Knowledge-Based Systems 233 (2021) 107507

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Concurrent time-series selections using deep learning and dimension reduction

Mohammed Ali [a,b], Rita Borgo [c], Mark W. Jones [a,*]

[a] Swansea University, United Kingdom
[b] King Khalid University, Abha, Saudi Arabia
[c] Kings College London, United Kingdom

ARTICLE INFO

*Article history:*
Received 30 July 2020
Received in revised form 15 September 2021
Accepted 16 September 2021
Available online 22 September 2021

*Keywords:*
User interaction
User study
Dimension reduction
Time-series data
Deep Learning

ABSTRACT

The objective of this work was to investigate from a user perspective linkage between a 1D time-series view of data and a 2D representation provided by dimension reduction techniques. Our hypothesis is that when such interaction happens seamlessly, the use of these linked views, compared to only interacting with the 1D time-series view, for the ubiquitous task of selection and labelling, is more efficient and effective both in terms of performance and user experience. To this end we examine different dimension reduction techniques (UMAP, t-SNE, PCA and Autoencoder) and evaluate each technique within our experimental setting. Results demonstrate that there is a positive impact on speed and accuracy through augmenting 1D views with a dimension reduction 2D view when these views are linked and linkage is supported through coordinated interaction.



## 1. Introduction

User exploration of large time-series data to elicit understanding and interpretation of the fundamental behaviour of the condition under observation is a challenging task. Scientists working with large time-series data in different domains tend to exhibit similar patterns of interaction [1,2]. The first and crucial step is to achieve familiarity with the data. This includes both an overview of the whole data stream, as well as the need to compare details across long time-duration to gain understanding. As part of this interaction, there is a need to **identify**, **select** and **label** patterns within the data, which supports the extraction of patterns for further statistical analysis or as labelled data for machine learning. As understanding develops, the scientist will form hypotheses and rule extractions as they attempt to describe the phenomena. Such a working pattern arises, for example, during the development and deployment of new sensors, new applications of sensors or development of a new theory to explain the phenomenon under observation.

A specific example emerges from the Natural Sciences research domain where biologists utilize tri-axial accelerometers to tag animals and collect high frequency data about the behaviour of the animal under observation [3,4]. Tags are often experimental in nature especially during the development of the devices, and since they are difficult to attach to animals, the data collected can be very different even between animals of the same species. Also the behaviour captured can be very different inter-species. In parallel to tagging devices development and data collection, the biologists develop, benchmark and validate their theories of behaviour [5].

The above scenario is exemplar of challenges common to experimental research and beyond, and has provided us with a strong motivation to explore developments that may enable scientific users to fully immerse themselves in their time-series data effectively.

Interacting with 1D time-series is a well-studied area [6–11]. The overall objective to identify, select and label patterns is supported by low level mouse interactions of pattern selection via point and click and data brushing. The approaches mentioned provide various interfaces to enhance the efficiency of these interactions through the use of additional degrees of freedom such as panning, zooming, annotation and other enhanced visual cues. Across the three tasks labelling using 1D interaction can represent the most tedious and repetitive task. Commonly, patterns must still be individually selected to be labelled, while it would be desirable for all similar patterns to be selected in a parallel operation to speed-up user interaction and at the same time allow the user to fine tune selections both in parallel and individually.

Our starting point was whether we could create a tool to effectively inspect multivariate time-series data, that would support interaction as follows: A user could select a pattern and all similar patterns in parallel (Fig. 1(top), **concurrent selection**); Moving a brushed selection left or right (back and forth in time) in parallel moves other selections also left or right (Fig. 1(bottom), **tuning**). Fig. 1 shows a summary of our concept where the orange

* Corresponding author.
*E-mail address:* m.w.jones@swansea.ac.uk (M.W. Jones).

https://doi.org/10.1016/j.knosys.2021.107507

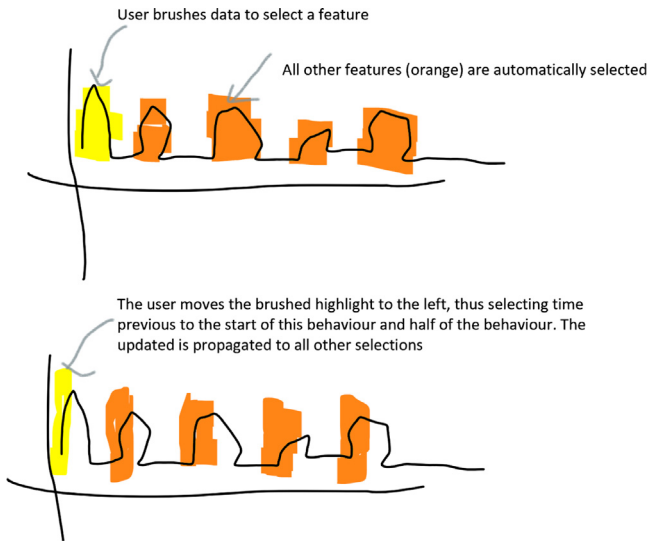


**Fig. 1.** Efficient interaction: Top: The concept of parallel selection. The user brushes a pattern (yellow) and similar patterns are automatically selected in parallel (orange). Bottom: The user slides the yellow selection to the left intending to capture the period before the pattern as well as half the pattern. This interaction is made automatically in parallel to the orange selections.

selections are all pattern based and selection is implemented using the sliding window method for time-series data.

The sliding window technique projects the data to higher dimensions (Fig. 2). Dimension reduction is a critical and effective method to express high dimensional data as a function of the most relevant features. We therefore investigate the possibility of implementing our proposed parallel interaction by linking 1D views with orthogonal 2D plots obtained by applying dimensionality reduction to the restructured data from the sliding window approach. Through the 1D to 2D linkage, feature exploration in the 1D view is supported and enhanced by the concise but highly expressive representation of the projected data. To achieve this, our proposed solution leverages and marries both similarity and dimension reduction. We treat a **window** of $n$ consecutive time-series data samples as points in $n$D space. If we **stride** one time sample to the right, each $n$D vector will be very similar to the previous and thus close in $n$D space. Patterns close in shape, but distant in time will also be close neighbours in $n$D space. As patterns increase in dissimilarity their distance in $n$D space will also increase. Distinct patterns will form separate clusters in high-dimension space. Paths between those clusters represent their temporal connection. A larger stride could be used to save computational time, but this was unnecessary and we use a stride of 1 throughout, which will capture the clusters. A stride length that is similar to or larger in size than the target patterns would reduce the ability of the method to resolve the clusters. We demonstrate that noisy features, where 30% noise is added, still project to similar positions, thus demonstrating this approach to dimension reduction is noise tolerant (Appendix B).

An effective dimension reduction technique will preserve clusters, prevent outliers and inliers (neighbourhood integrity), and preserve the temporal paths connecting clusters. By employing successful dimension reduction from high-dimension space to 2D we can provide an effective interface to satisfy our research goal. Neighbours in 2D space will correspond to similar patterns in the time-series data.

To maximize the expressiveness and clustering performance of the 2D view we leverage the power of deep learning within the dimensionality reduction pipeline. Deep learning techniques are relatively new and performant in the area of time-series analysis. From a data analysis and machine learning perspective it is common practice to use the sliding window approach (as described above) to capture patterns in a time-series as a set of feature points in high dimensional space. Neighbouring points in the high dimensional space will represent similar patterns. As a final step of machine learning, the post-processed (e.g., clustered) points are reduced to 2D using dimension reduction techniques for plotting. These static plots are coloured according to ground truth or the determined cluster membership, and thus the efficacy of the method can be qualitatively judged by spotting incorrectly located coloured points (coloured outliers). In the last decade t-SNE has been the favoured technique to perform dimensionality reduction, recent developments have also seen UMAP as a strong contender.

We are motivated by our example use case (biology sensors). We use accelerometry data from an Imperial Comorant [12,13]. Our overall research goal is two-fold: to introduce concurrent selection for time-series data exploration; and to integrate recent developments in machine learning and dimension reduction in the context of time-series data exploration and examine its effectiveness. The research questions we address are:

RQ1: Are state-of-the-art machine learning dimension reduction techniques useful for presenting an abstract 2D interface of time-series data to users?
RQ2: Do users understand an abstract 2D interface and relate it to the conventional 1D time-series representation?
RQ3: Do abstract 2D interfaces provide any benefits compared to only utilizing 1D time-series representations?
RQ4: Do different dimension reduction techniques lead to differences in either user perception of difficulty or user performance?

Our contributions are:

- A comparative review of the expressive power of dimension reduction techniques to create a 2D embedding of time-dependent data points which are the result of using the sliding window approach on time-series data (RQ1 and RQ4).
- The implementation of a novel interface integrating linkage between 1D views to 2D view of the 2D embedding of time-series data. (RQ2 and RQ3)
- The design of a user study to evaluate and validate the interface and 2D embeddings generated by different dimensionality reduction methods to assess their fitness in the context of projecting time-series data (RQ2 and RQ3).

These contributions demonstrate our objective to introduce effective concurrent selections for time-series data.

The rest of the paper is organized as follows. In Section 2 we present the related work. Section 3 presents the methodology behind the sliding window approach for multivariate time-series data and linked views. In Section 4 we report the design of the user study and its analysis. We report and discuss our findings in Section 5.

## 2. Related work

The most common form of representing time-series data is a line graph [11]. It effectively works when dealing with a small data space, but there comes a point when performing such tasks becomes more challenging due to many concurrent time series or large time series datasets [7]. Finding frequently occurring patterns or outlier patterns in large time series dataset is not easy. These tasks are quite distinct, and are usually tackled separately. Many works have been proposed to tackle these issues for example, VizTree [14] where frequently occurring patterns

are represented by the thickest branches and surprising patterns are the thinnest branches across the tree. Streamstory [15] which represents cyclical behaviour as states and transitions derived from a Markov Chain model to provide a multi-scale data representation. Streamstory removes the time dimension and cycles are dependent on clusters formed from similar data points. Dealing with such systems demands user domain knowledge to find interesting patterns. Other works such as TimeSearcher 2 [6] and TimeClassifier [16] are used for pattern discovery through query-by-example in the time-series data. Both systems require users having a general notion of what constitutes interesting patterns to enable detection and labelling of interesting regions. Other distance measures and metrics, such as Dynamic Time Warping (DTW) [17] and Longest Common Subsequence (LCS) [18] can have $O(N^2)$ complexity which TimeClassifier improved by implementing cross-correlation in the frequency domain for fast matching. It also included an interface to allow the user to remove incorrectly matched patterns. Both these problems have also been solved in DTW [19]. In the above works, the user needs to provide an example for matching. The approach we take here results in patterns clustered together, enabling similar features to be selected through a user interface and also allowing concurrent panning in the interface. Some works focus on the interaction techniques while analysis techniques have been given less attention such as pixelbased [9], lensing techniques [8,10], stackzoom and chronoLens [7].

Recently, projection-based methods have gained more attention in the visualization field. Based on the current state of the art, we categorize literature pertinent to our work into two categories: **projection-based methods** that are predominantly sourced from the visualization literature and **dimensionality reduction** techniques that are reported in the machine learning literature.

### 2.1. Projection-based methods (visualization)

With the growing importance of time series analysis, different visualization techniques have been proposed. The connected scatterplot technique has been used with temporal data that can be visually interesting and effective [20]. Haroz et al. [21] use the connected scatterplot to tackle multivariate data. The connected scatterplot is employed to visualize two related time series. The connection between points in a scatterplot is based on their temporal sequence. Despite its lack of familiarity and under-exploration in the visualization community, Haroz et al. [21] suggest the technique given its merit for presenting and communicating data. Sedlmair et al. [22] present a data study comparing class separability of dimension reduction data in 2D scatterplots, interactive 3D scatterplots and scatterplot matrices. They conclude that using 2D scatterplots to explore the output of different dimensionality reduction techniques is the most promising approach. Also, they advocate avoiding interactive 3D scatterplots for dimension reduction data, especially for cluster verification tasks.

Xie et al. [23] encourage using projection-based methods for two reasons. First, a scatterplot, as a visualization technique, is intuitive and easy to read. Second, scatterplots can provide a unified co-embedding space for visualizing data and their similarities and show the embedded semantic content. Bach et al. [24] introduce time curves to visualize patterns of evolution in temporal data where a timeline is folded into itself. Different events of an object are depicted by dots where they are connected following the chronological order of the events. In such a way, the similar time points end up being close to each other. Also, the characteristics of the temporal behaviour can be comprehended from the curve shape which is recognized as a single object. Similarly, Elzen et al. [25] propose a projection-based method to explore and analyse the change of dynamic networks by transforming each time-step network into a high-dimensional vector which is then projected onto a two-dimensional space using dimensionality reduction techniques. Thus, each point in the projection space refers to a snapshot of a time step which helps the user to identify the abnormal state, steady state, and recurring state of the dynamic network and the transition process.

Among multiple dimensions, patterns usually evolve over time in such data which are hard to be detected. Projection based methods leverage dimensionality reduction techniques to allow the user to analyse and visualize multivariate data, but dimensionality reduction techniques used alone do not provide means of exploring multivariate patterns over time [26]. Jackle et al. [26] use a sliding window approach which is the major difference to the methods of Bach et al. [24] and Elzen et al. [25]. Thus, dimensionality reduction is performed for each data window separately which are then plotted sequentially along the time axis obtaining the similarity across multiple time points evolving over time.

Ali et al. [27,28] use PCA to reduce multivariate time-series data to a 2D plot view of the entire data set in order to find repetitive and outlier patterns. They do not evaluate the use of such views. Similar to Ali et al. [27], Omata et al. [29] propose a method for analysing the time series data of unsteady flow fields with the major difference using a deep convolutional autoencoder (DCAE). Dimension reduction is performed and the spatial structure of the flow field at each time period is represented by low-dimensional features (latent space representation or bottleneck) using DCAE. After that, the conventional dimensionality reduction techniques such as PCA are performed on the features that are extracted from the bottleneck layer to visualize the trajectory of the flow in a 2D plot.

Guo et al. [30] integrate projection based methods within their EventViewer, a tool to support event sequence analysis. t-SNE is employed with the event overview module of the tool to provide projection of event vectors, to support users in comparison of how events co-occur within each sequence stage. LDSScanner [31] is an interesting and important tool for exploring structures within latent features and high dimensional space using t-SNE as 2D visualization. It is tested on low dimensional synthetic data and MNIST which is known to work well with t-SNE. We deal with dynamic data and our findings that t-SNE is difficult for users are supported by others which find that t-SNE is less able to preserve continuums [32–34]. Strobbelt et al. [35] propose a visual analytics tool to support "what if"-style exploration of trained sequence-to-sequence models across every stage of the translation process. States and neighbourhood are projected to lower dimensional space to simplify visual interpretation. Different to ours the tool aim is to identify which patterns have been learned and to validate the model.

Rauber et al. [36] investigate projection of time-dependent data, and propose an adaptation of t-SNE, referred to as dynamic t-SNE, that strives to strike a balance between temporal coherence and projection reliability.

### 2.2. Dimensionality reduction techniques

Dimensionality reduction techniques aim at representing high-dimensional data in low-dimensional spaces to facilitate visual interpretation and support analytical process. Many techniques exist, ranging from simple linear projections like principal component analysis (PCA) [37] and multidimensional scaling (MDS) [38] to more complex nonlinear transformations (NLDR) [39]. Modern NLDR techniques are sometimes referred to as manifold learning, most recent examples are t-Distributed Stochastic Neighbour

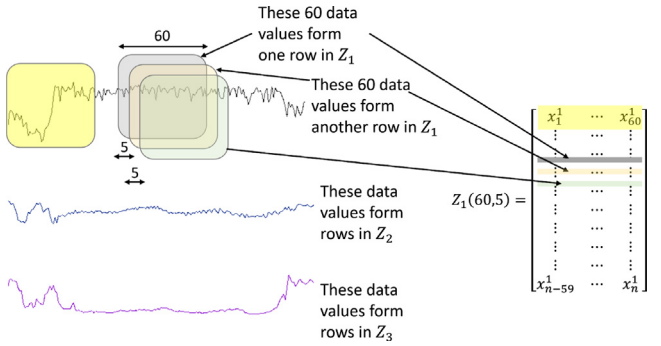


**Fig. 2.** Sliding window approach on multivariate time-series data with an example window width of 60 and stride of 5. $Z_1$ is populated from the first data channel. The entire time-series data is transformed to 3 × 60D lists of points on which dimension reduction is performed using a Deep Convolutional Autoencoder (DCAE) that learns a complex feature preserving dimension reduction. We derive a lower dimensional feature space from the original 180 ($3 \times 60$) dimensions using DCAE.

Embedding (t-SNE) [40] and Uniform Manifold Approximation and Projection (UMAP) [32]. To enhance the efficiency of extracting patterns in data, dimensionality reduction techniques are utilized [41]. Using dimension reduction in combination with further visual encodings that detect the internal state of the learning model improves the performance of visual-interactive labelling [42]. As a feature extraction method, Principal Component Analysis (PCA) is applied to time-series data [43–45]. PCA is the most used in the projection-based methods [25,27,29]. Some works such as [24,26,46] use multi-dimensional scaling (MDS) to embed time points in 2D space. If Euclidean distance is used, the results of classical MDS will yield as the same results as PCA [47]. t-Distributed Stochastic Neighbour Embedding (t-SNE) [40] is utilized helping to visualize high-dimensional data by giving each datapoint a location in a two or three-dimensional map. Omata et al. [29] and Elzen et al. [25] use non-linear dimensionality reductions (t-SNE). Omata et al. [29] mention some of the weaknesses of t-SNE where the trajectory changes greatly and its orbit has an irregular shape because of some discontinuity. Thus, the result of t-SNE is quite difficult to interpret. These points will be discussed later during this study.

One of the practical applications of autoencoders is dimensionality reduction for data visualization. It has a high ability to learn data projections that are more interesting than other basic techniques [48,49]. Huang et al. [50] 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. Therefore, Omata et al. [29] use the DCAE to extract important features which are then reduced to a 2D space assisting the use to visualize the spatio-temporal structure of an unsteady flow. In their work, there are no interaction techniques associated with the static images, which assist the user to explore and interact with raw and projected data. Compared to [46], we offer better separability of features by utilizing DCAE, and we investigate using an objective user study the usefulness of the abstract interface in achieve high F-score for selecting patterns or sequences from the data. DCAE also allows integrated working with multivariate time-series data compared to approaches that handle each channel separately.

## 3. Methodology

Despite the effectiveness of dimensionality reduction methods to support analysis of datasets with multiple layers of observations these methods have been mostly designed and evaluated in the context of static data. Few user studies have focused on their application in the context of multidimensional and multivariate temporal data. Vernier et al. [51] highlight the challenges involved in applying projection techniques to dynamic data and the difference in quality of results and stability of the methods compared to their application to static data. The authors evidence a gap in literature as well as potentials for improvements. Our research questions lie in the remit of these challenges — that is exploring the expressive power of projection techniques and how this can be leveraged effectively. To address these research questions we create a 2D view of the entire 1D time-series data through data transformation using the sliding window approach and subsequent dimension reduction to 2D. We create coordinated 1D and 2D views that enable users to brush selections in both views, thus supporting the investigative phase of gaining familiarization with data and labelling of data. The following sections give details of our data preparation.

### 3.1. Sliding window

We 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 $X_i = \{x_i^1, \ldots, x_i^d\}$ is the $d$-dimension vector of data recorded at time-step $i$. Let $w$ be the window width and $s$ the stride. (See Fig. 2 for an example where $w = 60$ and $s = 5$ and also the explanation in the primary submission video.)

Define a matrix $Z_k$ where each row is a vector (window) of size $w$ of data extracted from the $k$th dimension. Each row is a time slice from the original data. (See Fig. 2.)

$$Z_k(w,s) = \begin{bmatrix} x_1^k & x_2^k & \cdots & x_w^k \\ x_{1+s}^k & x_{2+s}^k & \cdots & x_{w+s}^k \\ \vdots & \vdots & \ddots & \vdots \\ x_{1+(r-1)s}^k & x_{2+(r-1)s}^k & \cdots & x_{w+(r-1)s}^k \end{bmatrix}$$

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

When more than one dimension of the multivariate data is used, $Z$ becomes a three-dimensional array.

### 3.2. Dimension reduction

The sliding window approach results in a set of $r$ points (rows from $Z$) in $w-$dimensional space (each row is a window of size $w$ on the data). We use dimension reduction techniques to transform these $w-$dimensional points to 2D. The set of 2D points then becomes our secondary view of the time-series data complementing the 1D time-series graph. Each point in the 2D view represents one $w-$dimensional point in $Z$ which represents $w$ contiguous time-steps (a window) in the original time-series data set. It is understood that if the dimension reduction preserves neighbourhood coherency of the original data, then close points in the 2D space will correspond to windows capturing similar patterns in the higher dimensional space. Thus, linking the two views will provide efficient parallel selection of patterns.

Dimension reduction techniques include linear and non-linear approaches [52]. We chose popular and state-of-the-art dimension reduction techniques: Deep Convolutional Autoencoder (DCAE), Principal Component Analysis (PCA), t-Distributed Stochastic Neighbour Embedding (t-SNE) [40] and Uniform Manifold Approximation and Projection (UMAP) [32].

Following the latest trends in machine learning we use DCAE to learn an effective dimension reduction to a set of latent features and reduce these further using PCA, t-SNE or UMAP. We considered using DCAE or PCA alone and discuss this at the end of Section 5. DCAE is a performant approach to dimension reduction providing a good separation of features in latent space. PCA, UMAP and t-SNE provide alternative pathways for reducing the latent space to 2D thus allowing us to evaluate different approaches to the 2D user interface.

We discuss the characteristics of the various approaches and their anticipated impact on user behaviour. Non-linear DR techniques are thought to help to avoid overcrowding issues [33] which is also a natural goal from a user perspective. While t-SNE is currently the most commonly used technique, the new UMAP algorithm shows high competitiveness compared to t-SNE [33]. 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 [32–34]. Therefore the proposed methodology (DCAE followed by one of the approaches) should provide good feature separation (provided by DCAE), avoid overcrowding (t-SNE and UMAP), better computation (PCA and UMAP), and meaningful clusters (UMAP). The relative behaviour of the methods and their impact on user performance is to be evaluated.

***Multivariate dimension reduction.*** Our approach to multivariate dimension reduction is to use autoencoder to learn a complex data projection [48]. Compared to the conventional autoencoder, Deep Convolutional Auto-Encoder (DCAE) has fewer parameters resulting in less training time. Also, DCAEs use local information to reconstruct the signal while conventional autoencoders 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. The two main parts are the encoding part that projects the data to a set of latent features and the decoding part that reconstructs the original data from the latent features. DCAE is a strong nonlinear dimensionality reduction method [53].

Dimensionality reduction for the 2D view is achieved by unsupervised training of an encoder and a decoder neural network, minimizing the reconstruction error (MSE) [53–55]. 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. Our user study examines the impact on usability of utilizing PCA, UMAP or t-SNE on the latent features from DCAE.

Table 1 shows input and output layer shape, filter size, number of kernels and activation functions used in our network. The dense layer is the latent space to which PCA, t-SNE or UMAP are applied to reduce to 2D. Above the dense layer is the encoder and below, the decoder. The table depicts a specific example for our tri-axial accelerometer data and a window width of $w = 60$ (other window widths are explored in the secondary video at 3m03s). Therefore, the input to the autoencoder is $60 \times 3$. The number of feature maps, size of filter and depth of the model are set based on the reconstruction error on a validation set. The first layer uses convolution of size $10 \times 3$ to learn 64 kernels on the tri-axial data thus lifting the original 180 dimensions to higher dimensions. Max pooling is used to down-sample the feature maps. Using max pooling has two main benefits: first it obtains translation-invariant features [56]. Second, it reduces the computational cost for the upper layer [50]. There are two further layers of convolution and max pooling before flattening to a dense layer. Decoding is the reverse of encoding, with upsampling replacing max pooling, where upsampling repeats each temporal data $n$ times along the time dimension (where $n = 3, 2, 2$ in the three upsampling stages). As activation function, a Rectified Linear Unit activation function (Relu) [57], defined as $ReLU(x) = max(0, x)$, 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.

**Table 1**
Architecture of the Deep Convolutional Auto-Encoder.

| Layers | Shape | Filter size | Number of kernels | Number of units | Activation |
|---|---|---|---|---|---|
| Input | 60x3 | | | | |
| Convolution | 60x64 | 10 | 64 | | ReLu |
| MaxPool | 30x64 | 2 | | | |
| Convolution | 30x32 | 5 | 32 | | ReLu |
| MaxPool | 15x32 | 2 | | | |
| Convolution | 15x12 | 5 | 12 | | ReLu |
| MaxPool | 5x12 | 3 | | | |
| Flatten | | | | | |
| **Dense** | | | | **60** | Linear |
| reshape | 5x12 | | | | |
| Convolution | 5x12 | 5 | 12 | | ReLu |
| Upsample | 15x12 | 3 | | | |
| Convolution | 15x32 | 5 | 32 | | ReLu |
| Upsample | 30x32 | 2 | | | |
| Convolution | 30x64 | 10 | 64 | | ReLu |
| Upsample | 60x64 | 2 | | | |
| Output | 60x3 | 10 | 3 | | Linear |

***Model training.*** The proposed model was implemented using the libraries TensorFlow [58] and Keras [48] and trained on the data to be reduced. Adam optimizer [59] 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 model learns an effective transformation from multivariate time-series data into latent feature representation by minimizing the reconstruction error. After that, the features in the latent space (bottleneck) are projected to 2D using PCA, t-SNE or UMAP which provide three distinct 2D views of the data whose effectiveness we can test in the user study. Source code for UMAP and t-SNE are provided in [32,40]. Both t-SNE and UMAP use as default the standard Euclidean distance between data points.

### 3.3. Linked views

The two views we employ in the user study are the 1D Graph Viewer and 2D Plot Viewer (Fig. 3). Views are linked such that a selection in either view highlights the corresponding data in the alternate view. Both views are interactive, allowing the user to zoom and pan. Selections are made using the direct manipulation metaphor of left click and mouse drag in both views. The 2D view additionally allows K nearest neighbours (Knn) search to the current mouse position in either 2D spatial location or 1D time. Fig. 3 shows Knn in 2D space. K can be varied using CTL-mousewheel. A KD-tree allows millions of points to be searched in real-time (see primary video at 9m30s).

Specifically, a single point in the 2D view will correspond to $w$ = *window size* contiguous data elements in the 1D view. A collection of points in the 2D view will correspond to multiple windows in the 1D view. For efficiency of rendering this collection is processed so any overlapping windows (in 1D) are merged to one selection.

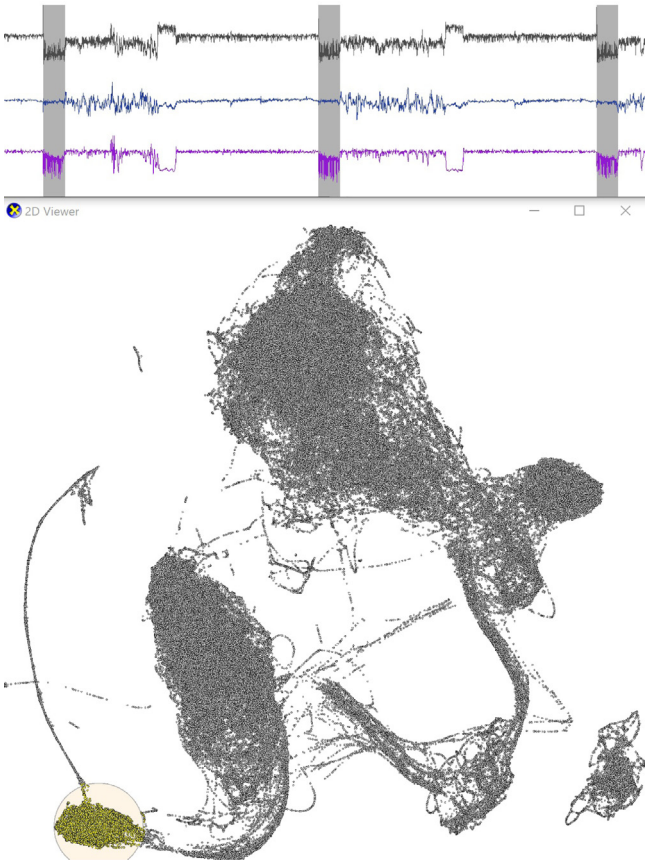


**Fig. 3.** Linked 1D and 2D views. The transformed time-series data interaction window shows the tri-axial accelerometer data lifted to $w$-dimensions using sliding window approach and reduced to 2D using autoencoder and UMAP. The mouseover event selects the k-nearest neighbours. Any selection in the 2D window selects the corresponding time-slices in the 1D view (in real-time) of the raw time-series data. Selection of proximate points in reduced 2D space results in selections of similar patterns in the 1D time-series. This linkage provides an effective interaction for data selection. The top window shows pattern one highlighted in grey (f2, as discussed in Section 4.2), as found by the k-nearest neighbours to the current mouse position (yellow).

The hypothesis behind the sliding window approach and rationale for linking the two views is that if the dimension reduction is expressive, points located spatially close to each other in 2D will correspond to signals (or patterns) of similar characteristics in the original time-series. Therefore, a selection of a cluster of points in 2D should result in many distinct selections in 1D of similar signals. We demonstrate this *qualitatively* in the accompanying videos and Fig. 3 demonstrates a selection made in 2D and the resulting multiple selections in 1D. It is demonstrated *quantitatively* through the successful minimization of the MSE metric during training, and the data collected during the user study and analysed in Section 4.6. The primary video describes the sliding window approach and demonstrates the exploration of the 2D view. The secondary video shows additional material (varying the window size, etc.) They show how the combination of dimension reduction on the sliding window approach for time-series provides an excellent basis for visual analytics exploration of long time-series data sets.

## 4. User study design

We identify **selection** and subsequent **labelling** as the most important task. Selection is the act of brushing data to indicate a pattern of user interest. Labelling is an extension of selection as the subsequent step to classify patterns. We provide two major motivations for identifying this important task. **Machine learning** requires extensive labelled data sets for training which demands huge resources to provide. Tools that allow efficient labelling of data are necessary to alleviate this bottleneck. Secondly, scientists need **effective user interfaces** in order to label time-series data as indicated by our use case introduced in Section 1.

### 4.1. Study views

In consequence of the two alternate views comprising of the raw 1D time-series data and 2D dimension reduced data we create four experimental conditions with respect to the views users can interact with during the study.

- **View 1** The user will interact with the raw time-series data using the primary 1D view window only.
- **View 2** The secondary view will be the 2D window using DCAE+PCA to embed the data to 2D. Autoencoder will learn a representation from $Z(w, s)$ (the sliding window representation of the original raw 3D time-series) and PCA will transform the dense layer to 2D.
- **View 3** The secondary view will use UMAP to replace PCA as used in View 2.
- **View 4** The secondary view will use t-SNE to replace PCA as used in View 2.

For an example of the user study interaction refer to the primary submission video at time 4:16.

### 4.2. Stimulus design

We define as **stimulus** a combination of target pattern and view. For the purpose of the study we select two patterns from the dataset (see Fig. 4, referred to as f1 and f2 respectively, with the following characteristics:

- each pattern is representative of a distinctive animal behaviour
- both patterns belong to the same activity e.g. the feeding process. This guarantees temporal continuity and context similarity
- patterns are equivalent from a behavioural point of view: pattern 1 (f1) corresponds to a cormorant surfacing pattern, pattern 2 (f2) to the corresponding diving pattern (complementary to f1). The data set labels are available online [60]
- each pattern has a distinctive visual signature, to ensure a clear perceptual separation between each other and every other pattern in the signal

The periodic nature of the behavioural pattern ensures that both patterns are present in the same quantity and equally distributed within the dataset.

### 4.3. Similarity labelling

The participant will use the view condition to enact a labelling of the data. The user will be shown a signal representing the pattern in the time-series data, and will be required to select all similar patterns by highlighting them. This simulates the process of locating and labelling all similar patterns in a data set. The labelled patterns would be used for statistical treatment and hypothesis forming by the research scientist, or could be used as a set of ground truth patterns for model testing or machine learning.

For the 1D view, this will require the user to move linearly through the data set selecting all similar patterns. The user must determine if the pattern is a target for selection, then enact that selection, and move to the next pattern by panning through the

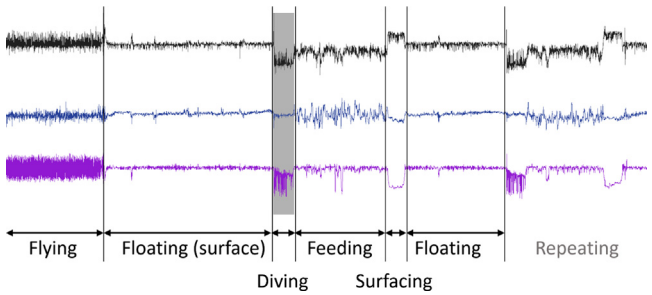


**Fig. 4.** A description of the patterns present within the data. The surface swimming, diving (f2), feeding, and surfacing (f1) behaviours repeat, interspersed with some flying (also see video). One diving pattern (f2) is currently highlighted.

data. The user will use an amount of time linear to the size of the data set to be inspected, plus time linear to the number of patterns selected. For the other views, it is hypothesized that the approach will allow similar patterns to be co-located in the 2D view, and thus tools that brush and select points local in the 2D view will locate many similar patterns simultaneously in the 1D view (Fig. 3). The sliding window approach should map windows with similar patterns to proximate points in the 2D view. The 2D view provides the capability to view the entire time-series in one view and perform labelling on multiple selections simultaneously.

### 4.4. Experimental procedure

We performed an initial pilot study to evaluate our experimental setting. Five participants were involved in the pilot, age range 26 to 40. Pilot feedback were used to tune the study procedure in terms of task duration and complexity, interaction and overall interface, and refine our hypothesis.

Our main study included a total of 8 views (4 views $\times$ 2 patterns). We randomize the view order for the stimuli per participant using random sequences generated from https://www.random.org/sequences. We consistently ask participants to undertake each of the four views on target pattern one (f1) in their random sequence order, followed by the four views in a different random sequence for target pattern two (f2). This allowed us to analyse performances on the two patterns separately as well as together. Our hypothesis was that performances for 1D view will demonstrate little variation in time between stimuli. Familiarity will not improve time, since the user must visit and label each pattern individually, patterns appear at similar frequency within the spectral signal, the same number of both patterns is present, both patterns are distinctive.

### 4.5. Participants

A total of 20 participants took part in the main study (15 male, 5 female). Participants were students and researchers at Swansea University. All have scientific backgrounds. Ages ranged from 20 to 48. All participants had normal or corrected to normal vision and were not informed about the purpose of the study prior to the beginning of the session. Due to the nature of task and stimuli we required participants to have familiarity with chart based visualization and interpretation, knowledge of terminology such as signal, pattern, cluster. We therefore selected participants with either technical or scientific background e.g. engineering, physical and mathematical sciences, computer science.

### 4.6. Study analysis

In this section we report data related to the main study.

Our main hypothesis is that users will be able to make more accurate selections in faster time. We therefore report and analyse accuracy using the well recognized Jaccard Index (an intersection over union similarity measure). We also provide precision, recall and F1-Score. True positives are where selections by the user match a ground truth selection (Jaccard index will be $> 0$ and $\leq 1$, with 1 indicating perfect match. A perfect match would be the selection matches the start and end time of the ground truth selection). False positives are user selections that do match any ground truth selections (Jaccard index would be zero). False negatives are ground truth selections that are unmatched by user selections. These results are presented in Tables 2–4.

In our analysis we also considered the effect of each visual encoding on the task of pattern search and selection for labelling purpose. We perform a first pass of analysis to look at effects of visual encoding on both patterns as a whole. In a second pass we look at effects of each pattern as a separate entity. We consider as independent variable the type of visual embedding in terms of: raw data as a 1D spectral signal representation, Principal Component Analysis (PCA), t-Distributed Stochastic Neighbour Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP). We consider as dependent variables the total number of patterns found (PF) and time required to complete the task (RT). Tables 5 and 6 respectively report descriptive statistics values for each embedding with patterns considered as a whole and individually.

A Shapiro–Wilk test was performed to check for data normality, when patterns were not considered individually results were consistently below the $\alpha = 0.05$ threshold, with values often as $\alpha < 0.001$, showing data significantly deviating from a normal distribution. When patterns were considered separately a subset of results showed a normal distribution: t-SNE embedding response time for f1 ($p = 0.970$), PCA embedding response time for f2 ($p = 0.114$). Detailed breakdown of results is provided in supplementary material.

To choose the most appropriate approach to test for significance we performed a Levene test to measure homogeneity of variance across embedding, test results revealed significance ($p < 0.01$), we explored data distribution in corresponding quantile–quantile (qq) plots, and given the sample size we opted to rely on non-parametric tests for significance analysis. Friedman's test with standard statistical level $\alpha = 0.05$ was therefore employed to determine the statistical significance between conditions, while Kendall coefficient of concordance $W$ was used to measure effect size.

Post-hoc analysis was conducted using Wilcoxon paired-samples test for all conditions that passed Friedman's test.

Tables 7 and 8 report $\chi^2$, significance values and effect size $W$ for patterns across embedding. Table 9 summarize significance values for post-hoc analysis for each embedding and patterns. For completeness we report a summary of the significance values for pairwise comparison per pattern and embedding with f1 results reported in Table 10 and f2 results in Table 11, for pairwise comparison of patterns for the same embedding in Table 12.

*Response time.* Performances with respect to response time across embedding and by patterns are summarized in Fig. 6. Tables 9–12 report results of analysis in function of patterns considered as a whole (f1 & f2) and separately. Analysis showed a significant difference across embedding (Tables 7 and 8). Post-hoc analysis across both patterns f1 and f2 (Table 9) revealed a strong effect of embedding with 1D slower overall and t-SNE significantly slower than PCA and UMAP (mean values reported in Table 5). Further analysis ((Tables 10 and 11)) confirmed a strong effect

**Table 2**
Data Analysis for pattern f1. The table shows the total number of labels/selections made during the study using the indicated method, and the precision, recall, F1-Score, and Jaccard Index mean for each embedding. The best results are highlighted with grey colour.

| Embedding | Selections Made | False Positives | True Positives | False Negatives | Precision | Recall | Overall F1 | Jaccard Mean |
|---|---|---|---|---|---|---|---|---|
| 1D | 1234 | 3 | 1231 | 50 | 0.997 | 0.961 | 0.979 | 0.757 |
| PCA | 1285 | 68 | 1217 | 3 | 0.947 | 0.997 | 0.972 | 0.837 |
| t-SNE | 2504 | 1193 | 1311 | 211 | 0.523 | 0.861 | 0.651 | 0.390 |
| UMAP | 1306 | 13 | 1293 | 6 | 0.99 | 0.995 | 0.993 | 0.829 |

**Table 3**
Data Analysis for pattern f2. The table shows the total number of labels/selections made during the study using the indicated method, and the precision, recall, F1-Score, and Jaccard Index mean for each embedding. The best results are highlighted with grey colour.

| Embedding | Selections Made | False Positives | True Positives | False Negatives | Precision | Recall | Overall F1 | Jaccard Mean |
|---|---|---|---|---|---|---|---|---|
| 1D | 1346 | 13 | 1333 | 18 | 0.99 | 0.987 | 0.988 | 0.734 |
| PCA | 1401 | 53 | 1348 | 7 | 0.962 | 0.995 | 0.978 | 0.878 |
| t-SNE | 1886 | 354 | 1532 | 81 | 0.812 | 0.950 | 0.876 | 0.66 |
| UMAP | 1355 | 23 | 1332 | 26 | 0.983 | 0.98 | 0.982 | 0.916 |

**Table 4**
Data Analysis for both patterns combined. The table shows the number of labels/selections made during the study using the indicated method, and the precision, recall, F1-Score, and Jaccard Index mean for each embedding. The best results are highlighted with grey colour.

| Embedding | Selections Made | False Positives | True Positives | False Negatives | Precision | Recall | Overall F1 | Jaccard Mean |
|---|---|---|---|---|---|---|---|---|
| 1D | 2580 | 16 | 2564 | 68 | 0.994 | 0.974 | 0.984 | 0.745 |
| PCA | 2686 | 121 | 2565 | 10 | 0.955 | 0.996 | 0.975 | 0.859 |
| t-SNE | 4390 | 1547 | 2843 | 292 | 0.648 | 0.907 | 0.756 | 0.526 |
| UMAP | 2661 | 36 | 2625 | 32 | 0.986 | 0.988 | 0.987 | 0.874 |

**Table 5**
Exploratory Data Analysis. Results for number of patterns found and response time for each embedding across both f1 and f2.

| Embedding | Pattern | Mean/Median/SD (PF) | Mean/Median/SD (RT/sec.) |
|---|---|---|---|
| 1D | f1 & f2 | 70.8/71.0/1.56 | 217.0/199.0/6.38 |
| PCA | f1 & f2 | 73.3/72.0/4.12 | 28.6/23.0/2.15 |
| t-SNE | f1 & f2 | 112.0/88.0/5.60 | 99.1/95.9/5.45 |
| UMAP | f1 & f2 | 71.9/71.5/2.32 | 31.3/27.0/2.27 |

**Table 6**
Exploratory Data Analysis. Results for number of patterns found and response time for each embedding by pattern f1 and f2.

| Embedding | Pattern | Mean/Median/SD (PF) | Mean/Median/SD (RT/sec.) |
|---|---|---|---|
| 1D | f1 | 70.6/71.0/1.31 | 221.0/200.0/6.17 |
| | f2 | 70.9/71.0/1.80 | 213.0/196.0/6.71 |
| PCA | f1 | 74.0/71.0/5.63 | 37.5/29.1/2.47 |
| | f2 | 72.6/72.0/1.43 | 19.7/18.0/1.32 |
| t-SNE | f1 | 130.0/126.0/5.9 | 136.0/131.0/4.58 |
| | f2 | 94.2/75.0/4.8 | 62.6/48.4/3.46 |
| UMAP | f1 | 72.4/72.0/3.02 | 39.3/30.8/2.73 |
| | f2 | 71.3/71.0/1.13 | 23.4/20.0/1.34 |

**Table 7**
Friedman results across embedding for both patterns f1 and f2. Effect size measured via Kendall's $W$.

| Factor | Pattern | $\chi^2(3)$ | Sig. | $W$ |
|---|---|---|---|---|
| Patterns Found | f1 & f2 | 18.989 | $p < 0.001$ | 0.36 |
| Response Time | f1 & f2 | 54.06 | $p < 0.001$ | 0.9 |

**Table 8**
Friedman results across embedding for each pattern individually. Effect size measured via Kendall's $W$.

| Factor | Pattern | $\chi^2(3)$ | Sig. | $W$ |
|---|---|---|---|---|
| Patterns Found | f1 | 18.989 | $p < 0.001$ | 0.31 |
| | f2 | 9.623 | $p = 0.022$ | 0.16 |
| Response Time | f1 | 52.86 | $p < 0.001$ | 0.89 |
| | f2 | 51.12 | $p < 0.001$ | 0.85 |

of both patterns, whenever significance was reached, with significant difference between t-SNE and other embedding ($p < 0.001$), no significant effect was measured between PCA and UMAP (mean values reported in Table 6). Pairwise comparison of pattern across the same embedding (Table 12) showed significant difference across all embedding except 1D. PCA, UMAP and t-SNE all performed faster with respect to selection of pattern f2, t-SNE however did not significantly improve in accuracy with respect to the number of patterns found suggesting a possible trade-off effect.

*Total patterns found.* Performances with respect to the total number of patterns found across embedding and by pattern are summarized in Fig. 5 and Fig. A.10 (supplemental material). Due to tSNE data being an outlier with respect to other embedding we have removed it from Fig. 5 to increase readability of the chart. We included all 4 embeddings in Fig. A.10 in supplemental material. Tables 9–12 report results of analysis in function of patterns considered as a whole (f1 & f2) and separately. Analysis showed a significant difference across embedding (Tables 7 and 8). Post-hoc analysis across both patterns f1 and f2 (Table 9) revealed a strong effect of embedding whenever significance was reached, with 1D and t-SNE being significantly different from all other embeddings ($p < 0.05$ or $p < 0.001$). Post-hoc analysis also revealed PCA being less accurate than UMAP ($p < 0.05$). Further analysis by pattern (Tables 10 and 11) revealed no effect of pattern f1 for t-SNE, with performances consistently lower than other embedding ($p < 0.001$). A strong effect of 1D was found across all embeddings for both f1 and f2 with performances consistently higher. A strong effect of f2 was identified with PCA performances improving against UMAP, no effect was measured on PCA with respect to f1. A strong effect of patterns f2 was identified with t-SNE performances improving against both PCA and 1D but not UMAP. Pairwise comparison of pattern across the same embedding (Table 12) showed a significant difference of f2 on UMAP with a large effect in increase in performances, and of

f2 on t-SNE with a moderate effect in increase in performances. Overall other embeddings performed equally on each pattern.

## 5. Findings and discussion

Data collected through observation, quantitative and qualitative evaluations provided interesting insights with respect to effectiveness of the use of 1D–2D linked views, their use to navigate data which are subjected to projection in lower dimensions and also behaviour of different dimensionality reduction techniques when dealing with continuous data with local connectivity and cyclic patterns.

***Jaccard index results***. The Jaccard Index results demonstrate that UMAP and PCA allow users to provide a much higher labelling quality with reference to the ground truth compared to the other approaches, even better than 1D which requires longer interaction. t-SNE suffers from large numbers of false positives. t-SNE presents clusters of similar points that can be selected, but these clusters seem to be fragmented or distributed randomly throughout the 2D embedding which results in difficulties locating the entire target pattern. It is known that t-SNE is less able to preserve continuums [32–34]. In contrast, both PCA and UMAP retain feature coherency in the 2D embedding which is seen in the accompanying video. As the user moves the lens across the embedding it is possible to follow paths in the embedding space that provide adjacency in the original signal space. Users detected these paths quickly and followed them to select the target pattern.

***Quantitative evaluation***. Quantitative data evaluation highlighted two important results. As initially hypothesized 1D spectral representation performance, in terms of response time and accuracy, did not change over time across different patterns. Pattern labelling in 1D translates to a linear search task across a list. With a total number of 71 patterns (71 f1 and 71 f2) present in the data 1D and UMAP were the closest in terms of overall accuracy. PCA, UMAP and 1D provided an effective interface for achieving accurate selections with PCA and UMAP enabling a far faster mode of working through parallel selections. Examining the video recordings, the inaccuracies are very minor. Some users appeared to overlook one pattern in 1D through scrolling past it. For PCA and UMAP, users select a region in 2D corresponding to parallel selections in 1D. While favouring a more natural behaviour this implies that users finish selections when it seems all patterns are selected. On close inspection of the video and data we found the errors are where a selection has missed a midpoint of a pattern and thus the pattern is split into (and counted as) two parts (two true positives). This error can be avoided by performing a check for continuity in temporal space and proximity to selection in 2D space to aid user selection, which is a common practice for intersection over union tests in pixel-based CNNs. The errors experienced by t-SNE are more serious and are a result of the selection area including neighbouring patterns not part of the target expected to be selected. This behaviour is very prominent in t-SNE where large chunks of non relevant patterns end up being part of a selection. A significant difference was registered between tasks involving f1 and tasks involving f2 for speed. A reason for performance improvement can be increased user familiarity with the interface, dataset and task. Improvement however did not register any significant changes across embedding with PCA and UMAP still outperforming. The decrease in response time for t-SNE was not accompanied by a significant increase in its accuracy, confirming what we measured for f1 in terms of difficulty in precisely selecting patterns. We would like to mention that these results are not meant to put t-SNE effectiveness under discussion — we are instead interested in the fact that UMAP and PCA afford users a more intuitive interface for pattern selection with this type of data. Our analysis showed that UMAP seems to yield representations that are as meaningful as t-SNE, while preserving more of the global structure as well as continuity of behaviours that are inherently linked. Similar results have been recently reported in other fields where phenomena under analysis follow tightly linked evolution patterns with local connectivity [33].

***Qualitative and observational feedback***. In our evaluation we were also interested in observing user behaviour while interacting with linked views and dimensionally reduced representations of the data. All participants where provided with training on how to use the interface and were given time to familiarize with both interface and task. During execution of the experiment we screen captured their interaction and an experimenter also acted as silent observer collecting notes. At the end of the study a debriefing session was used to collect participants feedback. Analysis of our observations provided the following evidences:

- User expectation is to assume that if part of a pattern overlapped with the selection window, then trialling movement in various directions would bring further parts of the pattern into selection and remove undesirable signals. Whilst this occurs with UMAP and PCA, it is not the case for t-SNE. This provided some confusion for participants, which led to descriptions of t-SNE as being quite confusing or not as intuitive at the PCA and UMAP views.
- All users appreciate the significant difference in speed between using the 2D views of UMAP and PCA compared to 1D labelling and how such an approach (linking the 2D view to 1D) would greatly enhance and accelerate accurate labelling of data.

**Take-Home message:** Duality of information, even across spaces with different dimensionality, increases accuracy as well as response time. From a cognitive point of view co-location of semantically linked patterns is an expectation. Projection techniques that favour grouping based on both saliency and semantics not only favour performances but also comprehension [61].

Focusing on interaction behaviour we can identify the following emerging pattern, consistent and reinforced by results reported in the quantitative analysis:

- When dealing with 1D view participants adopted the expected linear scanning behaviour, scrolling back and forth along the spectral diagram. Given the uniform distribution of patterns the behaviour did not favour selection of either pattern, as also reported by the statistical analysis.
- When dealing with dimensionally reduced data users begin scanning from top to bottom in the 2D window which would have benefited t-SNE since the main clusters were positioned towards the top of the window, compared to PCA (at the sides) and UMAP at the bottom. For t-SNE some users chanced on the correct cluster without needing to search the window. Some users also chanced on the correct windows for PCA and UMAP, but with fewer occurrences of such cases.

**Take-Home message:** Projection techniques favour spatial organization of patterns, this in turn supports the natural top-down processing employed in real world visual search which allows more efficiently identification of very complex targets than those represented in a pattern or conjunction search task [62].

***Timing***. The training of the DCAE in this paper takes 41 min on the data set of 173,256 points. Each of the dimension reduction techniques are executed on the DCAE latent features with PCA taking 1 s, UMAP 3 min 36 s and t-SNE taking 2 h 25 min 56 s.

**Table 9**
Pairwise comparison for each embedding across features f1 and f2 ($p$-value $p$ and effect size $r$).

| Patterns Found (#) and Response Time (RT) - f1 & f2 | | | | |
|---|---|---|---|---|
| Embedding | # and RT | PCA | t-SNE | UMAP |
| 1D | # | $p < 0.001/r = 0.64$ | $p < 0.001/r = 0.61$ | $p = 0.02/r = 0.44$ |
| | RT | $p < 0.001/r = 0.88$ | $p < 0.001/r = 0.88$ | $p < 0.001/r = 0.88$ |
| PCA | # | – | $p < 0.001/r = 0.51$ | $p = 0.012/r = 0.26$ |
| | RT | – | $p < 0.001/r = 0.88$ | $p = 0.728/r = 0.08$ |
| t-SNE | # | – | – | $p < 0.001/r = 0.60$ |
| | RT | – | – | $p < 0.001/r = 0.88$ |

**Table 10**
Pairwise comparison for each embedding for pattern f1 ($p$-value $p$ and effect size $r$).

| Patterns Found (#) and Response Time (RT) - f1 | | | | |
|---|---|---|---|---|
| Embedding | # and RT | PCA | t-SNE | UMAP |
| 1D | # | $p = 0.0028/r = 0.69$ | $p < 0.001/r = 0.74$ | $p < 0.05/r = 0.67$ |
| | RT | $p < 0.001/r = 0.88$ | $p < 0.001/r = 0.85$ | $p < 0.001/r = 0.88$ |
| PCA | # | – | $p < 0.001/r = 0.70$ | $p = 0.95/r = 0.004$ |
| | RT | – | $p < 0.001/r = 0.88$ | $p = 0.893/r = 0.03$ |
| t-SNE | # | – | – | $p < 0.001/r = 0.73$ |
| | RT | – | – | $p < 0.001/r = 0.88$ |

**Table 11**
Pairwise comparison for each embedding for pattern f2 ($p$-value $p$ and effect size $r$).

| Patterns Found (#) and Response Time (RT) - f2 | | | | |
|---|---|---|---|---|
| Embedding | # and RT | PCA | t-SNE | UMAP |
| 1D | # | $p < 0.05/r = 0.56$ | $p < 0.05/r = 0.48$ | $p = 0.59/r = 0.18$ |
| | RT | $p < 0.001/r = 0.88$ | $p < 0.001/r = 0.88$ | $p < 0.001/r = 0.88$ |
| PCA | # | – | $p = 0.178/r = 0.31$ | $p < 0.001/r = 0.63$ |
| | RT | – | $p < 0.001/r = 0.84$ | $p < 0.132/r = 0.34$ |
| t-SNE | # | – | – | $p < 0.034/r = 0.46$ |
| | RT | – | – | $p < 0.001/r = 0.84$ |

**Table 12**
Pairwise comparison of number of patterns found for each embedding between patterns f1 and f2.

| Patterns Found (#) and Response Time (RT) - f1 vs f2 ($p$-value $p$ and effect size $r$). | | | | | |
|---|---|---|---|---|---|
| Embedding | #/RT | 1D | PCA | t-SNE | UMAP |
| 1D | # | $p = 0.88/r = 0.018$ | – | – | – |
| | RT | $p = 0.388/r = 0.2$ | – | – | – |
| PCA | # | – | $p = 0.749/r = 0.075$ | – | – |
| | RT | – | $p < 0.001/r = 0.73$ | – | – |
| t-SNE | # | – | – | $p = 0.039/r = 0.46$ | – |
| | RT | – | – | $p < 0.001/r = 0.83$ | – |
| UMAP | # | – | – | – | $p = 0.019/r = 0.55$ |
| | RT | – | – | – | $p < 0.003/r = 0.63$ |

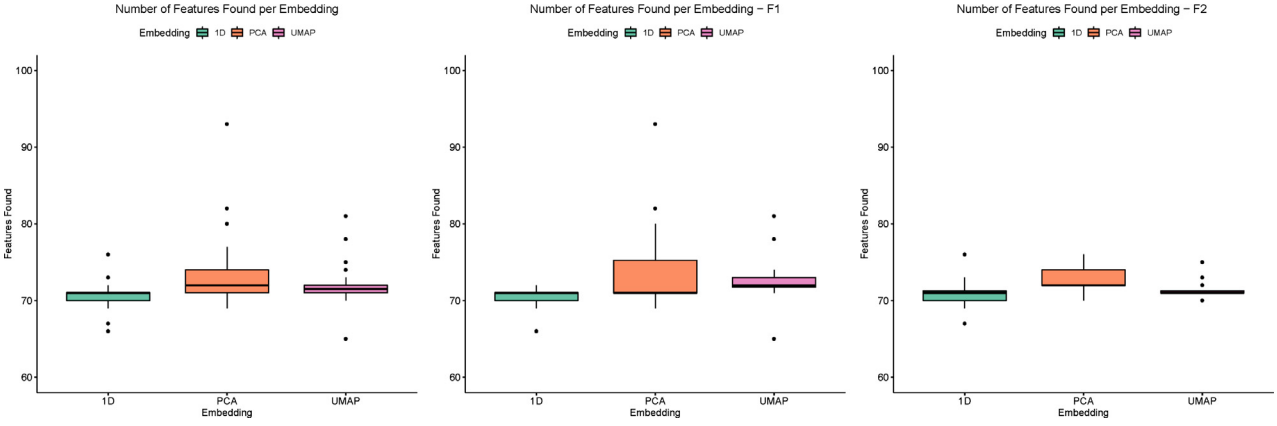


**Fig. 5.** Number of selected patterns for 1D, PCA and UMAP embeddings across all patterns (tSNE not included to improve readability).

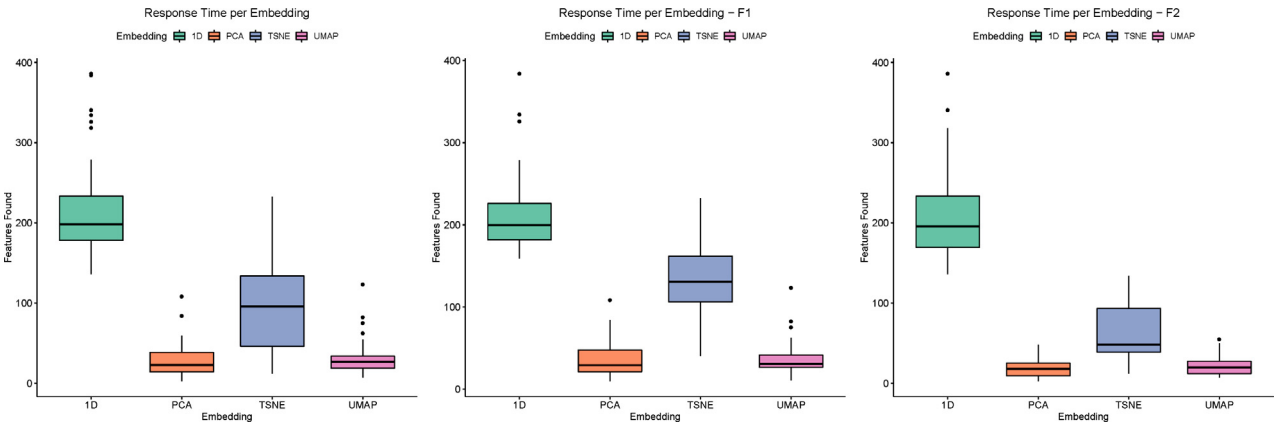


**Fig. 6.** Response time (RT/sec.) by embedding across all patterns.

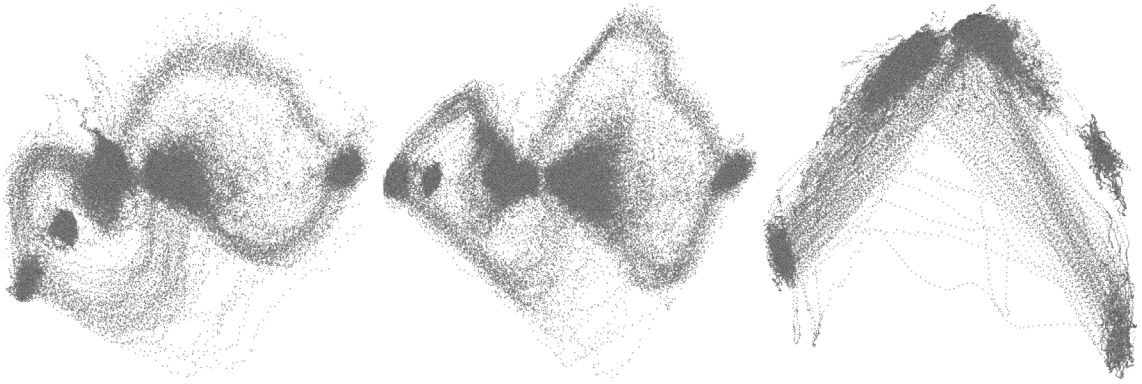

**Fig. 7.** Left: DCAE learnt 2D embedding. Note, this image has been rotated by 90° to visually match other views. Middle: DCAE with PCA. Right: PCA only.

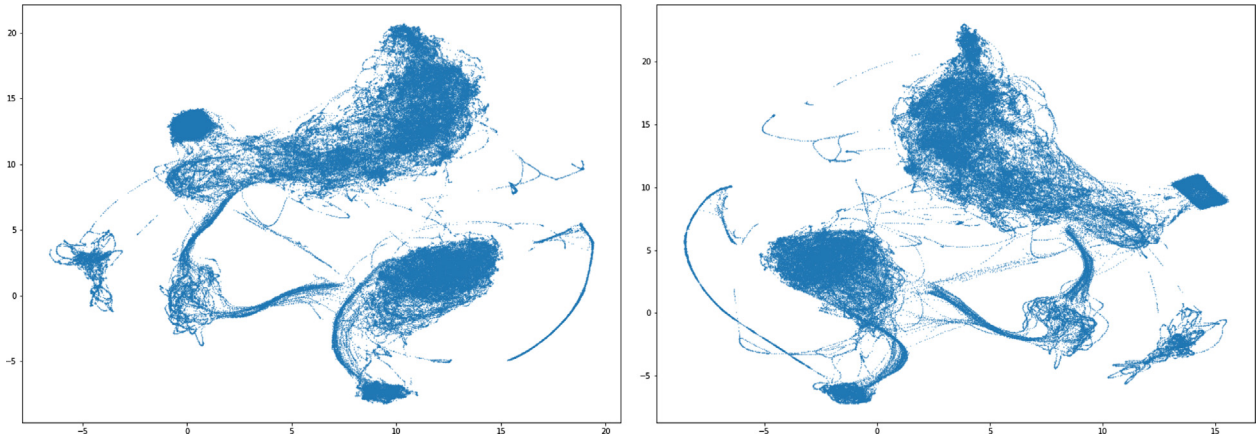

**Fig. 8.** Left: Using 20 latent features in the dense layer (to replace **60** in Table 1). Right: Using 120.

***DCAE or PCA alone***. DCAE lifts the sliding window matrix to high dimension using 64 $10 \times 3$ kernels before compressing down to 60 features (see Table 1) resulting in an expressive dimension reduction of the original data that separates and preserves features. UMAP, PCA and t-SNE provide three alternative paths from 60 latent features to 2D to provide the user interface. We investigated which of these is most suited to user input. We conclude PCA and UMAP are the most efficient and should be used for interfaces where dimension reduction is involved. The fact that users were able to intuitively explore pattern similarity across multiple time instances using PCA and UMAP is also strongly indicative that DCAE accurately preserves features in the latent space.

We also tested DCAE alone to learn the 2D embedding (Fig. 7 (Left)). Like other researchers we find that DCAE seems to learn an embedding similar to PCA (DCAE alone to learn a 2D embedding also takes 41 min on this data). Since our user study concludes that PCA provides a good interface for this task, using DCAE alone would provide a good solution. But UMAP is also competitive, and some runs of DCAE+UMAP seemed to be very effective in segmenting the data, so there may be a compelling reason to use DCAE+UMAP rather than DCAE+PCA (Fig. 7 (Middle)) or DCAE alone (Fig. 7 (Left)).

We also used PCA alone (Fig. 7 (Right)) on the original data (after the sliding window approach). Although PCA also shows 5 clear clusters, the far right cluster is part of the surfacing

signal. The top middle cluster is a merge of surface swimming and flying (see video). In the case of using DCAE first, there is a successful segmentation of surface swimming and flying (which also visually have very different signals). It is recognized that DCAE is able to learn a more complex data projection than PCA directly [48,49]. This provides support for employing DCAE for the initial compression before using PCA, UMAP or t-SNE to provide the 2D interface.

***Latent features experiments***. Following on from the experiment of using PCA alone we also present the results of varying the number of latent features in the dense layer of the architecture (see the 60 in bold in Table 1). Our primary goal was that we wanted the dense layer to be expressive, leading to low loss. Then employing the three experimental conditions to test their suitability for constructing user interfaces. We experimented with dense layers of 20 (Fig. 8 (left)) and 120 (Fig. 8 (right)). Both produce interfaces similar to 60 latent features (in this case with UMAP providing the further dimension reduction from 20 and 120 respectively to 2D). This is very useful as it suggests that the method is not overly sensitive to parameter choice.

***Filter size experiments***. Regarding the size of the receptive field, i.e., how large the field is when you map back to the original data, with 10-5-5 (see Table 1), our receptive field is 10-10-20 respectively (maxpool of 2). We favoured a slow increase in the receptive field rather than have dramatic increases (e.g., versus 5-5-10 with a receptive field of 5-10-40). The search space for the most effective filter size is large. Here we include images of the interface we obtain when we replace the filter sizes with 3-3-3 (Fig. 9 (left)), 5-5-10 (Fig. 9 (middle)), and 8-6-4 (Fig. 9 (right)) respectively. These produce interfaces very similar to the one used in the user study. It suggests that the method is quite robust to filter size choice with no suggested preference between gradual or larger increases in the receptive field. For this aspect the user study results would be valid over a wide range of parameter choices.

***Reproducibility***. DCAE has a random initialization which leads to different latent features for input into PCA, UMAP and t-SNE for their respective 2D interfaces. t-SNE also introduces inconsistency. We executed the DCAE part of the methodology several times and provide a comparison in the secondary submission video. Indeed, the 2D plots can change significantly between runs, but there is consistency in the number of clusters and the signals they represent within the data. Both PCA and UMAP remain effective in providing meaningful clusters to this task. t-SNE, in each view, requires several clusters to be identified and selected to isolate the patterns. Different views could impact on the time for participants to conduct the user study in the following way. For 1D, there is no difference since this operates on the raw data. For PCA and UMAP, we notice that the clusters still remain well defined, and would therefore not expect the task time to vary significantly. Additionally for UMAP, some views spread the clusters further apart. For these views we would predict that the time for UMAP would reduce, but we did not encounter a view where we would expect it to take significantly longer. For t-SNE the signals seemed to remain split at spatially distant locations, and therefore we conclude it would not get significantly faster depending on the input from DCAE. This method has also been validated successfully on data collected from 9 triaxial accelerometers monitoring human behaviour (27D vector at each time-stamp) and tidal breathing data (1D flow data at each time-stamp).

## 6. Conclusions

Labelling of patterns in large time-series data from biology sensors is a challenging task. Most of data pre-labelling, to be able to generate enough samples to use automatic or supervised methods, is still done by hand. The limited control in how sensors can be positioned and/or kept in place together with unexpected changes in animal behaviour in reaction to changes in the surrounding environment, causes data collected from each sampled subject to be different. To ease this inevitable search-and-label step we have investigated the effects of linking traditional multi-dimensional spectral representation with 2D reduced views of the same data. We implemented a linked coordinated 2D view with a 1D view. We provided a mechanism to make selections in either 1D or 2D and see the corresponding selections in the alternate view. To test the efficacy of the 2D view we turned off the 1D $\longrightarrow$ 2D link to focus the study on the 2D $\longrightarrow$ 1D link. Our hypothesis that similar patterns map spatially close in the 2D view using dimension reduction techniques is validated qualitatively and quantitatively. PCA and UMAP provide intuitive linked interactions which is demonstrated by the ability for participants to complete the pattern selection task significantly faster than utilizing the 1D view only or the coordinated 1D and 2D view where the 2D view interface is provided by t-SNE. We also experiment with using DCAE or PCA to directly generate the 2D view. To strengthen our analysis we have recorded participants interactions and also observed their behaviour real-time. Recordings and observations were used to further validate our findings. Linking proved to be an effective means to traverse and label the data. Some of the 2D reduction techniques we employed performed considerably better than others, this result has spawned additional research worth further investigation. We also suggest that the employed approach to find closely related patterns could be compared to approaches based on alternative distance measures [63,64].

## CRediT authorship contribution statement

**Mohammed Ali:** Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing. **Rita Borgo:** Conceptualization, Methodology, Formal analysis, Investigation, Writing. **Mark W. Jones:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing, Supervision, Project administration.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

This work was supported by EPSRC, UK, grant number EP/N028139/1.

## Appendix A. Supplemental material - statistical analysis

Embeddings: Raw Data (1D), Principal Component Analysis (PCA), t-Distributed Stochastic Neighbour Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP) (see Tables A.13–A.16).

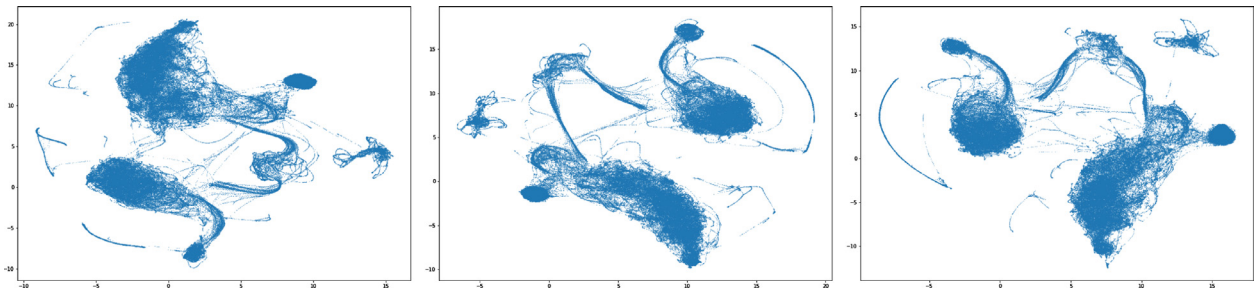

**Fig. 9.** Left: Using filter sizes of 3,3,3 (to replace sizes 10,5,5 in Table 1). Middle: 5,5,10. Right: 8,6,4.

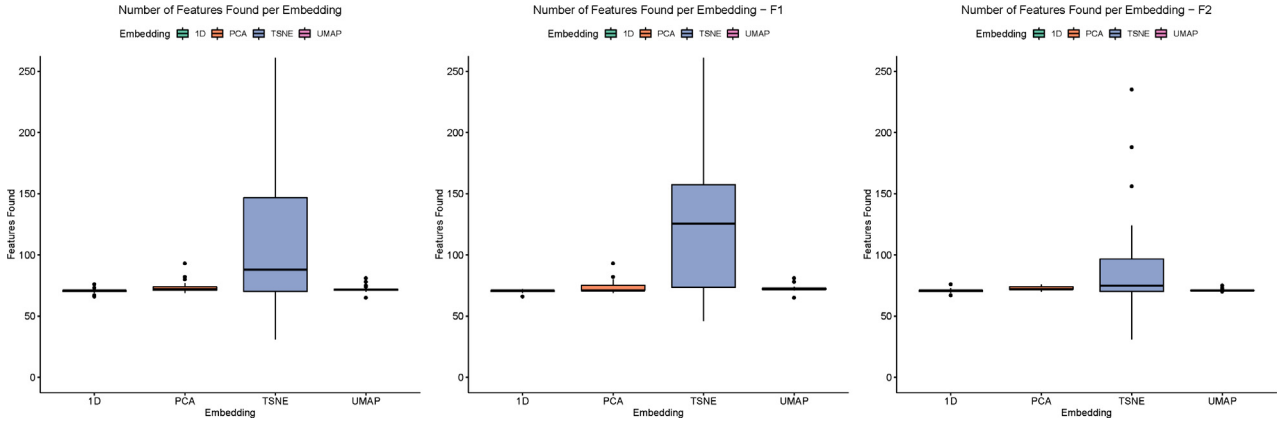


**Fig. A.10.** Number of selected patterns by embedding, including t-SNE, across all patterns.

**Table A.13**
Shapiro–Wilk test of normality. Results for number of patterns found for each embedding across patterns f1 and f2.

| Embedding | Pattern | Sig. (PF) |
|---|---|---|
| 1D | f1 & f2 | $p < 0.001$ |
| PCA | f1 & f2 | $p < 0.001$ |
| t-SNE | f1 & f2 | $p < 0.001$ |
| UMAP | f1 & f2 | $p < 0.001$ |

**Table A.14**
Shapiro–Wilk test of normality. Results for response time for each embedding across patterns f1 and f2.

| Embedding | Pattern | Sig. (RT) |
|---|---|---|
| 1D | f1 & f2 | $p < 0.001$ |
| PCA | f1 & f2 | $p < 0.001$ |
| t-SNE | f1 & f2 | $p = 0.012$ |
| UMAP | f1 & f2 | $p < 0.001$ |

**Table A.15**
Shapiro–Wilk test of normality. Results for number of patterns found for each embedding and patterns f1 and f2.

| Embedding | Pattern | Sig. (PF) |
|---|---|---|
| 1D | f1 | $p < 0.001$ |
| | f2 | $p = 0.02$ |
| PCA | f1 | $p < 0.001$ |
| | f2 | $p = 0.015$ |
| t-SNE | f1 | $p = 0.048$ |
| | f2 | $p < 0.001$ |
| UMAP | f1 | $p < 0.001$ |
| | f2 | $p < 0.001$ |

**Table A.16**
Shapiro–Wilk test of normality. Results for response time for each embedding and patterns f1 and f2.

| Embedding | Pattern | Sig. (RT) |
|---|---|---|
| 1D | f1 | $p < 0.001$ |
| | f2 | $p = 0.009$ |
| PCA | f1 | $p = 0.006$ |
| | f2 | $p = 0.114$ |
| t-SNE | f1 | $p = 0.970$ |
| | f2 | $p = 0.053$ |
| UMAP | f1 | $p < 0.001$ |
| | f2 | $p = 0.008$ |

## Appendix B. Noise analysis

Here we present the results of an experiment to determine whether non-linear dimension reduction is sufficiently tolerant to noise to still enable an effective interface. The original data consists of tri-axial accelerometer data. Within the ground truth labelled regions, we set 10% of the values to zero (i.e., (0, 0, 0)). e.g., if a sequence representing the surfacing behaviour is 180 samples in duration, then 18 of those samples would be set to (0, 0, 0). This noisy data is projected to 60 latent features using our DCAE model and projected down to 2D using UMAP. The user interface generated from the 2D projection is presented, and the user has selected the surfacing behaviour by brushing the points in the same cluster as presented by the interface generated from the data without noise (Fig. B.11). Fig. B.12 indicates that the method is tolerant of this noise. The same experiment was run with 30% noise with similar results (Fig. B.13).

## Appendix C. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.knosys.2021.107507.

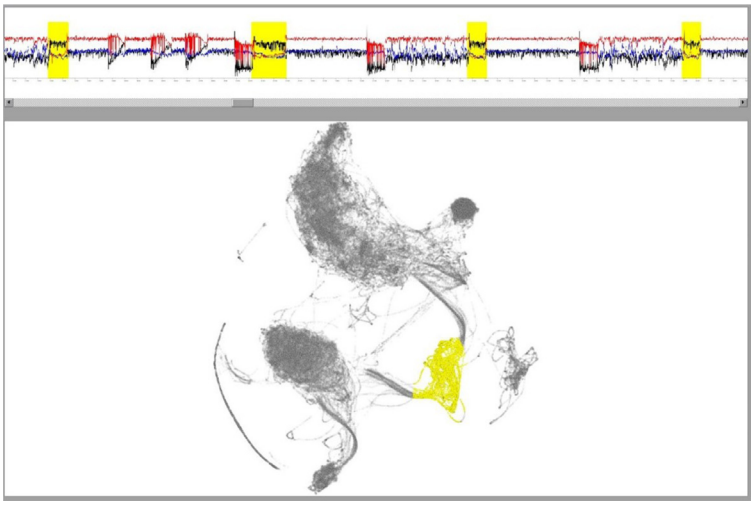

**Fig. B.11.** Pattern f2 is selected using a 2D projection of the original data set (with no noise).

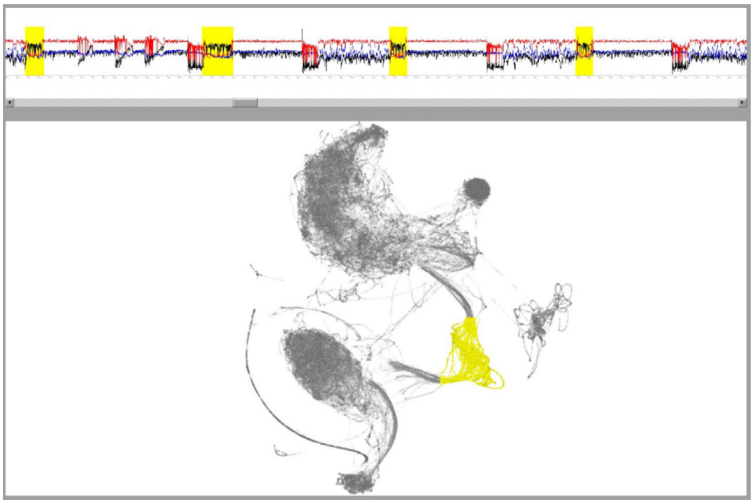

**Fig. B.12.** Pattern f2 is selected using a 2D projection of the data set after 10% of data values within f2 are set to zero.

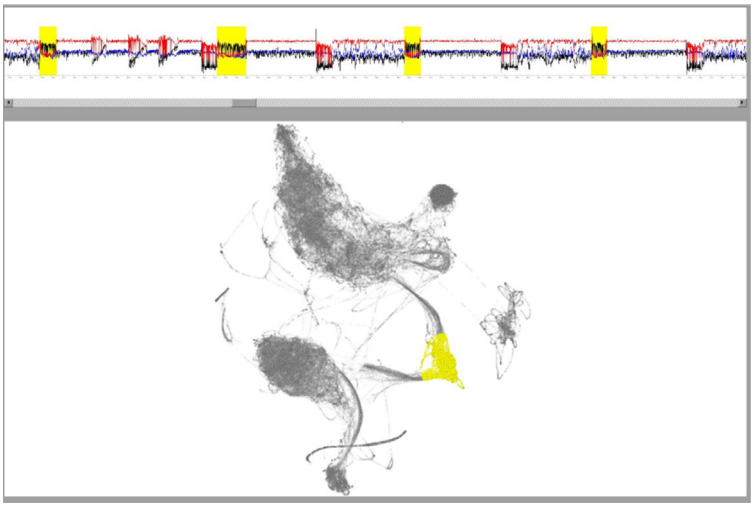

**Fig. B.13.** Pattern f2 is selected using a 2D projection of the data set after 30% of data values within f2 are set to zero.

## References


[1] R. Amar, J. Eagan, J. Stasko, Low-level components of analytic activity in information visualization, in: IEEE Symposium on Information Visualization, 2005. INFOVIS 2005, 2005, pp. 111–117, http://dx.doi.org/10.1109/INFOVIS.2005.24.

[2] M. Ali, A. Alqahtani, M.W. Jones, X. Xie, Clustering and classification for time series data in visual analytics: A survey, IEEE Access 7 (1) (2019) 181314–181338, http://dx.doi.org/10.1109/ACCESS.2019.2958551.

[3] R. Wilson, A. Neate, M. Holton, E. Shepard, D. Scantlebury, S. Lambertucci, A. di Virgilio, E. Crooks, C. Mulvenna, N. Marks, Luck in food finding affects individual performance and population trajectories, Curr. Biol. 28 (3) (2018) 3871–3877, http://dx.doi.org/10.1016/j.cub.2018.10.034.

[4] T. van Walsum, A. Perna, C. M. Bishop, C. Murn, P. Collins, R. Wilson, L. Halsey, Exploring the relation between flapping behaviour and accelerometer signal during ascending flight, and a new approach to calibration, Ibis 162 (2) (2019) 13–26, http://dx.doi.org/10.1111/ibi.12710.

[5] D.E. Cade, K.R. Barr, J. Calambokidis, A.S. Friedlaender, J.A. Goldbogen, Determining forward speed from accelerometer jiggle in aquatic environments, J. Exp. Biol. (2017).

[6] P. Buono, A. Aris, C. Plaisant, A. Khella, B. Shneiderman, Interactive pattern search in time series, Proc. SPIE 5669 (2005).

[7] J.S. Walker, R. Borgo, M.W. Jones, Timenotes: A study on effective chart visualization and interaction techniques for time-series data, IEEE Trans. Vis. Comput. Graphics 22 (1) (2016) 549–558.

[8] R. Kincaid, Signallens: Focus+context applied to electronic time series, IEEE Trans. Vis. Comput. Graphics 16 (6) (2010) 900–907.

[9] R. Kincaid, H. Lam, Line graph explorer: scalable display of line graphs using focus+context, in: Proceedings of the Working Conference on Advanced Visual Interfaces, 2006, pp. 404–411.

[10] J. Zhao, F. Chevalier, E. Pietriga, R. Balakrishnan, Exploratory analysis of time-series with ChronoLenses, IEEE Trans. Vis. Comput. Graphics 17 (12) (2011) 2422–2431.

[11] W. Aigner, S. Miksch, H. Schumann, C. Tominski, Visualization of Time-Oriented Data, Springer, 2011.

[12] R. Wilson, F. Quintana, A. Gómez-Laich, Accelerometry data for an Imperial Cormorant, Zenodo, 2021, http://dx.doi.org/10.5281/zenodo.5500402.

[13] E. Shepard, R. Wilson, F. Quintana, A.G. Laich, N. Liebsch, D. Albareda, L. Halsey, A. Gleiss, D. Morgan, A. Myers, C. Newman, D. McDonald, Identification of animal movement patterns using tri-axial accelerometry, Endanger. Species Res. 10 (2008) 47–60, http://dx.doi.org/10.3354/esr00084.

[14] J. Lin, E.J. Keogh, S. Lonardi, Visualizing and discovering non-trivial patterns in large time series databases, Inf. Visual. 4 (2) (2005) 61–82.

[15] L. Stopar, P. Skraba, M. Grobelnik, D. Mladenic, Streamstory: Exploring multivariate time series on multiple scales, IEEE Trans. Vis. Comput. Graphics 25 (4) (2019) 1788–1802.

[16] J.S. Walker, M.W. Jones, R.S. Laramee, O.R. Bidder, H.J. Williams, R. Scott, E.L.C. Shepard, R.P. Wilson, Timeclassifier: a visual analytic system for the classification of multi-dimensional time series data, Vis. Comput. 31 (6–8) (2015) 1067–1078.

[17] E. Keogh, C.A. Ratanamahatana, Exact indexing of dynamic time warping, Knowl. Inf. Syst. 7 (3) (2005) 358–386, http://dx.doi.org/10.1007/s10115-004-0154-9.

[18] Abdulla-Al-Maruf, H.-H. Huang, K. Kawagoe, Time series classification method based on longest common subsequence and textual approximation, in: Seventh International Conference on Digital Information Management (ICDIM 2012), 2012, pp. 130–137, http://dx.doi.org/10.1109/ICDIM.2012.6360087.

[19] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh, Searching and mining trillions of time series subsequences under dynamic time warping, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD'12, ACM Press, 2012, http://dx.doi.org/10.1145/2339530.2339576.

[20] R. Kosara, Presentation-oriented visualization techniques, IEEE Comput. Graph. Appl. 36 (1) (2016) 80–85.

[21] S. Haroz, R. Kosara, S.L. Franconeri, The connected scatterplot for presenting paired time series, IEEE Trans. Vis. Comput. Graphics 22 (9) (2016) 2174–2186.

[22] M. Sedlmair, T. Munzner, M. Tory, Empirical guidance on scatterplot and dimension reduction technique choices, IEEE Trans. Vis. Comput. Graphics 19 (12) (2013) 2634–2643.

[23] X. Xie, X. Cai, J. Zhou, N. Cao, Y. Wu, A semantic-based method for visualizing large image collections, IEEE Trans. Vis. Comput. Graphics (2018).

[24] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, P. Dragicevic, Time curves: Folding time to visualize patterns of temporal evolution in data, IEEE Trans. Vis. Comput. Graphics 22 (1) (2016) 559–568.

[25] S. van den Elzen, D. Holten, J. Blaas, J.J. van Wijk, Reducing snapshots to points: A visual analytics approach to dynamic network exploration, IEEE Trans. Vis. Comput. Graphics 22 (1) (2016) 1–10.

[26] D. Jäckle, F. Fischer, T. Schreck, D.A. Keim, Temporal MDS plots for analysis of multivariate data, IEEE Trans. Vis. Comput. Graphics 22 (1) (2016) 141–150.

[27] M. Ali, M. Jones, X. Xie, M. Williams, Towards visual exploration of large temporal datasets, in: 2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA), 2018, pp. 1–9.

[28] M. Ali, M.W. Jones, X. Xie, M. Williams, Timecluster: Dimension reduction applied to temporal data for visual analytics, Vis. Comput. 35 (6) (2019) 1013–1026, http://dx.doi.org/10.1007/s00371-019-01673-y.

[29] N. Omata, S. Shirayama, A novel method of low-dimensional representation for temporal behavior of flow fields using deep autoencoder, AIP Adv. 9 (1) (2019) 015006, http://dx.doi.org/10.1063/1.5067313.

[30] S. Guo, Z. Jin, D. Gotz, F. Du, H. Zha, N. Cao, Visual progression analysis of event sequence data, IEEE Trans. Vis. Comput. Graphics 25 (1) (2019) 417–426, http://dx.doi.org/10.1109/TVCG.2018.2864885.

[31] J. Xia, F. Ye, W. Chen, Y. Wang, W. Chen, Y. Ma, A.K.H. Tung, Ldsscanner: Exploratory analysis of low-dimensional structures in high-dimensional datasets, IEEE Trans. Vis. Comput. Graphics 24 (1) (2018) 236–245.

[32] L. McInnes, J. Healy, J. Melville, UMAP: Uniform manifold approximation and projection for dimension reduction, 2018, ArXiv e-Prints arXiv:1802.03426.

[33] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I.W.H. Kwok, L.G. Ng, F. Ginhoux, E.W. Newell, Dimensionality reduction for visualizing single-cell data using UMAP, Nature Biotechnol. 37 (2019) 38–44.

[34] V. van Unen, N. Li, I. Molendijk, M. Temurhan, T. Höllt, A.E. van der Meulen-de Jong, H.W. Verspaget, M.L. Mearin, C.J.J. Mulder, J. van Bergen, B.P. Lelieveldt, F. Koning, Mass cytometry of the human mucosal immune system identifies tissue- and disease-associated immune subsets, Immunity 44 (5) (2016) 1227–1239.

[35] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, A.M. Rush, Seq2seq-vis: A visual debugging tool for sequence-to-sequence models, IEEE Trans. Vis. Comput. Graphics 25 (1) (2019) 353–363, http://dx.doi.org/10.1109/TVCG.2018.2865044.

[36] P.E. Rauber, A.X. Falcão, A.C. Telea, Visualizing time-dependent data using dynamic t-SNE, in: Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers, in: EuroVis '16, Eurographics Association, 2016, pp. 73–77, http://dx.doi.org/10.2312/eurovisshort.20161164.

[37] I.T. Jolliffe, Principal Component Analysis, Springer Verlag, 1986.

[38] W.S. Torgerson, Multidimensional scaling: I. Theory and method, Psychometrika 17 (1952) 401–419.

[39] J.A. Lee, M. Verleysen, Nonlinear Dimensionality Reduction, first ed., Springer Publishing Company, 2007, Incorporated.

[40] L. van der Maaten, G.E. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (2008) 2579–2605.

[41] J. Lonardi, P. Patel, Finding motifs in time series, in: Proc. of the 2nd Workshop on Temporal Data Mining, 2002, pp. 53–68.

[42] J. Bernard, M. Hutter, M. Zeppelzauer, D.W. Fellner, M. Sedlmair, Comparing visual-interactive labeling with active learning: An experimental study, IEEE Trans. Vis. Comput. Graphics (2018) 298–308.

[43] R.H. Lesch, Y. Caillé, D. Lowe, Component analysis in financial time series, in: Computational Intelligence for Financial Engineering, 1999.(CIFEr) Proceedings of the IEEE/IAFE 1999 Conference on, 1999, pp. 183–190.

[44] K. Yang, C. Shahabi, A PCA-based similarity measure for multivariate time series, in: Proceedings of the 2Nd ACM International Workshop on Multimedia Databases, 2004, pp. 65–74.

[45] K. Yang, C. Shahabi, On the stationarity of multivariate time series for correlation-based data analysis, in: Fifth IEEE International Conference on Data Mining (ICDM'05), 2005, pp. 805–808.

[46] C.-C.M. Yeh, H. Van Herle, E. Keogh, Matrix profile III: The matrix profile allows visualization of salient subsequences in massive time series, in: 2016 IEEE 16th International Conference on Data Mining (ICDM), 2016, pp. 579–588, http://dx.doi.org/10.1109/ICDM.2016.0069.

[47] T.F. Cox, M.A.A. Cox, Multidimensional Scaling, Chapman and hall/CRC, 2000.

[48] F. Chollet, et al., Keras, 2015, https://keras.io (last accessed 2020-07-14).

[49] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science (2006) 504–507.

[50] H. Huang, X. Hu, Y. Zhao, M. Makkie, Q. Dong, S. Zhao, L. Guo, T. Liu, Modeling task fMRI data via deep convolutional autoencoder, IEEE Trans. Med. Imaging 37 (7) (2018) 1551–1561.

[51] E.F. Vernier, R. Garcia, I.P. da Silva, J.L.D. Comba, A.C. Telea, Quantitative evaluation of time-dependent multidimensional projection techniques, 2020, http://arxiv.org/abs/2002.07481.

[52] L. Van Der Maaten, E. Postma, J. Van den Herik, Dimensionality reduction: a comparative, J. Mach. Learn. Res. (2009) 66–71.

[53] C.M. Cheung, P. Goyal, V.K. Prasanna, A.S. Tehrani, OReONet: Deep convolutional network for oil reservoir optimization, in: 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 1277–1282.

[54] X. Guo, X. Liu, E. Zhu, J. Yin, Deep clustering with convolutional autoencoders, in: Neural Information Processing, 2017, pp. 373–382.

[55] F.J. Martinez-Murcia, A. Ortiz, J.M. Gorriz, J. Ramirez, D. Castillo-Barnes, D. Salas-Gonzalez, F. Segovia, 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, 2019, pp. 47–56.
[56] D. Scherer, A. Müller, S. Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, in: Artificial Neural Networks – ICANN 2010, 2010, pp. 92–101.
[57] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.
[58] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: A system for large-scale machine learning, in: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), 2016, 265–283.
[59] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, CoRR http://arxiv.org/abs/1412.6980.
[60] M. Jones, M. Ali, R. Borgo, Data for Concurrent Time-Series Selections Using Deep Learning and Dimension Reduction, Zenodo, 2021, http://dx.doi.org/10.5281/zenodo.5503031.
[61] M. Montefinese, E. Ambrosini, B. Fairfield, N. Mammarella, Semantic significance: A new measure of feature salience, Memory Cogn. 42 (2013).
[62] M. Ashcraft, G. Radvansky, Cognition, sixth ed., Pearson, 2014.
[63] J. Lines, A. Bagnall, Time series classification with ensembles of elastic distance measures, Data Min. Knowl. Discov. 29 (3) (2014) 565–592, http://dx.doi.org/10.1007/s10618-014-0361-2.
[64] A. Bagnall, J. Lines, A. Bostrom, J. Large, E. Keogh, The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances, Data Min. Knowl. Discov. 31 (3) (2016) 606–660, http://dx.doi.org/10.1007/s10618-016-0483-9.