Mulder, C.J.J., van Bergen, J., Lelieveldt, B.P.F., Koning, F.: Mass cytometry of the human mucosal immune system identifies tissue- and disease-associated immune subsets. Immunity **44**(5), 1227–1239 (2016)
52. Walker, J.S., Borgo, R., Jones, M.W.: Timenotes: a study on effective chart visualization and interaction techniques for time-series data. IEEE Trans. Vis. Comput. Graph. **22**(1), 549–558 (2016)
53. Walker, J.S., Jones, M.W., Laramee, R.S., Bidder, O.R., Williams, H.J., Scott, R., Shepard, E.L.C., Wilson, R.P.: Timeclassifier: a visual analytic system for the classification of multi-dimensional time series data. Vis. Comput. **31**(6–8), 1067–1078 (2015)
54. Walker, J.S., Jones, M.W., Laramee, R.S., Holton, M.D., Shepard, E.L.C., Williams, H.J., Scantlebury, D.M., Marks, N.J., Magowan, E.A., Maguire, I.E., Bidder, O.R., Virgilio, A.D., Wilson, R.P.: Prying into the intimate secrets of animal lives; software beyond hardware for comprehensive annotation in daily diary tags. Mov. Ecol. **3**, 29 (2015)
55. Whited, L., Graham, D.: Abnormal respirations (2018). https://www.ncbi.nlm.nih.gov/books/NBK470309/. Accessed 9 Feb 2019
56. Wilson, W., Birkin, P., Aickelin, U.: Motif detection inspired by immune memory. In: Artificial Immune Systems, pp. 276–287 (2007)
57. Wilson, W., Birkin, P., Aickelin, U.: The motif tracking algorithm. Int. J. Autom. Comput. **5**(1), 32–44 (2008)
58. Xie, C., Xu, W., Mueller, K.: A visual analytics framework for the detection of anomalous call stack trees in high performance computing applications. IEEE Trans. Vis. Comput. Graph. **25**(1), 215–224 (2019)
59. Yang, K., Shahabi, C.: A PCA-based similarity measure for multivariate time series. In: Proceedings of the 2nd ACM International Workshop on Multimedia Databases, pp. 65–74 (2004)
60. Yang, K., Shahabi, C.: On the stationarity of multivariate time series for correlation-based data analysis. In: 5th IEEE International Conference on Data Mining (ICDM'05), pp. 805–808 (2005)
61. Yuan, G., Drost, N.A., McIvor, R.A.: Respiratory rate and breathing pattern. McMaster Univ. Med. J. **10**, 23–25 (2013)


**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Mohammed Ali** is a lecturer in the Department of Computer Science at King Khalid University, SA. He is currently a Ph.D. candidate in Computer Science at Swansea University in the Visual Computing Research group, under the supervision of Prof Mark Jones. His research interests include data mining and knowledge discovery, information visualization, machine learning, and visual analytics.

**Mark W. Jones** has received the B.Sc. and Ph.D. degrees from Swansea University. He is a Professor in the Department of Computer Science at Swansea University, where he leads the Visual Computing Research group. His research interests include global illumination, visualization, data science, and associated algorithms and data structures. http://cs.swan.ac.uk/~csmark/.



**Xianghua Xie** received the M.Sc. (with commendation) and Ph.D. degrees in computer science from the University of Bristol, Bristol, UK, in 2002 and 2006, respectively. He is a full Professor with the Department of Computer Science, Swansea University, Swansea, UK. He was a recipient of an RCUK academic fellowship, and he is currently leading the Computer Vision and Machine Leaning Lab at Swansea University. His research interests include various aspects of pattern recognition and machine intelligence and their applications to real-world problems. He has published more than 140 refereed conference and journal publications and (co-)edited several conference proceedings. He is an associate editor of a number of journals, including IET Computer Vision. He is a senior member of IEEE and a member of BMVA.

**Mark Williams** is a respiratory physiologist by training, Mark has been involved in clinical research and the development of medical technology for more than 35 years. Professor Williams began his research career at the University of Dundee, before moving to the Nuffield Department of Anaesthetics, University of Oxford, and finally taking up his current position in the Faculty of Life Sciences and Education at the University of South Wales. He is presently running clinical trials in cancer care and surgery, lymphoedema, respiratory disease and public health.