

TimeClassifier

A Visual Analytic System for the Classification of Multi-Dimensional Time-Series Data

James S. Walker · Mark W. Jones · Robert S. Laramee · Owen R. Bidder · Hannah J. Williams · Rebecca Scott · Emily L. C. Shepard · Rory P. Wilson

Abstract Biologists studying animals in their natural environment are increasingly using sensors such as accelerometers in animal-attached ‘smart’ tags because it is widely acknowledged that this approach can enhance the understanding of ecological and behavioural processes. The potential of such tags is tempered by the difficulty of extracting animal behaviour from the sensors which is currently primarily dependent on the manual inspection of multiple time-series graphs. This is time-consuming and error-prone for the domain expert and is now the limiting factor for realising the value of tags in this area. We introduce TimeClassifier, a visual analytic system for the classification of time-series data for movement ecologists. We deploy our system with biologists and report two real-world case studies of its use.

Keywords Visual analytics · Time series analysis · Movement ecology

1 Introduction

The development of animal-attached ‘smart’ tags have revolutionized biologists understanding of the ecology of wild animals [28]. While this might seem empowering, the reality of researchers faced with perhaps 10 channels of data recorded at sub-second rates, is that it is time consuming to manually decode. There is currently no effective protocol for the derivation of animal behaviour from these tags [3].

James S. Walker · Mark W. Jones · Robert S. Laramee
Department of Computer Science, Swansea University, Swansea,
SA2 8PP, UK
E-mail: {csjames, m.w.jones, r.s.laramee} @swansea.ac.uk

Owen R. Bidder · Hannah J. Williams · Emily L.C. Shepard · Rory P. Wilson
Swansea Lab for Animal Movement, Biosciences, College of Science,
Swansea University, Swansea
SA2 8PP, UK.
E-mail: {367097, 798684, e.l.c.shepard, r.p.wilson} @swansea.ac.uk

The goal of this work is to provide a visual analytic system which assists in the labelling and understanding of animal behaviour.

The output of our work is a two year long collaboration between visualization researchers and movement ecologists. Researchers typically manually inspect several time-series graphs for known wave forms corresponding to behaviour. A typical deployment spans anywhere from two days and up to a week. Two days of data (48 hours) recorded at a frequency of 40Hz logs 6,912,000 measurements. Given such a volume of data, analysis can typically take several hours for the domain expert to undertake. Machine learning algorithms have been considered, but have been difficult to introduce because of the large numbers of training sets required and their low discriminating precision in practise.

In this paper we present TimeClassifier, a visual analytics system for the semi-automatic classification of animal behaviour, combining the search of large time-series data with the semi-automatic classification of events using user-defined templates. Our system requires one instance of behaviour for the matching process to take place. We utilize visualization to search for matching behaviour classifications and interaction to facilitate a user in the loop approach for the checking, rejecting and accepting of results to maintain a high accuracy. Our system provides biologists with a working solution which they have confidence in, and can analyse large complex data sets in a shorter time-frame.

Our work consists of the following contributions:

1. A generalisation of the associated tasks towards classification applied to time-series data.
2. Fast normalized cross-correlation for pattern matching which executes on large data sets in real time to provide an interactive application.
3. TimeClassifier a system that classifies large data with good precision and recall.

2 Related Work

We discuss two themes of related work. Firstly, existing methods for visualizing smart tag data, specifically the tri-axial accelerometer channels to discover visual human-oriented methods for the identification and analysis of behavioural patterns. Secondly, we discuss visual methods which assist in pattern discovery in time-series data.

2.1 Visualization of Tri-Axial Data

Current literature focuses on enhancing the exploration of tri-axial data to assist in finding interesting features and patterns. Spherical coordinates has been applied to show common behavioural cycles [14] and extended to show them in relation to other sensor channels by combining with parallel coordinates [37]. Further attributes have been derived to abstract from the raw sensor values, including: animal movement via reconstructed pseudo-tracks using a dead reckoning approach [38], and labelled behaviour to show higher-order state transitions between behaviours [5]. Unlike previous work which assist in understanding and exploring sensor data, we introduce a system for the analysis of animal behaviour through classification.

2.2 Visual Pattern Discovery in Time-series

The visual discovery of patterns in time-series data is an established research area in information visualization [26]. Lensing techniques have been widely applied to provide overlaid data transformations [44] and time axis deformations [21] to obtain details-on-demand. Alternative spatial layouts can reveal periodic structure [40] and accentuate interesting data regions [15]. Visualisations have been applied to the result of pattern discovery algorithms, including: wavelet analysis [18], cluster analysis across of multiple time scales [41], motif discovery with augmented suffix trees [25], tabular views [6] and frequently occurring patterns using k-means clustering [16].

We find no work on the interactive classification of time-series data. However, interactive classification has been considered by Elzen et al. [9] who facilitate interactive construction and analysis of decision trees for multi-dimensional data. More relevant to this work is the information retrieval of time-series data. Buono et al. [7] present TimeSearcher 2 for finding similar occurrences of selected regions in a time-series by interactively modifying tolerance levels. Holz et al. [17] search for specified patterns in time-series by defining similarity prior to search through a single gesture interaction. Gregory and Shneiderman [13] identify an array of shapes and the attributes by which time series can be identified, compared, and ranked. QuerySketch [31] introduces pattern definition from scratch. The link between how the signal relates to animal movement is complex, it is therefore not trivial to specify expected variances and shapes in the signal in our domain.

We incorporate visualisation, matching, and human interaction into one system for the classification of time-series

data and show it is effective through our expert case studies and comparison with machine learning techniques.

3 Tasks and Design

In this section we present the domain level tasks and design of our system. The project has been an iterative development with our collaborators from Swansea University's Swansea Laboratory for Animal Movement (SLAM) research centre.

3.1 Data

Smart tags [43], are autonomous logging devices which record parameters such as acceleration, magnetic field intensity, pressure, light intensity and temperature [30]. These devices acquire large quantities of high quality quantitative data from free-living animals which can be used to derive, and quantify, animal behaviour. For example, animal activity can be accessed using data from tri-axial accelerometers recording at high (infra-second e.g. 40 Hz) rates because specific behaviours are identified by animal posture (or attitude) and changes in body velocity, both of which are derived from accelerometers. Various authors have presented derived channels which highlight many specific features of sensor data which indicate behaviour including posture [27], signal dynamism [39], repetition in patterns [33] and specificity in rates of change of particular signals [32]. Our system provides a user oriented visual interface for deriving animal behaviour from these tags which can perform classification on any given channel (including those derived).

3.2 Problem Statement

Machine learning concerns a class of algorithms which derive training from data to discover previously unknown properties [10]. The learning aspect typically can be split into two categories, supervised, and unsupervised learning. Supervised techniques build a model from labelled data which generates predictions in response to new data. Traditionally, K-nearest neighbour (K-NN) [3], support vector machines (SVM) [11], and random forests [8] have all been applied to accelerometry data. Unsupervised learning algorithms deal with unlabelled data to find natural groupings of data. The accuracy is measured using precision and recall.

Classification is approached in the data mining community by having extensively labelled data demonstrating positive and negative instances of the template signal. Obtaining this data is time consuming, requires domain expertise, and the undertaking of field studies to gather video synchronised data. It is not possible to obtain large quantities of such data due to logistical and environmental constraints. Secondly, choosing the classification algorithm and parameters introduces its own class of problems. Typically, in this process, the data dimensionality is reduced to a few parameters which contain the relevant information to perform classification; feature extraction. Good classification results rely

heavily on the features chosen, however, extracting a desirable feature set is considered more of an art than a science and takes a great amount of skill along with trial and error [36]. Once the data is classified, if the precision and recall are less than desired, decisions must be made as to whether it is useful to invest more time creating additional training input, modify the parameters, or use a different learning algorithm. It is not obvious what the next best step to take is without expert knowledge of the underlying algorithms. Our system exploits and incorporates the knowledge of the domain expert to guide the classification process.

3.3 Domain Characterization

The domain requirements and uses of labelling animal data was discussed and refined through several informal project trials with our end users.

Domain Requirements -To be most useful to the movement ecology community, software to help in the analysis of smart tag-acquired data needs to be able to deal with large quantities of data to identify and classify behaviours quickly to a high accuracy. Validating results is essential to be able to see what was classified, along with the ability to manually accept and reject results via applied domain knowledge. Extensively large collections of labelled data do not exist in the marine wildlife domain so any tool must be able to operate effectively on a low number of provided instances of a behaviour.

Domain Uses -Two measures are used as a proxy for VO2 (oxygen consumption): Overall Dynamic Body Acceleration (ODBA) [12] and Vectorial Dynamic Body Acceleration (VeDBA) [4]. VeDBA can be used for dead reckoning to determine animal position and movement (thus removing reliance on battery demanding GPS and/or being available for aquatic or subterranean animals). Correct and accurate usage requires the identification of behavior, for instance so that energetic stationary behavior like scratching is not interpreted as progressive motion.

3.4 Tasks

We give a generalized classification of the tasks associated with classification and apply them to the context of time-series data.

Identify: Find data subsets which correspond to the specified behaviours in the time-series. *Identify(t)* returns all occurrences of a template (t) throughout the data series. The result is a set of unlabelled subsets from the time-series which are of the same classification as the input sequence t .

Associate: Classify behaviour to a specified group. *Associate(us, s)* associates an unlabelled subset of data (us) to a specified class (s). This is used to manually classify data, and for accepting found behaviours from an *Identify* to the correct group.

Reject: Remove a previous result or classified instance from the system. *Reject(ls)* rejects a labelled subset (ls) from the current group it is classified in. *Reject(us)* removes an unlabelled subset (us) from the result set of found behaviours (from an *Identify*).

Move: Relocates a classified instance to a specified classification group. *Move(ls, s)* moves a labelled subset (ls) to the classification group (s). This is equal to applying a *(reject(ls))* and an *(associate(ls, s))* operation.

Membership: Inspects which classification group a subset belongs to. *Membership(ls)* returns the classification group s_i the labelled subset (ls) belongs to.

Compare: Compare signals between classifications and over them. *Compare(a, b)* enables two data subsets a and b to be compared for similarity. *Compare(a, s_i)* allows the comparison of subset a to the subsets contained in the specified classification group s_i .

3.5 Algorithm

We consider classification as an extended form of search in the time domain. For example, given a set of labelled data, classification operates by searching for matches for each labelled item and assigns the results accordingly to the corresponding classification group. A user oriented approach involving search to keep the user in-the-loop exposes control to the user for specifying the results they require [7] for each classification.

Signal matching is a process for determining the presence of a known waveform in a larger dataset. In essence this works by sliding the specified template across the data set, computing the similarity of the template at each position in the data series corresponding to how similar the sample was at each position. This allows the user to select a single positive example of a behaviour and classify all occurrences of it in the data.

A distance measure is used to determine a quantitative value corresponding to similarity or dissimilarity between time-series. The choice of distance measure is important when considering the ability to deal with outliers, amplitude differences, and time axis distortion [19]. The most common form of distance measure is Euclidean distance, although it is heavily subjective to noise, and is not capable of depicting objects stretched or compressed along the x or y-axis. Dynamic Time Warping [19] (DTW), and Longest Common Subsequence [1] (LCS), have been introduced to overcome these issues, however, these come as a trade-off with execution time (both quadratic complexity $O(N^2)$). Recent optimizations to DTW [20] have been applied to finding a singular most similar sub-sequence. We require the similarity of a sub-sequence at each position in the series.

Correlation - Correlation is the optimal technique for detecting a known waveform in random noise [36]. In signal processing it is well known that Correlation has a lin-

ear complexity frequency space implementation which other techniques (i.e. DTW and LCS) cannot offer.

There are a number of variations of the cross-correlation technique for different purposes. In time-series space, cross-correlation is the sliding dot product of a template t over a signal f . The result at position x indicates the similarity of the template t in the signal f at the position x .

Cross-correlation in time-series space is a slow operation, but it corresponds to point-wise multiplication in frequency space [36]. This can be used to speed up the cross-correlation process, from $O(N^2)$ in time-series space to $O(2*N)$ complexity in frequency space, where N is the length of the data series. This is implemented by transforming both the template and signal into the frequency domain, which are then multiplied together in frequency space. The inverse Fourier transform of the result forms the filtered answer.

$$c(x) = \mathcal{F}^{-1}[\mathcal{F}(f')\mathcal{F}(t')] \quad (1)$$

The traditional cross-correlation method is sensitive to linear changes in the amplitude of two compared signals [24]. Ralanamahatana et al. [29] state it may be necessary to normalize data to extract meaningful matches. Normalizing excludes linear changes in amplitude by normalizing the signals before performing correlation. The formula for the normalized cross-correlation follows. In the equation, \bar{f}_s denotes the mean value of the signal f under the template t , \bar{t} denotes the mean value of the template t , σ_f denotes the standard deviation of the signal $f(a)$ under the template t , and σ_t denotes the standard deviation of template t .

$$c(x) = \frac{1}{n} \frac{\sum_{s=-a/2}^{a/2} (f(x) - \bar{f}_s)(t(x+s) - \bar{t})}{\sigma_{f(a)}\sigma_t} \quad (2)$$

Fast Normalized Cross-Correlation -The normalized cross-correlation method (Eq. 2) is a slow operation in time-space. To maintain interactivity in our application it is essential this process executes in almost real time. However, this method does not have a simple frequency space expression [24]. Therefore we speed up the operation in the following way.

Our technique is adapted from that presented by Lewis [24]. Utilizing our technique reduces the normalized cross-correlation complexity from quadratic $O(N^2)$ time in time-space to linear $O(N)$ in frequency space. Memory usage is traded-off for algorithm speed to make the system fit for purpose. A total of seven look-up tables of length N are used, with a maximum of four tables required at any one time. The fast normalized cross-correlation approach splits the normalized cross-correlation formula up into two sub-problems, the numerator, and denominator. We first start with the numerator of equation 2.

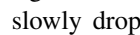
The mean of the template (\bar{t}) can be precomputed when storing the template. An array of the same size as the dataset

is created containing the mean of data values under the template (\bar{f}_s) for each index in the data set. This is precomputed upon executing the normalized cross-correlation function. Utilizing the mean values we can compute a further array containing the signal minus the mean ($(f(x) - \bar{f}_s)$) and another containing the template minus the mean ($(t(x+s) - \bar{t})$). This reduces the numerator to the cross-correlation formula which can be computed using fast Fourier transform in frequency space (Eq. 1).

Next we move on to efficiently computing the denominator of equation 2. The standard deviation (σ) of the signal under the template ($\sigma_{f(a)}$) and template (σ_t) needs to be computed. The standard deviation is computed as follows where μ is the mean input value and N is the length of the input array.

$$\sigma^2 = \frac{1}{N-1} \sum_{i=0}^{N-1} (x_i - \mu)^2 \quad (3)$$

The earlier computed running mean under the template (\bar{f}_s), and the mean of the template (\bar{t}) can be used to speed this up. The standard deviations of both of the means are computed and stored in additional lookup tables. The lookup tables are multiplied together to get the resulting denominator value. The numerator and denominator are then divided together to get the final result set.

Extracting Matches -The results returned by the normalized cross-correlation algorithm may feature multiple matches within the same part of a matching signal. This occurs because of a drop off in similarity values from a matching signal. A match will typically return a high value and then slowly drop off with time (). Directly returning all matches above a set threshold, will result in multiple matches for the same signal at neighboring points in time.

To ensure each matching signal is represented only once, we only extract the maximal point of a match. This involves searching the entire length of the template for the highest matched point along the signal.

Multi-Dimensional Data -The correlation theorems introduced are designed for one-dimensional signals. The data we deal with is of a multi-dimensional nature, and as such the biologists require the ability to define behaviour templates consisting of multiple attributes (e.g. acceleration in three axes). The presence of a behavior may be more dominant in one dimension, for this purpose we compute the sum of the cross-correlation values for each attribute multiplied by an attribute coefficient. This is defined by the strength of a match ($x_i = \sum_{j=0}^N c_j \times s_{i,j}$) at position i , where N is the number of data attributes, c_j defines the coefficient for attribute j , and $s_{i,j}$ defines the similarity value for the associated attribute (j) and position (i). By default each attribute is given an equal weighting ($c = 1/N$), such that, an average of the similarity values for each position is used.

3.6 Interface Design

The design of TimeClassifier has been guided by our domain characterization and classification tasks. The result is a visual analytic system which supports the manual labelling process. We complement this using an extended form of search by applying visual analytics to signal matching for the semi-automatic classification of animal behaviour. Visualization is utilized to assist in finding mis-classifications and interaction techniques to facilitate the checking, rejecting and accepting of results for maintaining a high accuracy. We refer the user to the complementary video in the supplementary material for a demonstration of the system in action. We now detail our system.

The user interface is split into three components (see Figure 1 for overview). Firstly, a data view at the top, being composed of the data in a stacked time-series graph format. Coloured segments overlaid on the graph indicate classified animal behaviour (*Membership*). A search panel is located in the bottom left which the user can perform searches on the data utilizing the template search wizard (*Identify*). Results are shown in this panel for the user to verify, reject (*Reject*) or accept (*Associate*) results before moving them to the appropriate classification in the bottom right panel, where the classification widget is situated. Classified behaviours are shown to the user in this tabulated panel (*Membership*). Each tab represents a behavior group with visualisations for the corresponding set of classified behaviours (*Compare*) and buttons to move (*Move*) or reject (*Reject*) classified instances. The categorical colours assigned to each tab correspond to those overlaid transparently on the time-series graph (*Membership*).

3.6.1 System Workflow

We now detail each of the components of the system workflow for the classification of animal behavior.

Behaviour Selection

The first step is for the user to select a behaviour to classify in the data. There are two methods for this in the application. Firstly, query-by-example, and secondly selecting previously saved behaviour instances from the template database.

Query By Example - Query by example allows the user to directly execute the classification wizard from the time-series display by applying rubber band brushing to subsets of data corresponding to a known behaviour. Specific attributes will be dominant for identifying a behaviour, therefore after selection, a dialog is displayed where the user can select which data attributes to utilize for the template.

Database of Templates - Behaviour templates used in the system can be stored in a database for future use. The database is relational with behaviour templates assigned to classes of animals. The user can query for all patterns present for a specific animal or select an existing behaviour template

previously saved in the database by navigating to the animal of interest and then selecting the appropriate behaviour template.

Signal Resampling - The signal may be resampled to capture events at different frequencies as some behaviours occur at different speeds, for example running. To capture these events independently of the time duration we can store and search for the signal at different time-intervals using re-sampling [36]. Re-sampling is implemented by specifying an irrational factor consisting of an interpolation factor (rate of up-sample) and a decimation factor (rate of down-sample) prior to search.

Classification Wizard

The algorithm discussed in section 3.5 is applied to the data set. The result is the similarity of the specified template at each position in the time-series. Similarity is represented as a percentage of the match, with one hundred percent similarity representing an exact match, whilst for zero there are no matching features. Once the algorithm is executed, the user is presented with the pattern matching results in the classification wizard (Figure 3). This is used to guide the user through refining a similarity threshold to test, reject and accept matched signals by applying their expert knowledge of behavioural patterns and temporal position in the data set.

We introduce two views to assist in extracting matching behaviors. On the left (Figure 3 (b)) are visualisations to show the temporal positions of matches in the data series, while on the right (Figure 3 (c)) we depict all of the extracted matches overlaid on top of each other to show the variance between matches. The visualisations are updated as the threshold value is refined by adjusting a slider corresponding to the threshold percentage (Figure 3 (a)). We now detail these visualizations and the associated user-options for accepting and rejecting matches.

Positions of Matches - We contextualise behaviors over the time domain since the biologist may know additional information through prior exploration of the series. We visualise this using three graphical views (Figure 3 (b)) spatially aligned with a time-series graph of the data series (Figure 3 (b1)). The large nature of our data means the time-series graph is dense. Often this means there can be more than one value per pixel which results in overplotting. We adopt density estimation to represent multiple data items per pixel. The x axis encodes segments of time, such that, each pixel in the display represents many data elements in the series. We compute statistics from each segment of time and represent them using different visual encodings to define our three views.

The confidence of a match visualization (Figure 3 (b2)) depicts a heatmap showing an overview of the pattern matching results to encode where high (blue) and low (yellow) similarity matches occur in the data series. We map maximum match strength to color. The extracted matches view

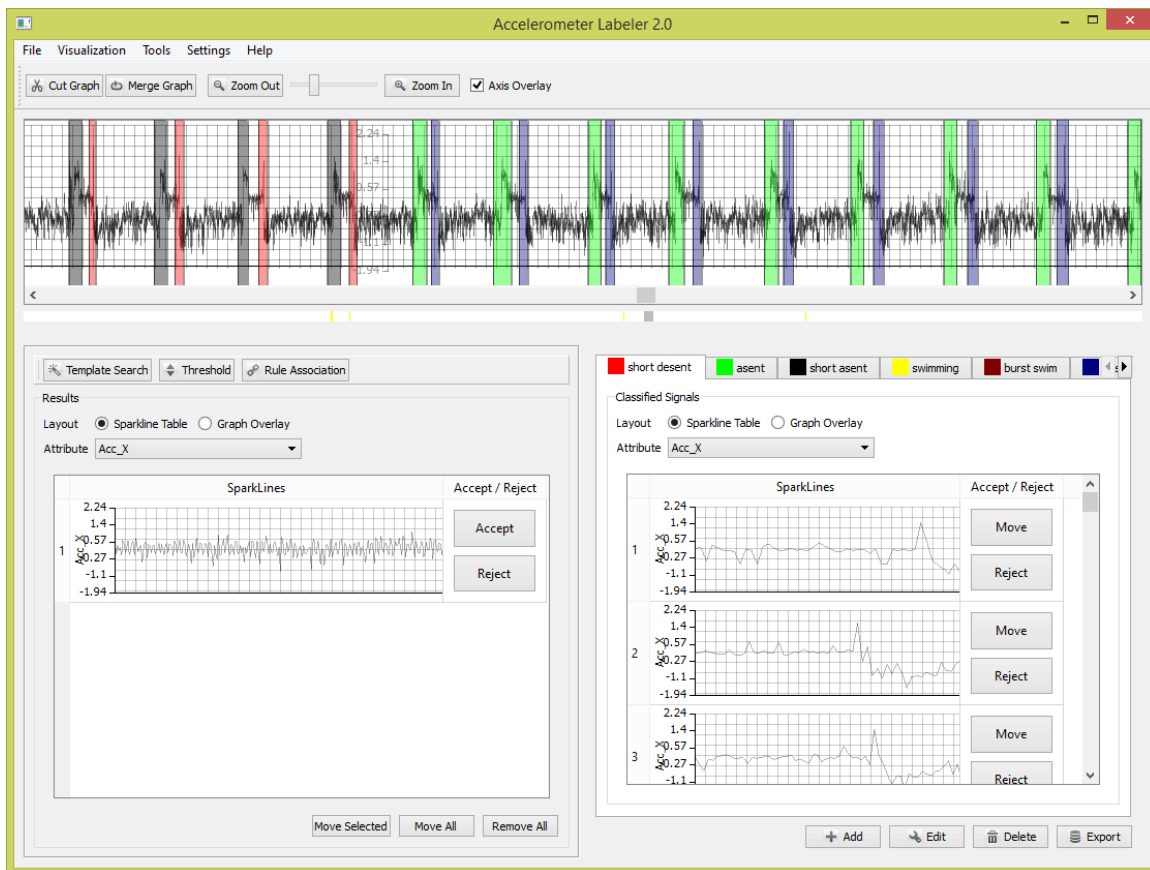


Fig. 1: This figure shows the main user interface of TimeClassifier operating on data from a deployment on an Imperial Cormorant. Top: A stacked time-series graph of the whole data set with overlaid colored regions illustrating labelled animal behaviour. The user can apply zooming to see more detail. Bottom-left: The results widget is used for searching for behaviour instances and displaying results. Bottom-right: The classification widget encapsulates classified results. Each tab represents a classification group with each classified behaviour instance represented using embedded time-series and overlaid plot views.

(Figure 3 (b3)) depicts only extracted matches (above the threshold) and is updated as the similarity threshold is adjusted. Color encodes average extracted match strength. Finally, a distribution of extracted matches (Figure 3 (b4)) which utilises a histogram to map the number of matches at each position to bar height.

The user may refine the result set to reject results from the data series via rubber band brushing across the density displays. Additionally, details on demand can be obtained to delve deeper into the data.

Overlaid Matches - All of the extracted matched signals are overlaid in a stacked time-series graph format, one graph for each data attribute of the pattern (Figure 3 (c)). The user can gain an overview of the general shape of the extracted signals from the graphs. This allows the verification of the shape of extracted matches by the domain expert. Furthermore, most outliers will stand out immediately as they will not fit into the general shape of the extracted results. Some matches will get lost in the overall trend, we therefore utilize line transparency to aid in this. Our users indicated that this is one of the most powerful interactive

elements of TimeClassifier. Changes in the threshold introduce signals which deviate closer or further away from the pattern template (overlaid in red), enabling the user to directly see the cause and effect of modifying the threshold on the general shape of matched signals.

Results can be rejected in this view by the manual selection of lines on the time-series graphs. All results falling within the selection are removed from the result set.

Matched Results

After the user finds an appropriate threshold value, the results are extracted and added to the results widget in the bottom left of figure 1. The user can further inspect the results using our two views. Firstly the separated display, this puts the classifications in a tabular format, with each row corresponding to an identified instance of a behaviour visualized using an embedded time-series plot. The user can accept or reject results by selecting the corresponding button on each row. Secondly, the overlaid plot view overlays the classified instances in a time-series graph. The user can accept or reject results by rubber-band selection on the time-series. The overlaid plot view is useful where the signals shape is simi-

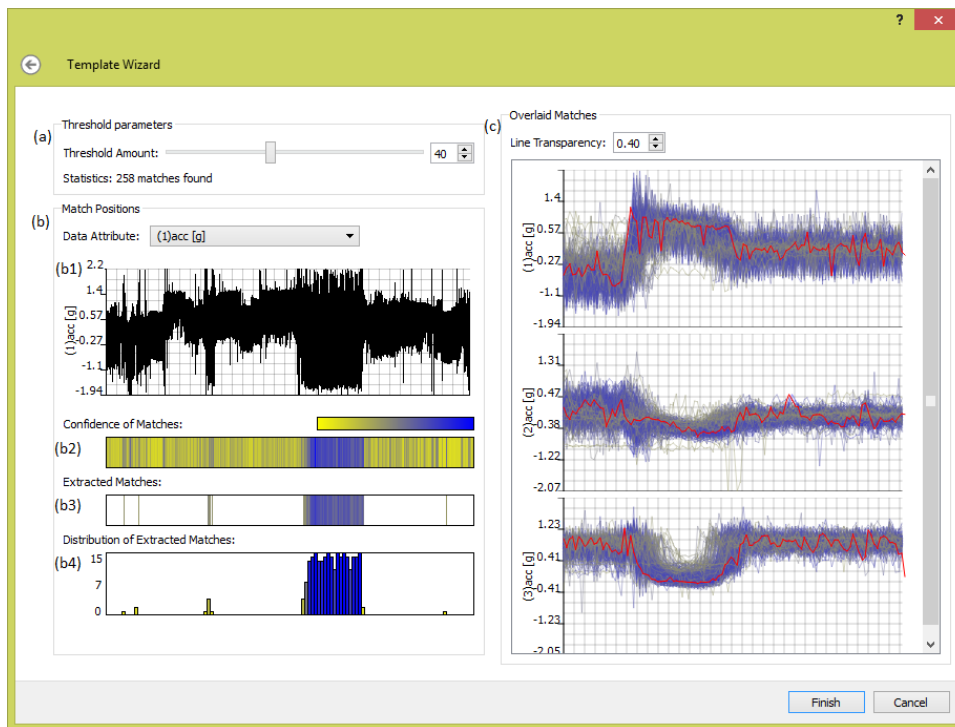


Fig. 2: This figure shows the classification wizard. (a) illustrates our wizard parameters for dynamically adjusting the threshold. (b) shows our density based visualizations to gain an understanding of where matches occur in the data series. (c) shows our overlaid signals visualization of all the extracted matches in a stacked graph format, with one graph for each attribute of the template. The template signal is overlaid in red to show a direct comparison. A yellow to blue color scheme is used, yellow representing low similarity matches, while blue encodes high similarity matches.

lar amongst results, conversely the separated display is best suited where each behaviour signal varies widely. Matches displayed in the results view are also shown in the data view overlaid on top of the time-series graph in grey.

Improving Precision and Recall (Feedback Loop)

It is widely accepted in the machine learning community that achieving 100% precision and recall is a difficult, if not impossible task. The variability and inconsistency of animal behaviour is further challenging. In order to support the classification process and boost the precision and recall we propose that the domain expert is involved in the data analysis loop via feedback with the result set. We introduce three methods for this purpose. Firstly, the user can provide secondary examples of a behaviour to find more behaviour instances. Secondly, the user can directly manipulate the result set to accept and reject matches. Finally, the user can manually classify behaviour.

Secondary Examples - Where the user believes recall to be low, boosting can be used to retrieve more instances. More examples are selected by the user and input into the search wizard. This widens the search space to find patterns related to the secondary retrieved patterns but may not be directly related to the initial search pattern.

Accept / Reject Results - The results panel provides an effective means to inspect the newly found behaviour clas-

sifications. This view allows the user to examine spherical coordinates and parallel coordinates visualizations to assist in determining the correctness of the behaviour classification [37]. Using these additional visualizations along with the time-series views, the user can choose to accept or reject results for the associated classification. Results are accepted by moving them to an appropriate classification tab in the classification widget, or rejected by clicking the reject button. The user should aim to keep accepting / rejecting results until this panel is empty. The user can manually label any part of the time-series by dragging and dropping it onto the classification label

Classified Results

Classified behaviours are shown in two views. Firstly, the classification widget which displays classified behaviour in a corresponding tabulated view. Secondly classified instances are aligned and overlaid on top of time-series graphs as colored rectangular regions identifying where in the data a match for the behaviour has occurred. We allow the user to collapse the labelled regions of the time-series in this view so they can keep track of the remaining unlabelled signal. Each behaviour is identified by a unique color assigned to each classification tab in the classification widget. The final result set of labels can be exported to CSV format aligned with the original data file. This is an important feature requested by

the domain experts to enable them to utilise results in further experiments.

4 Evaluation

The project has been an iterative design process with many user interface and algorithmic improvements to support the biologists way of working. The system has been deployed with 14 biologists using data from animals such as turtles, penguins, condors, and badgers, to name a few. Feedback was collected and improvements were made throughout the project. Many small requests (like derived data and CSV export) were made to increase convenience and reduce the time of use. We present the results of two field trials on the current evolution of our system to demonstrate the usefulness of our technique from a movement ecology context in comparison to the current manual process implemented in our software. The feedback we received has been positive, to quote one expert, *We are very excited about the power of the iterative approach.* We then discuss a formal user study to evaluate against the manual approach. We refer the user to the supplementary material for an evaluation against traditional data mining approaches

4.1 Case Study 1 - Turtles

Biologists are very interested in identifying wild animal locomotion because it is one of the most energetically costly behaviours [2]. This is particularly germane in air-breathing vertebrates such as penguins, seals and turtles because the costs of locomotion are dependent on depth, being modulated by the degree of compression of air spaces within the body and resultant upthrust [42]. Specifically, knowledge of when air-breathing vertebrates engage in limb stroke behaviour helps understand power, and therefore oxygen, use during dives, and can help biologists understand strategies for maximizing performance [34].

Identification of limb strokes is, however, challenging. Although it can be undertaken manually using acceleration data, with limb beat frequencies of up to several Hertz, in datasets that can last days or even months this is effectively impossible. A particular challenge is that the acceleration channel values change with animal posture [35] so that the limb beat signal is superimposed on this variable baseline.

A domain expert analysed a section of a turtle dataset, 154,090 data points of surge acceleration recorded at 40 Hz (using a Daily Diary tag [43]), both by manual inspection and using the program to identify flipper beating, and compared the two performances. In a first, simple run based on a single flipper beat, the program took 12 minutes to set up, modify parameters, and finally run to produce an output compared to 86 minutes for the manual option. TimeClassifier correctly identified flipper beating, and absence of flipper beating 79% of the time. However, manual determination of flipper beating was considered to be highly subjective

for at least 20% of the time because turtle flipper beats vary greatly in amplitude, period and even in the extent to which one stroke (wavelength) is executed fully. Thus, the program performed extremely well in identifying effectively almost all clear-cut cases. In a second run, the program was run iteratively with, first, a single turtle flipper beat being used as a template to find its equivalence in the dataset using a high similarity threshold. The data set was then examined to determine which flipper beats had not been found by this process before another single flipper beat was added and the process repeated. Five iterations of this type identified all but an estimated 4 flipper beats. This gave a time-based accuracy of 99.86%. This process also took 12 mins (the faster time being largely due to greater familiarity with the program). Assuming that 12 minutes is the standard time allocated to using the program, and that this effectively varies little with the size of the dataset, the program takes the same time to categorize about 9 minutes of dataset (21,600 data points) into behaviours as the manual approach. With any comparable data set longer than this, the program is therefore more efficient than the manual approach by a factor of the dataset length (mins)/9 in terms of time.

4.2 Case Study 2 - Soaring Birds

Soaring birds such as vultures rely almost entirely on environmentally generated lift to remain aloft [23]. These updrafts may be broadly classified as either slope-generated lift or thermal lift, the latter providing a much greater energetic gain for birds. Identification of where and when birds use the two different lift types is important for conservation but problematic for biologists. Multiple transducers in animal-attached tags can help identify slope-soaring and thermal soaring behaviours but barometric pressure sensors and magnetometers are most useful because pressure transducers can indicate height gain while magnetometers can indicate circling. However, manual inspection of multiple sensors operating at high frequencies is time consuming and requires a great deal of expert knowledge.

A domain expert analysed data recorded during 7 flights of a condor totalling 4 hours (577,770 data points), manually inspecting primarily the pressure and magnetometry data for patterns indicative of thermal and slope soaring. TimeClassifier, was then applied with a threshold in the barometric pressure to identify where lift occurred and repetitive waveforms in the magnetometry data to differentiate slope soaring from thermal soaring. The manual inspection took 110 mins to complete while the program, including parameter modification, took 23 mins. The concurrence between the manual and program performance was 76%, an excellent result given that, as with the penguin data set, there was a substantive amount of subjectivity in the manual interpretation. The second, iterative, approach (described above) with the vulture data set led to a behavioural accuracy of 100% after

18 mins (the faster time again being largely due to greater familiarity with the program second time around). Using the same criteria for time efficiency as with the turtle case study, the program is more efficient than the manual approach by a factor of the dataset length (mins)/50 in terms of time for the simple (first) approach.

4.3 Formal User-Study

A user study was carried out to assess the effectiveness of our new approach for labelling animal activity data with respect to the existing manual solution by simulating the behavioural labelling process that biologists undertake. To simplify the study we highlight the behaviours in the time-series with a gray block. Users are required to indicate whether each highlighted signal is behavior A or B (by right clicking in the block and selecting the appropriate label). The block turns to the color representing that behavior.

We provide a counter of the number of segments left to label. We ask users to target getting this to zero, but not to spend minutes looking for the last remaining one or two segments. We time from the presentation of the stimuli until the user clicks next. Accuracy is measured according to how many behaviours are correctly labelled.

We obtained a total of 30 participants who each received a 10 book voucher. They were all computer scientists with average age of 25.

The experiment began with a brief overview read by the experimenter using a predefined script. Detailed instructions were then given through a self-paced slide presentation. The presentation included a description of the study and also a briefing on how to interact with the software for labelling purpose. Participants also received a color copy of the presentation for reference during the study if desired. The study was closely monitored, at least two experimenters were always present in the room and participants abode to the study requirements

We report the average time to label of 91.9 seconds and an average accuracy of 95% - so they get 5% of labels incorrect. There are around 25 labels per stimuli and 12 stimuli overall. Each stimuli was constructed from a small data set (between 25,000 to 30,000 data items in length) where we pre-segment the data and just require users to choose between two labels for each segment. The computed template version achieved 100% accuracy in an average of 1.2 seconds.

5 Conclusions and Future Work

In this paper, we present a visual analytics system for the semi-automatic classification of animal behaviour which brings together the expert knowledge of the user with those of the pattern matching capabilities of the matched filtering algorithm applied to classification. We deploy our system with

biologists and report a number of real-world case studies. The results demonstrate the value of visual analytics to making a positive impact on the work of movement ecologists, as the two case studies demonstrate substantial improvement to workflow efficiency. While the scope of our application is applied to a movement ecology context. We believe TimeClassifier has uses in other domains where classification of time-series is applied, such as, electrocardiograms, seismographs, and industrial processes, to name a few.

After working extensively with biologists we have seen beyond standard criticisms of machine learning (i.e. 'black box'). They encounter low precision and recall from these systems and therefore have low confidence in them and express frustration with the black box phrase [22]. When faced with low precision and recall, the available algorithms do not have an accessible means by which to rectify any mistakes beyond supplying an unknown quantity of further training examples, choosing alternative algorithms or changing parameters. Involving the user-in-the-loop overcomes these issues which leads to more trust in results, along with increased efficiency and accuracy.

In future work we wish to apply our system and workflow across other data domains. Furthermore, the application of a visual analytic approach may enhance other mining tasks in time-series data. i.e. clustering, indexing, pattern discovery, prediction, and segmentation. Finally, we wish to investigate other means of visualizing the labelled data to provide insight into animal behaviour.

6 Acknowledgements

This work was funded by an EPSRC doctoral training grant.

References

1. Abdulla-Al-Maruf, A., Huang, H.H., Kawagoe, K.: Time series classification method based on longest common subsequence and textual approximation. In: Digital Information Management (ICDIM), 2012 Seventh International Conference on, pp. 130–137 (2012)
2. Alexander, R.M.: Models and the scaling of energy costs for locomotion. *Journal of Experimental Biology* **208**(9), 1645–1652 (2005)
3. Bidder, O.R., Campbell, H.A., Gmez-Laich, A., Urg, P., Walker, J., Cai, Y., Gao, L., Quintana, F., Wilson, R.P.: Love thy neighbour: Automatic animal behavioural classification of acceleration data using the k-nearest neighbour algorithm. *PLoS ONE* **9**(2), e88,609 (2014)
4. Bidder, O.R., Qasem, L.A., Wilson, R.P.: On higher ground: How well can dynamic body acceleration determine speed in variable terrain? *PLoS ONE* **7**(11), 50,556 (2012)
5. Blaas, J., Botha, C.P., Grundy, E., Jones, M.W., Laramee, R.S., Post, F.H.: Smooth graphs for visual exploration of higher-order state transitions. *IEEE Transactions on Visualization and Computer Graphics* **15**(6), 969–976 (2009)
6. Bouali, F., Devaux, S., Venturini, G.: Visual mining of time series using a tubular visualization. *The Visual Computer* pp. 1–16 (2014)

7. Buono, P., Aris, A., Plaisant, C., Khella, A., Shneiderman, B.: Interactive pattern search in time series. pp. 175–186. *SPIE* (2005)
8. Ellis, K., Kerr, J., Godbole, S., Lanckriet, G., Wing, D., Marshall, S.: A random forest classifier for the prediction of energy expenditure and type of physical activity from wrist and hip accelerometers. *Physiological measurement* **35**(11), 2191 (2014)
9. van den Elzen, S., van Wijk, J.: Baobabview: Interactive construction and analysis of decision trees. In: *Visual Analytics Science and Technology (VAST)*, 2011 IEEE Conference on, pp. 151–160 (2011). DOI 10.1109/VAST.2011.6102453
10. Esling, P., Agon, C.: Time-series data mining. *ACM Comput. Surv.* **45**(1), 12:1–12:34 (2012)
11. Gao, L., Campbell, H.A., Bidder, O.R., Hunter, J.: A web-based semantic tagging and activity recognition system for species' accelerometry data. *Ecological Informatics* **13**(0), 47–56 (2013)
12. Gleiss, A.C., Wilson, R.P., Shepard, E.L.C.: Making overall dynamic body acceleration work: on the theory of acceleration as a proxy for energy expenditure. *Methods in Ecology and Evolution* **2**(1), 23–33 (2011)
13. Gregory, M., Shneiderman, B.: Shape identification in temporal data sets. In: *Expanding the Frontiers of Visual Analytics and Visualization*, pp. 305–321. Springer (2012)
14. Grundy, E., Jones, M.W., Laramée, R.S., Wilson, R.P., Shepard, E.L.C.: Visualisation of sensor data from animal movement. *Comput. Graph. Forum* **28**(3), 815–822 (2009)
15. Hao, M.C., Dayal, U., Keim, D.A., Schreck, T.: Importance-driven visualization layouts for large time series data. In: *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pp. 203–210. IEEE (2005)
16. Hao, M.C., Marwah, M., Janetzko, H., Dayal, U., Keim, D.A., Patnaik, D., Ramakrishnan, N., Sharma, R.K.: Visual exploration of frequent patterns in multivariate time series. *Information Visualization* **11**(1), 71–83 (2012)
17. Holz, C., Feiner, S.: Relaxed selection techniques for querying time-series graphs. In: *UIST '09: Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 213–222. ACM, New York, NY, USA (2009)
18. Janicke, H., Bottinger, M., Mikolajewicz, U., Scheuermann, G.: Visual exploration of climate variability changes using wavelet analysis. *IEEE Transactions on Visualization and Computer Graphics* **15**(6), 1375–1382 (2009)
19. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. *Knowl. Inf. Syst.* **7**(3), 358–386 (2005)
20. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. *Knowledge and information systems* **7**(3), 358–386 (2005)
21. Kincaid, R.: Signallens: Focus+context applied to electronic time series. *Visualization and Computer Graphics, IEEE Transactions on* **16**(6), 900–907 (2010)
22. von Landesberger, T., Andrienko, G., Andrienko, N., Bremm, S., Kirschner, M., Wesarg, S., Kuijper, A.: Opening up the black box of medical image segmentation with statistical shape models. *The Visual Computer* **29**(9), 893–905 (2013)
23. Lanzone, M.J., Miller, T.A., Turk, P., Brandes, D., Halverson, C., Maisonneuve, C., Tremblay, J., Cooper, J., O'Malley, K., Brooks, R.P., Katzner, T.: Flight responses by a migratory soaring raptor to changing meteorological conditions. *Biology Letters* **8**(5), 710–713 (2012)
24. Lewis, J.P.: Fast template matching. *Vision Interface* **95**, 120–123 (1995)
25. Lin, J., Keogh, E., Lonardi, S.: Visualizing and discovering non-trivial patterns in large time series databases. *Information Visualization* **4**(2), 61–82 (2005)
26. Liu, S., Cui, W., Wu, Y., Liu, M.: A survey on information visualization: recent advances and challenges. *The Visual Computer* **30**(12), 1373–1393 (2014)
27. Nathan, R., Spiegel, O., Fortmann-Roe, S., Harel, R., Wikelski, M., Getz, W.M.: Using tri-axial acceleration data to identify behavioral modes of free-ranging animals: general concepts and tools illustrated for griffon vultures. *The Journal of experimental biology* **215**(6), 986–996 (2012)
28. Payne, N.L., Taylor, M.D., Watanabe, Y.Y., Semmens, J.M.: From physiology to physics: are we recognizing the flexibility of biological tools? *J Exp Biol* **217**(Pt 3), 317–22 (2014)
29. Ratanamahatana, C., Lin, J., Gunopulos, D., Keogh, E., Vlachos, M., Das, G.: *Data mining and knowledge discovery handbook 2010*. 2nd edn. o. maimon, l. rokach (2010)
30. Ropert-Coudert, Y., Wilson, R.P.: Trends and perspectives in animal-attached remote sensing. *Frontiers in Ecology and the Environment* **3**(8), 437–444 (2005)
31. Ryall, K., Lesh, N., Lanning, T., Leigh, D., Miyashita, H., Makino, S.: Querylines: Approximate query for visual browsing. In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems, CHI EA '05*, pp. 1765–1768 (2005)
32. Sakai, M., Aoki, K., Sato, K., Amano, M., Baird, R.W., Webster, D.L., Schorr, G.S., Miyazaki, N.: Swim speed and acceleration measurements of short-finned pilot whales (*Globicephala macrorhynchus*) in Hawai'i. *Mammal study* **36**(1), 55–59 (2011)
33. Sakamoto, K.Q., Sato, K., Ishizuka, M., Watanuki, Y., Takahashi, A., Daunt, F., Wanless, S.: Can ethograms be automatically generated using body acceleration data from free-ranging birds? *PLoS ONE* **4**(4), e5379 (2009)
34. Sato, K., Charrassin, J.B., Bost, C.A., Naito, Y.: Why do macaroni penguins choose shallow body angles that result in longer descent and ascent durations? *Journal of Experimental Biology* **207**(23), 4057–4065 (2004)
35. Shepard, E.L.C., Halsey, L.G.: Identification of animal movement patterns using tri-axial accelerometry. *Endangered Species Research* **10**, 47–60 (2008)
36. Smith, S.W.: *The scientist and engineer's guide to digital signal processing*. California Technical Publishing, San Diego, CA, USA (1997)
37. Walker, J., Geng, Z., Jones, M., Laramée, R.S.: Visualization of Large, Time-Dependent, Abstract Data with Integrated Spherical and Parallel Coordinates. pp. 43–47. Eurographics Association, Vienna, Austria (2012)
38. Ware, C., Arsenault, R., Plumlee, M., Wiley, D.: Visualizing the underwater behavior of humpback whales. *IEEE Computer Graphics and Applications* **26**(4), 14–18 (2006)
39. Watanuki, Y., Takahashi, A., Daunt, F., Wanless, S., Harris, M., Sato, K., Naito, Y.: Regulation of stroke and glide in a foot-propelled avian diver. *Journal of Experimental biology* **208**(12), 2207–2216 (2005)
40. Weber, M., Alexa, M., Muller, W.: Visualizing time-series on spirals. In: *Information Visualization, 2001. INFOVIS 2001. IEEE Symposium on*, pp. 7–13
41. van Wijk, J., Van Selow, E.: Cluster and calendar based visualization of time series data. In: *Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on*, pp. 4–9, 140
42. Wilson, R.P., Hustler, K., Ryan, P.G., Burger, A.E., Noldeke, E.C.: Diving birds in cold water: do archimedes and boyle determine energetic costs? *American Naturalist* pp. 179–200 (1992)
43. Wilson, R.P., Shepard, E., Liebsch, N.: Prying into the intimate details of animal lives: use of a daily diary on animals. *Endangered Species Research* **4**, 123–137 (2008)
44. Zhao, J., Chevalier, F., Pietriga, E., Balakrishnan, R.: Exploratory Analysis of Time-series with ChronoLenses. *IEEE Transactions on Visualization and Computer Graphics* **17**(12), 2422–2431 (2011)