# Visual discovery and model-driven explanation of time series patterns

Advait Sarkar*, Martin Spott†, Alan F. Blackwell*, Mateja Jamnik*
*Computer Laboratory, University of Cambridge, UK
{advait.sarkar, alan.blackwell, mateja.jamnik}@cl.cam.ac.uk

† HTW Berlin, Germany (previously at BT Research and Technology, Ipswich, UK)
Martin.Spott@HTW-Berlin.de

***Abstract*—Gatherminer is an interactive visual tool for analysing time series data with two key strengths. First, it facilitates bottom-up analysis, i.e., the detection of trends and patterns whose shapes are not known beforehand. Second, it integrates data mining algorithms to explain such patterns in terms of the time series' metadata attributes – an extremely difficult task if the space of attribute-value combinations is large. To accomplish these aims, Gatherminer automatically rearranges the data to visually expose patterns and clusters, whereupon users can select those groups they deem 'interesting.' To explain the selected patterns, the visualisation is tightly coupled with automated classification techniques, such as decision tree learning. We present a brief evaluation with telecommunications experts comparing our tool against their current commercial solution, and conclude that Gatherminer significantly improves both the completeness of analyses as well as analysts' confidence therein.**

## I. Introduction

Exploratory statistical analysis, as conducted through visual analytics tools, can be regarded as an instance of end-user programming. Like spreadsheets, visual analytics tools focus on presenting data, rather than the control flow of programs operating on the data.

In this paper, we present Gatherminer, a visual analytics tool specifically designed for the analysis of time series data. Time series analysis immediately presents a visual design challenge, because a natural mapping of the time dimension to one of the axes on the visual plane greatly reduces the remaining options for visualising relations between data and model. Just as the grid formalism introduced a strong design constraint that led to the spreadsheet paradigm, the constraint of time series analysis provides an opportunity for new creative exploration of the design space for statistical modelling.

*The BT problem:* This work is grounded in a specific application domain – analysing patterns of faults in the BT network. At BT Research and Technology, analysts study time series of faults in the various devices on BT's telecommunications network. An international network infrastructure comprises of the order of $10^6$ devices, including hubs, routers, cables, etc. in the core network, in exchanges, and in customers' homes. Each device is characterised by hundreds of metadata attributes, such as their geographic location, the customer type served, etc. Every day, faults on the network are logged, e.g., through devices raising alarms, by field engineers performing maintenance, and by customer reports. Thus, a time series of daily fault counts is created for each *type* of device (i.e., all devices which share metadata properties).

Analysts identify patterns of faults in subsets of this large database of time series, and look for potential causes of these patterns. Once interesting behaviour, such as "devices with unusually high fault rates", or "devices with an inflexion in the time series" is found, a corresponding explanation is sought, such as "do any device types consistently seem to have inflexions?", or "what attribute values are predictive of devices with unusually high fault rates?" These explanations are then used to drive business decisions, such as investment allocation and special investigations.

The analysts face two important challenges. The first is that the shape of interesting patterns is not known beforehand, so the system cannot be querying-oriented; i.e. "interesting" time series cannot simply be retrieved. Even if an interesting pattern is known, it may not always be straightforward to express using standard tools such as relational databases. The second challenge is that of finding a concise explanation for these patterns in terms of the metadata attributes. Having hundreds of attributes, each with several values, leads to a combinatorial explosion of attribute-value combinations which cannot each be manually inspected. A further complication is that the analysts are experts in the domain of the network data but have limited statistical expertise.

In studying this process at BT, we observed that the task of detecting and explaining interesting patterns is typically performed using opportunistic approaches with notable drawbacks: (1) they rely heavily on the domain expertise of the analyst to guide exploration of the large space of attribute-value combinations; (2) they can result in interesting features being overlooked; (3) they can result in spuriously correlated attribute explanations being 'discovered'; (4) they rely on extensive manual attribute inspection. Consequently, current methods result in *incomplete, inaccurate,* and *slow* analyses, and leave analysts feeling *unconfident* about their analysis.

Gatherminer directly addresses these drawbacks using a compact visualisation scheme, automated rearrangement, and explanations driven by machine learning. We compared our tool against Tableau [1], our expert analysts' current tool, and found that analyses using our tool are more complete, more often correct, faster, and improve analyst confidence.

978-1-5090-0252-8/16/$31.00 ©2016 IEEE

## II. Related work

The analysts' objectives can be expressed through Amar et al.'s analytical framework [2] as follows:

- *Identifying interesting features*: detecting groups of similar time series (Cluster) by identifying trends, peaks, inflexions and their overall shapes (Extrema, Range, Distribution, Anomalies).
- *Explaining features in terms of attributes*: detecting potential causal links between time series attributes and their behaviour (Correlation).

### A. Visualisations for bottom-up time series analysis

Since the nature of interesting time series is unknown *a priori* (e.g., it is not possible to say whether we wish to retrieve series with peaks, troughs, inflexion points, or some other behaviour), we must enable the analyst to conduct *bottom-up* analyses, where hypotheses about interesting behaviour are first generated by inspecting the data [3]. For this process to be robust, hypotheses must be generated from the most thorough, complete, and consistent inspection of the dataset possible. The problem of accurately and compactly representing multiple time series has been tackled in many ways [4]; a synthesis of multi-resolution techniques is given by Hao et al [5]. Pixel-matrix displays are used to represent time series in several scientific disciplines, such as gene expression data [6] and machine hearing [7].

To facilitate better discovery of collective trends from such overviews, the technique of reshuffling data series was proposed by Bertin with his 'reorderable matrix' [8]. Bertin proposed a visual procedure where pieces of paper representing rows of a matrix were cut and manually reordered on a flat surface. We now have the computational resources and advanced clustering techniques to adapt this method for large datasets. A survey of time series clustering techniques is given by Liao [9]. Elmqvist et al. incorporated a significant reordering step in their "Zoomable Adjacency Matrix Explorer" [10]. Mansmann et al. explored the use of correlation-based arrangements of time series for movement analysis in behavioural ecology [11]. The "Bertifier" is a general-purpose tool for applying reordering operations to tabular data [12]. Previous work has also been done on exposing motifs in time series [13], [14]. These systems do not, however, build on the rearranged visualisation to present a visual language for conducting automated analyses of the metadata attributes.

### B. Explaining behaviour in time series datasets

Bernard et al.'s system [15] visually guides the discovery of metadata properties of time series clusters, which is closely related to our goals. However, while their work was primarily focused on automated notions of "interestingness," due to the open-ended nature of BT analyses, we must necessarily take a mixed-initiative approach, with interestingness defined by ad-hoc user selections. A number of interfaces have been proposed for performing information retrieval tasks on time series databases, such as sketch editors and visual catalogues [16]–[18], but these systems are query-driven (i.e., assume the nature of the interesting pattern is known beforehand).

The Line Graph Explorer [19] compactly represents line graphs as rows of colour-mapped values, that is, a colour-mapped matrix. This provides a full overview of the time series data in a compact space. Line Graph Explorer provides a focus+context view using a lens-like tool, and has an elaborate metadata panel which draws on the table lens [20]. The metadata panel facilitates visual correlation of observed patterns with metadata attributes, but relies on manual inspection and so does not scale to large attribute spaces.

Keim et al. identify "advanced visual analytics interfaces" [21], which showcase an advanced synergy between visualisation and analytics. Hao et al. describe "intelligent visual analytics queries" [22], the process of selecting a focus area, analysing the selection, and presenting results of the analysis as appropriate. This is precisely the technique we employ.

A number of investigations have been made into improving visual interaction with various statistical procedures. These procedures include exploratory approximate computation [23], distance function learning [24], [25], and advanced feature space manipulations [26]–[28]. Fails and Olsen [29], Wu and Madden [30], and Behrisch et al. [31] present systems for interactive machine learning. These systems address a wide range of classification problems for different types of data; however, visual mining for explanations amongst a large set of attributes has not been addressed.

## III. Gatherminer design and architecture

In this section we describe the architecture and design decisions behind Gatherminer. Our prototype is implemented using web technologies and can load local CSV files.

Data is represented as a colour-mapped matrix (Fig. 1(a)), which is rearranged to expose patterns and clusters (we refer to this as "gathering" [32]). To navigate this visualisation, which can become very large for massive databases of time series, we provide an overview+detail mechanism, where the overview is facilitated by a thumbnail scrollbar (Fig. 1(b)), and detail is given through a scanning display (Fig. 1(c)). For analysis, we use selection on the core visualisation as annotation to deploy explanation procedures such as summary bar graphs and decision tree learning (Fig. 1(d)). We now elaborate upon the individual components.

### A. Core colour-mapped matrix visualisation

Our primary visualisation is a colour-mapped matrix where each row is an individual time series and each column is an individual time point. Thus, a cell is a single data point within a single time series, coloured according to its value. This representation has useful properties [5], most importantly compactness: each datum can be shrunk to a single pixel in size before the visualisation ceases to be lossless. Prior to colour-mapping, values must be normalised; by default Gatherminer provides cumulative distribution function normalisation, range normalisation and $Z$-score normalisation, but a user-supplied JavaScript normalisation function may also be used.

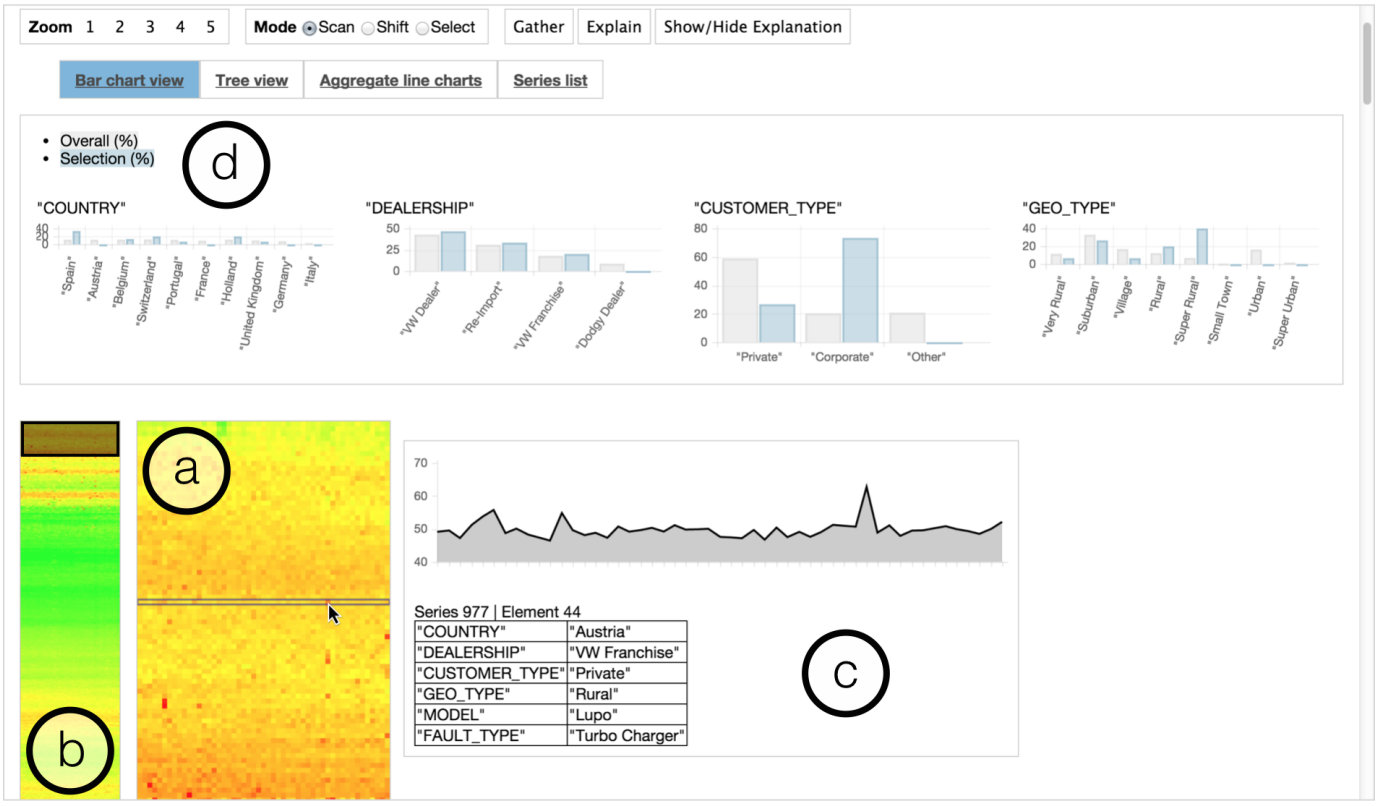


Fig. 1. The Gatherminer software showing (a) its primary colour-mapped matrix visualisation, (b) the thumbnail overview scrollbar, (c) scanning for detail, (d) attribute charts generated during analysis.

Importantly, even though pixel matrices are perceptually inferior to line charts for comparative quantitative analysis, since they rely on colour rather than height to convey magnitude, it is the *identification* of patterns which we consider to be more important than *characterisation*. That is, it is more important for analysts to be able to spot that interesting behaviour exists, rather than to immediately understand the nature of that behaviour (spike, trough, etc.). The colour-mapped matrix greatly facilitates identification. The exact behaviour can easily be further characterised by inspecting aggregate line graphs generated by selecting the time series.

To illustrate an example workflow, we have taken an actual BT dataset and disguised commercially sensitive material so as to resemble data about faults in cars. The colour-mapped matrix of this dataset when initially loaded can be seen at the top of Fig. 2. We will return to this figure in §III-B.

*Overview+detail:* The warping lens in Line Graph Explorer provides focus+context [33], allowing individual time series to be inspected in detail whilst still being aware of the series' location in the overall dataset. The lens dynamically distorts the underlying visualisation, which we require to be static for purposes of selection, so it is not applicable to our use. Instead, we allow the user to scrub over the visualisation, and display a detailed line graph and attributes table for the series being hovered over (Fig. 1(c)). This is complemented with an overview which never exceeds the height of the screen. The overview acts as a scrollbar [34]; thus, the scrollbar, main visualisation, and scanning together create an overview+detail view. This has an additional advantage over the lens approach: the time series dataset can be large, containing thousands of time series, but a shrunken representation is always visible and available for use as a navigational aid.

### B. Gathering: automated layout

The visualisation format alone facilitates some analysis, but for more efficient pattern detection a layout algorithm must now be applied. A number of clustering, sorting, or optimisation methods may be appropriate here; by default, Gatherminer reorders the time series such that those which are most similar are placed close together, hence "gather." Specifically, the final layout minimises the sum of pairwise distances between neighbouring time series.

In the following, we use $T$ to denote a univariate time series. The subscript $T_i$ denotes the $i^{th}$ element of $T$. In a collection of many time series, the superscript notation $T^{(k)}$ denotes the $k^{th}$ series. Thus, $T_i^{(k)}$ denotes the $i^{th}$ element of the $k^{th}$ series.

Initially, the distance between any two time series is defined using a sliding weighted metric:

$$distance(T^{(a)}, T^{(b)}) = \sum_i \sum_j \left| T_i^{(a)} - T_j^{(b)} \right| \cdot w(i,j) \quad (1)$$

where $w$ specifies how the neighbourhood of each element is weighted, e.g., $w(i,j) = \mathrm{e}^{-(|i-j|)}$.

For $n$ series we find an ordering $\{T^{(1)}, T^{(2)}, ..., T^{(n)}\}$ that minimises $\sum_{i=1}^{n-1} distance(T^{(i)}, T^{(i+1)})$. The visual principle behind this ordering is that by minimising the sum of pairwise distances between neighbouring series, we bring together series which have similar visual colour profiles. When 'stacked up', rows of similar colours create larger areas which are easily spotted. Finding such an ordering is not straightforward, as the problem is equivalent to that of finding a minimal Hamiltonian path (similar to the travelling salesman problem of finding a minimal Hamiltonian cycle), known to be NP-complete. To see why our problem is equivalent, imagine that each time series is a vertex in a fully-connected graph, and each edge has weight equal to the distance between its two vertices. A complete minimal ordering is equivalent to a minimal-weight path which touches each vertex exactly once, i.e., a minimal Hamiltonian path. Computing the distance matrix is an unavoidable $O(n^2)$. Thereafter, to compute the ordering, Gatherminer implements a greedy nearest-neighbour search ($O(n^2)$), and genetic optimisation algorithm (also $O(n^2)$, but is slower because of larger constant factors) for cases where the greedy search produces poor orderings [35].

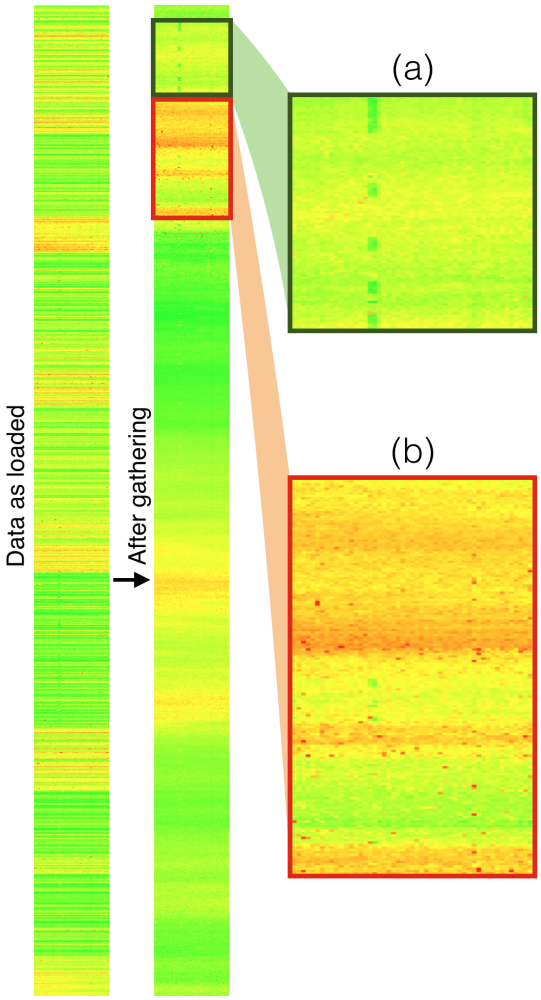


Fig. 2. Two patterns exposed by gathering a BT dataset of 1,335 series. (a) Series with a trough in the middle. (b) Clusters of high-valued time series.

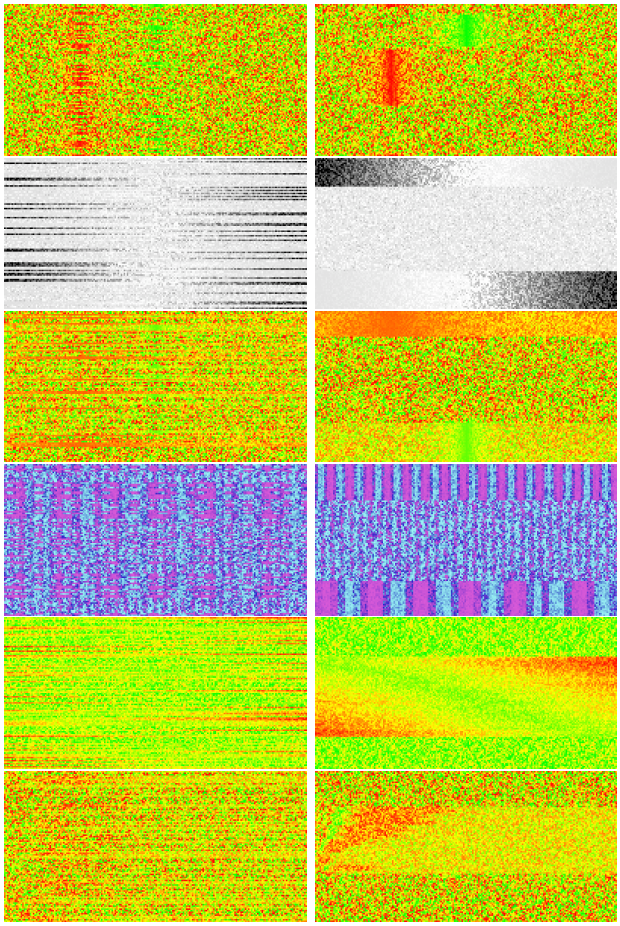

Fig. 3. Examples of the gathering process, with the dataset loaded as-is on the left, and after gathering on the right, demonstrating the detection and separation of different types of patterns from noise. From top to bottom: peaks and troughs, linear trends, features of different widths, functions of different periodicities, complex cross-series cascades. Gatherminer supports many colour mappings.

The "Gather" button triggers reordering. The resulting visualisation exposes groups of series bearing interesting analytical features such as peaks and trends (Fig. 3). The colour-mapped matrix representation of our faults dataset after gathering can be seen in Fig. 2. One limitation of this process is that each time series can only have two neighbours in the colour-mapped matrix, and so clusters are sometimes "flattened" counterintuitively, with similar rows being placed further apart than expected. However, this does not usually impair "pattern spotting" as it is an approximate visual process.

The distance metric can and should be changed for the task at hand, as various domains typically have very different notions of data similarity. For instance, Dynamic Time Warping [36] is a common metric for comparing time series which vary in speed or time. Currently, any user-supplied JavaScript distance function can be used. In future work, it would be useful to investigate interactive visual methods of specifying distance metrics [24]–[26].

Each time series in the collection is associated with attributes describing various properties of the series (i.e., "meta-

data" as used by Kincaid and Lam [19] and Bernard et al [15]). For instance, the BT fault data has "Device Type", "Location", etc. as attributes. We denote these attributes $A_j$, meaning the set of values they are allowed to have. Thus, each time series is characterised by an $n$-tuple of attribute values $(a_1, a_2, ..., a_n)$ where $\forall j.\ a_j \in A_j$.

We can now frame the two primary activities of BT analysts which are facilitated by our system as follows:

- "Identifying interesting features" corresponds to discovering the sets of time series *Interesting* such that for $k \in$ *Interesting*, $T^{(k)}$ contains interesting behaviour, for example, "devices with unusually high fault rate."
- "Explaining features in terms of attributes" corresponds to discovering attribute-value tuples $(a_1, a_2, ...)$ which discriminate well between $T^{(k)} \in$ *Interesting* and $T^{(k')} \notin$ *Interesting*. An example question is "what attribute values are predictive of devices with unusually high fault rates?"

### C. Selection to annotate interesting clusters

The Gathering process exposes interesting patterns as visual artefacts such as coloured blobs and streaks. Users select such regions of interest in order to mark them as 'interesting.' This constitutes manual annotation of a subset of data points, similar to the interactive applications presented by Fails and Olsen [29], and Wu and Madden [30]. In Fails and Olsen's *Crayons* application, the user drew on an image to interactively build a classifier to segregate the image (e.g., a classifier that detects a human hand against a background). Similarly, in Gatherminer, the user directly annotates the visualisation to build a classifier. While Crayons facilitated image classification on image data, Gatherminer extends that style of interaction to time series data visualised as a colour-mapped matrix; it provides an "intelligent visual analytics query" [22].

The gathering step is essential for this annotation to be effective. In the underlying dataset the time series may appear in any ordering, for instance the order of generation of the data entries, or sorted by attribute-values. Once gathered, however, the resultant ordering $\{T^{(1)}, T^{(2)}, ..., T^{(n)}\}$ is such that neighbouring time series have similar behaviours. Thus, "interesting" time series appear in contiguous regions, allowing the user to use their selection to specify an interval $[a, b]$, or $k$ intervals $[a_i, b_i]$ for $i = 1$ to $k$, such that $\bigcup_{i=1}^{k} \{T^{(a_i)}, T^{(a_i+1)}, ..., T^{(b_i)}\}$ constitute the *interesting* set, and the remaining time series constitute a *not-interesting* set.

Once the selection is made, clicking the "Explain" button deploys multiple strategies (e.g., decision tree learning) to discover which attributes of the time series best discriminate the interesting (selected) regions from the not-interesting ones. Thus, the user asks the software to "explain" regions of interest by querying for explanatory attributes.

Gatherminer currently supports two explanation methods, illustrating the variety of interesting possibilities for selection as annotation. The first explanation method is a set of bar charts which compares the distribution of the attribute values in the selection against the distribution of the attribute values in the overall dataset. These charts do not require statistical expertise for interpretation. "Explanations" are read off by comparing the heights of the bars. A large discrepancy between an attribute's values in the selection and its values in the overall dataset indicates that the presence or absence of that value is highly correlated with the time series marked "interesting." An example can be seen in Fig. 4.

The second explanation method demonstrates that selection-as-annotation supports any supervised learning algorithm. In general, the problem of supervised learning can be formalised as the process of discovering a hypothesis $h : X^n \to Y$, given a sequence of training examples $(\vec{x}_i, y_i)$. The intention is that the learnt hypothesis achieves a level of generality that renders it useful for modelling and prediction purposes. Here, each $\vec{x}_i$ is known as the *feature vector* and each $y_i$ is known as the *label* or *class*.

We implemented the ID3 decision tree algorithm [37] as it produces human-interpretable models in the form of rules. For each time series, our feature vector is the attribute vector of the series: $(a_1, a_2, ..., a_n)$, and our label is a binary value indicating whether the time series was part of the selection, that is, was marked as "interesting":

$$\textit{label}(T^{(k)}) = \begin{cases} \textit{Interesting}, & T^{(k)} \in \textit{Selection} \\ \textit{Not Interesting}, & T^{(k)} \notin \textit{Selection} \end{cases}$$

Subsequently our training dataset $D$ consists of $(n+1)$-tuples of the form $(a_1, ..., a_n, \textit{label}(T^{(k)}))$. The ID3 algorithm can now be called on $D$, specifying $(\forall j.A_j)$ as the attributes and *label* as the target attribute (class). We map the resulting data structure directly onto a tree visualisation. Explanations are read as a conjunction of nodes from root to leaf. The tree is interactive, featuring collapsible nodes, panning and zooming.

A key advantage of deploying the ID3 algorithm in this manner is that this method of mining explanations scales to arbitrarily large attribute-value spaces. The tree visualisation can be used to display precise combinations of explanatory attributes, using exactly the tree-depth (i.e., number of relevant attribute values) required. For instance, if a single attribute value contains complete discriminatory information about the user selection, the tree stops expanding at that attribute value. An example tree can be seen in Figure 5. The figure shows only one path (blue nodes are collapsed), but when completely uncollapsed (i.e., showing all paths), the full tree has only 100 nodes. For just the 8 attributes in our example dataset, there are over $10^5$ attribute-value combinations. The tree, constructed on an information-theoretic basis, represents only the most relevant ones; thus, it scales to the BT fault dataset with hundreds of attributes.

## IV. Comparative study

We conducted a user study to answer the following research questions. With respect to a current industry-standard analysis tool, does Gatherminer:

- result in more interesting features (patterns) found?
- result in more correct explanations being found?
- improve users' confidence in their analyses?

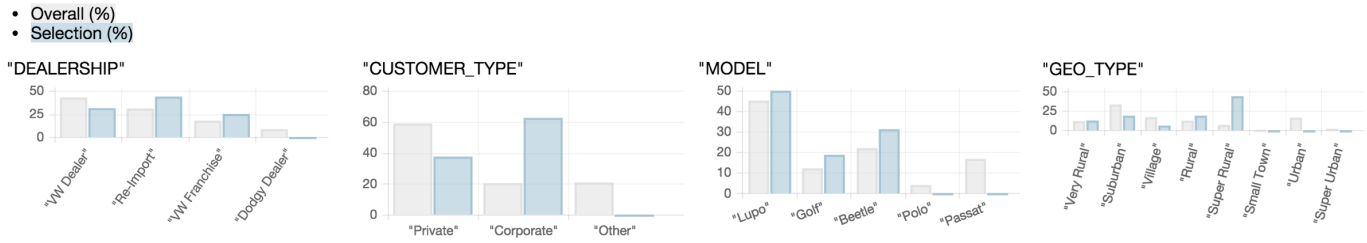


Fig. 4. Some explanatory charts for the high-valued clusters seen in Fig. 2. We see that ∼60% of the series in the selection have CUSTOMER_TYPE=Corporate, but that value only occurs in 20% of series overall. Thus, the attribute-value rule CUSTOMER_TYPE=Corporate potentially partially explains the behaviour of the selected time series. Similarly, MODEL≠Polo and MODEL≠Passat and GEO_TYPE=Super Rural are also potential explanations. Hovering on bars reports exact percentages in tooltips.

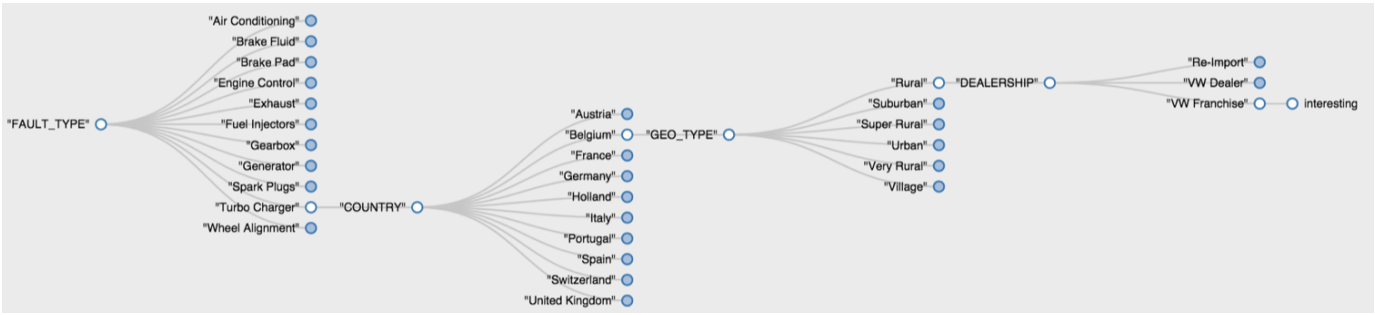


Fig. 5. In the ID3 tree, explanations are read as a conjunction of nodes on a path from root to leaf. One such path is shown above: when (FAULT_TYPE=Turbo Charger) ∧ (COUNTRY=Belgium) ∧ (GEO_TYPE=Rural) ∧ (DEALERSHIP=VW Franchise), the time series is likely to be interesting (i.e., in the selection).

*a) Recruitment:* Six participants were recruited from statistical analysis groups within BT Research (Adastral Park, UK). Participants were experienced professional analysts who regularly study the BT network data using Tableau. We chose Tableau [1] as the visual analytics tool against which to make comparisons, as this was most representative of our expert participants' typical workflows. A generic tool like Tableau is the only viable option in industry, since no tool tailored to this problem is available. Each participant had extensive prior experience of using Tableau, and no prior exposure to Gatherminer. The experiment was conducted in the participants' own office environments.

*b) Tasks:* Each participant completed 5 matched pairs of tasks (10 tasks in total). The first task of each pair was completed using Tableau, and the second using Gatherminer, allowing for within-subject comparisons. Task order was randomised between participants to account for order effects. For each task, participants were given a time series dataset of 500 time series, each of length 200. Each time series had six attributes: $A, B, C, D, E, F$. Each attribute had six values, $A = \{A1, A2, A3, A4, A5, A6\}, B = \{B1, ..., B6\}$, and so on. For our experimental tasks, each *not interesting* time series consisted of random integers from the uniform distribution between 1 and 100. Each *interesting* time series contained a segment where the distribution is heavily weighted towards 1 or 100, that is, an upward or a downward spike (e.g., Fig. 6). Each interesting feature was synthesised to occur when a series had a unique corresponding attribute value (e.g., in one task, all series with $A = A2$ contain an upward spike). The dataset for task pair #1 was synthesised to have 2 interesting features.

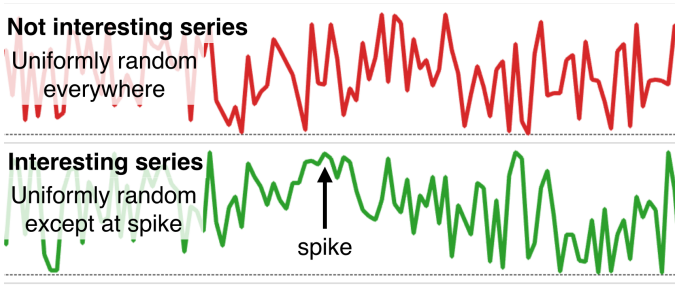


Fig. 6. Example of individual "not interesting" and "interesting" time series.

Task pairs #2 and #3 had 3 interesting features each, and task pairs #4 and #5 had 4 interesting features each.

Note that while the design of our tool was informed by and aimed towards real-world data (as in our examples), for the purposes of the experimental task we deliberately chose to synthesise domain-independent data. This is because the reliance of analysts on their domain expertise is so strong that it acts as a confound and prevents meaningful comparisons of the intrinsic benefits of various visualisation systems. This is further discussed in the next section.

Participants were requested to "find and explain as many interesting features" of the time series as they could. Additionally, participants were requested to rate their confidence about their performance after each task, using a 10-point scale in the format of the validated Computer Self-Efficacy inventory [38]. Specifically, they rated themselves on a scale of 1-10 with respect to the following two questions: (1) "How confident

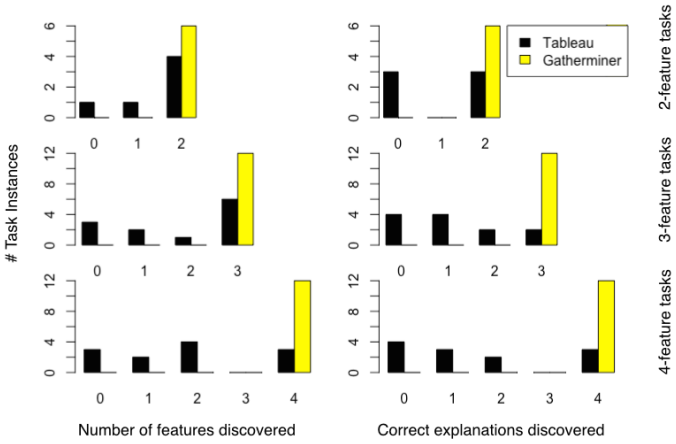


Fig. 7. Comparative histograms of discovered feature and explanation counts when using Gatherminer and Tableau. Observe the wide spread of completeness with Tableau.

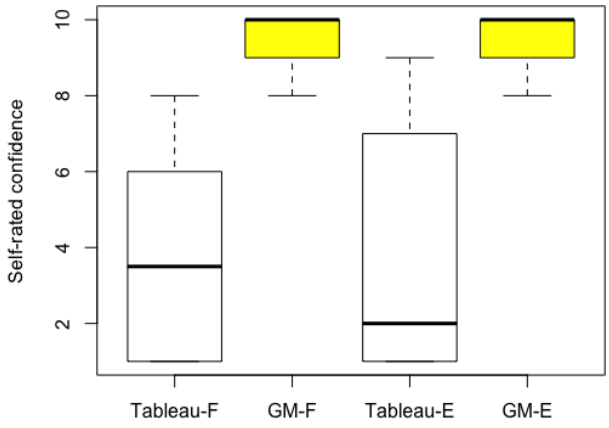


Fig. 8. Boxplots of self-reported confidence scores for feature discovery (F) and explanation discovery (E), comparing Gatherminer (GM) vs Tableau.

are you that you found all the interesting features?", and (2) "How confident are you that you found plausible explanations for the interesting features you found?" Participants were not made aware beforehand of the nature or number of interesting features in any task. Participants' remarks were also recorded; these are discussed in the next section.

### A. Experimental results

In general, our data was paired, not normally distributed, and had equal sample sizes for all conditions, so comparisons were drawn using the Wilcoxon signed rank test (WSRT).

*1) Task completeness:* Participants found significantly more interesting features with Gatherminer than with Tableau (WSRT: $V = 171, p = 1.7 \cdot 10^{-4}$); the effect size is a median discovery of an additional 50% of features with Gatherminer. Similarly, with Gatherminer they found significantly more correct explanations for those features (WSRT: $V = 276, p = 2.2 \cdot 10^{-5}$); a median of an additional 66.7% correct explanations were discovered with Gatherminer. This is illustrated in Fig. 7.

*2) Discovery times:* With Gatherminer, participants took significantly less time to discover features (WSRT: $V = 1176, p = 1.7 \cdot 10^{-9}$); the effect size is a median improvement of 110.5s using Gatherminer. They also took significantly less time to discover correct explanations (WSRT: $V = 654, p = 4.8 \cdot 10^{-7}$); a median improvement of 181.5s. This improvement is not altogether surprising, since Tableau is a much more general-purpose tool.

*3) Confidence:* Post-task, participants were significantly more confident that they had indeed discovered all major interesting features using Gatherminer than using Tableau (WSRT: $V = 465, p = 1.7 \cdot 10^{-6}$); the effect size is a median increase of 6.5. Similarly, they were more confident that they had discovered plausible explanations for all of the discovered features while using Gatherminer (WSRT: $V = 465, p = 1.6 \cdot 10^{-6}$); a strong median increase of 8. This is illustrated in Fig. 8.

## V. Discussion

### A. Analysis strategies

Gatherminer is a heavily specialised tool which emphasises certain types of analysis over others. Tableau, on the other hand, is a much more general-purpose analysis tool, facilitating many strategies for solving these tasks. It may appear an unfair comparison, but this is mitigated due to the fact that our participants were *expert* users experienced in performing precisely this type of statistical analysis using Tableau, which suggested a natural starting point for evaluation. In this section we report some observations regarding the strategies our expert participants employed to analyse data in Tableau, and discuss how Gatherminer's design improves upon these.

*1) Successful strategies in Tableau:* Successful strategies relied on finding levels of aggregation that generate visualisations with a manageable level of complexity, whilst simultaneously revealing interesting features. However, these strategies still fell back onto manually iterating over each attribute in turn, and this resulted in participants feeling less confident about their analysis (more detail in §V-B). One such strategy was to observe aggregate line charts of each individual attribute-value pairing (e.g., one line chart summing all series where $A = A2$). Here, any attribute-value that caused spikes or dips was clearly reflected. Since our tasks consisted only of 6 attributes, each with 6 values, and each feature only involved one attribute at a time, it was possible to apply these strategies effectively. However, in practice, with many more attributes and values, these strategies quickly become intractable. In contrast, since Gatherminer shows the completely disaggregated time series, it is possible for the user to view interesting features across all values of all attributes simultaneously. The analysis of interesting features drives the discovery of correlated attributes, not vice versa.

*2) Unsuccessful strategies in Tableau:* Unsuccessful strategies generally stemmed either from viewing data in completely disaggregated form (e.g., one line chart for each of the 500 series), which lead to unmanageable complexity in the visualisation, or aggregating the data too much (e.g., one line chart

that summed over all 500 series), which resulted in features going completely undetected. In Gatherminer, the data is also completely disaggregated, but the compactness of the colour-mapped matrix display, combined with automated reordering, makes the complexity of the visualisation manageable.

Another frequent issue was the discovery of false correlations. A common strategy was to take a few examples of time series with interesting features and inspect their attributes. If these series had more than one attribute value in common, the analysts were likely to conclude that the conjunction of those values together produced the effect, whereas in reality it may have just been one of the attributes, and the spurious correlation of the other attribute was simply a consequence of the small sample size. In Gatherminer, since series with interesting features are grouped together, it is trivial for the analyst to select large sets of series with shared behaviour to inspect the overall properties of their attributes.

### B. Confidence

It is important for analytical tools to enable analysts to have confidence in their analyses. In this regard, one major strength of colour-mapped matrices is that they can provide a lossless, exhaustive overview of the data; this satisfies analysts' desire to "leave no stone unturned." In particular, even the analysts who developed successful strategies in Tableau recognised that the manual nature of their strategy was not scalable, remarking: "*You've got too many dimensions to visualise simultaneously*"; "*Maybe I should just focus on one attribute for a start*"; "*I'm going to scroll through this list, and when I see one...*"; "*I feel like I'm missing a lot if I do it manually*". Remarks regarding the confidence of their analysis in Gatherminer include: "*I can explore all of it. I don't have to drill down.*"; "*Am I confident I have discovered all the features? Yes, of course, I have seen it.*"

### C. Value of the gathering process

Gatherminer strongly encourages partially-automated analysis. On almost every occasion, while using Gatherminer, participants first deployed the "Gather" function before doing anything else. While using Tableau, participants often mentioned dissatisfaction with the limitations of the (nonetheless sophisticated) built-in sorting functionality: "*If I can some way get a cluster*"; "*What I want is interesting features grouped together*"; "*I want to see groups of lines that are behaving [similarly] because then I can see which of these variables is impacting the series*." Remarks from our participants regarding Gatherminer's reordering function include: "*It's nice to have this hybrid approach where you get [the reordering] automatically and then the analyst can also scan it manually to see what is going on*"; "*At first the [colour-mapped matrix] view itself is helpful because you understand that there is something going on. The clustering then makes it very evident*."

### D. Role of domain expertise

Our tasks deliberately used meaningless codes for attributes and values in order to separate the utility of our tool from the domain expertise of the participant. Had we used network fault data, then the relative experience of the participant in that domain could potentially have impacted their ability to effectively analyse the data, independently of the analytical tool. Domain expertise provides a variety of prior expectations regarding what types of features might be present (e.g., linear trends, peaks, troughs, periodic functions), and what attributes might be of explanatory value – providing an efficient order of consideration for brute force attribute value checking. Note that these prior expectations may not necessarily be beneficial, as they may lead to the discovery of spurious correlations, or overlooking attributes not expected to be related.

Our hypothesis about the latent confounding power of domain expertise was further substantiated by comments made by our expert participants whilst analysing the data in Tableau. One participant said: "*If this was data I knew about, then I'd have some idea of where to start. Here, I'm lost.*" Other remarks include: "*Part of that [difficulty experienced with experimental tasks] is that I have no sense of the features*"; "*Doing data analysis when you have no idea of the data is quite unusual*"; "*Given that I have no idea of the attributes, I have to ignore them.*"

These comments illustrate how domain expertise actually plays a significant role in the analyst's heuristic approach to discovering explanations of interesting features. Thus, controlled usability experiments designed with real-world data in order to preserve external validity may have the opposite effect; the participants' use of domain expertise may confound any meaningful comparison between visualisation systems.

## VI. Conclusion

We have presented a visual language approach to an industrially important class of analytical tasks involving the study of time series, where the shape of interesting patterns is not known beforehand, and the space of explanatory attributes is large. The Gatherminer tool employs a novel combination of colour-mapped and reorderable matrices, adding a visual language layer for exploratory construction of statistical models based on patterns observed in the reordered matrix. We have evaluated our design in a user study which demonstrates that for the aforementioned class of tasks, Gatherminer results in significantly faster and more complete analyses of time series datasets, and significantly improves analysts' confidence.

## VII. Acknowledgements

Advait is supported by an EPSRC+BT iCASE award and a Cambridge Computer Laboratory Robert Sansom scholarship.

## References

[1] "Business Intelligence and Analytics — Tableau Software," http://www.tableau.com/, accessed: June 30, 2016.

[2] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pp. 111–117, 2005.

[3] P. Pirolli and S. K. Card, "The Sensemaking Process and Leverage Points for Analyst Technology as Identified Through Cognitive Task Analysis," *Proceedings of International Conference on Intelligence Analysis*, vol. 5, pp. 2–4, 2005.

[4] W. Mullër and H. Schumann, "Visualization methods for time-dependent data-an overview," in *Simulation Conference, 2003. Proceedings of the 2003 Winter*, vol. 1. IEEE, 2003, pp. 737–745.

[5] M. Hao, U. Dayal, D. A. Keim, and T. Schreck, "Multi-resolution techniques for visual exploration of large time-series data," *Eurographics/IEEE VGTC Symposium on Visualization (EuroVis)*, pp. 27–34, 2007.

[6] A. Lex, M. Streit, E. Kruijff, and D. Schmalstieg, "Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context," *Pacific Visualization Symposium (PacificVis), 2010 IEEE*, pp. 57–64, 2010.

[7] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system." in *ISMIR*, vol. 2002, 2002, pp. 107–115.

[8] J. Bertin, *Graphics and graphic information processing*. Walter de Gruyter, 1981.

[9] T. W. Liao, "Clustering of time series data – a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857 – 1874, 2005.

[10] N. Elmqvist, T.-N. Do, H. Goodell, N. Henry, and J. Fekete, "Zame: Interactive large-scale graph visualization," in *Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific*. IEEE, 2008, pp. 215–222.

[11] F. Mansmann, D. Spretke, H. Janetzko, B. Kranstauber, and K. Safi, *Correlation-based Arrangement of Time Series for Movement Analysis in Behavioural Ecology*. Bibliothek der Universität Konstanz, 2012.

[12] C. Perin, P. Dragicevic, and J.-D. Fekete, "Revisiting bertin matrices: New interactions for crafting tabular visualizations," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 20, no. 12, pp. 2082–2091, Dec 2014.

[13] M. C. Hao, M. Marwah, H. Janetzko, U. Dayal, D. A. Keim, D. Patnaik, N. Ramakrishnan, and R. K. Sharma, "Visual exploration of frequent patterns in multivariate time series," *Information Visualization*, vol. 11, no. 1, pp. 71–83, 2012.

[14] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 493–498.

[15] J. Bernard, T. Ruppert, M. Scherer, T. Schreck, and J. Kohlhammer, "Guided discovery of interesting relationships between time series clusters and metadata properties," in *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*. ACM, 2012, p. 22.

[16] J. Bernard, D. Daberkow, D. Fellner, K. Fischer, O. Koepler, J. Kohlhammer, M. Runnwerth, T. Ruppert, T. Schreck, and I. Sens, "Visinfo: a digital library system for time series research data based on exploratory search–a user-centered design approach," *International Journal on Digital Libraries*, pp. 1–23, 2014.

[17] P. Buono, A. Aris, C. Plaisant, A. Khella, and B. Shneiderman, "Interactive pattern search in time series," in *Electronic Imaging 2005*. International Society for Optics and Photonics, 2005, pp. 175–186.

[18] J. Lin, E. Keogh, and S. Lonardi, "Visualizing and discovering non-trivial patterns in large time series databases," *Information visualization*, vol. 4, no. 2, pp. 61–82, 2005.

[19] R. Kincaid and H. Lam, "Line graph explorer: scalable display of line graphs using Focus+Context," in *Proceedings of the working conference on Advanced visual interfaces - AVI '06*. New York, New York, USA: ACM Press, 2006, p. 404.

[20] R. Rao and S. K. Card, "The table lens," in *Proceedings of the SIGCHI conference on Human factors in computing systems celebrating interdependence - CHI '94*. New York, New York, USA: ACM Press, 1994, pp. 318–322.

[21] D. A. Keim, P. Bak, E. Bertini, D. Oelke, D. Spretke, and H. Ziegler, "Advanced visual analytics interfaces," *Proceedings of the International Conference on Advanced Visual Interfaces - AVI '10*, p. 3, 2010.

[22] M. C. Hao, U. Dayal, D. A. Keim, D. Morent, and J. Schneidewind, "Intelligent visual analytics queries," *VAST IEEE Symposium on Visual Analytics Science and Technology 2007, Proceedings*, no. Vast 2007, pp. 91–98, 2007.

[23] D. Fisher, I. Popov, S. Drucker *et al.*, "Trust me, i'm partially right: incremental visualization lets analysts explore large datasets faster," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 1673–1682.

[24] J. Bernard, D. Sessler, M. Behrisch, M. Hutter, T. Schreck, and J. Kohlhammer, "Towards a User-Defined Visual-Interactive Definition of Similarity Functions for Mixed Data," in *Poster at Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, Nov. 2014.

[25] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang, "Dis-function: Learning distance functions interactively," *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 83–92, Oct. 2012.

[26] G. M. Mamani, F. M. Fatore, L. G. Nonato, and F. V. Paulovich, "User-driven feature space transformation," in *Computer Graphics Forum*, vol. 32, no. 3pt3. Wiley Online Library, 2013, pp. 291–299.

[27] M. Sedlmair, C. Heinzl, S. Bruckner, H. Piringer, and T. Moller, "Visual parameter space analysis: A conceptual framework," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 20, no. 12, pp. 2161–2170, Dec 2014.

[28] A. Endert, C. Han, D. Maiti, L. House, S. Leman, and C. North, "Observation-level interaction with statistical models for visual analytics," in *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, Oct 2011, pp. 121–130.

[29] J. A. Fails and D. R. Olsen Jr, "Interactive machine learning," in *Proc. 8th Int'l Conf. on Intelligent user interfaces*. ACM, 2003, pp. 39–45.

[30] E. Wu and S. Madden, "Scorpion: Explaining away outliers in aggregate queries," *Proc. VLDB Endow.*, vol. 6, no. 8, pp. 553–564, Jun. 2013.

[31] M. Behrisch, F. Korkmaz, L. Shao, and T. Schreck, "Feedback-driven interactive exploration of large multidimensional data supported by visual classifier," in *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, Oct 2014, pp. 43–52.

[32] P. Pirolli, P. Schank, M. Hearst, and C. Diehl, "Scatter/gather browsing communicates the topic structure of a very large text collection," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1996, pp. 213–220.

[33] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[34] A. Cockburn, A. Karlson, and B. B. Bederson, "A review of overview+detail, zooming, and focus+context interfaces," *ACM Comput. Surv.*, vol. 41, no. 1, pp. 2:1–2:31, Jan. 2009.

[35] G. Gutin, A. Yeo, and A. Zverovich, "Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the tsp," *Discrete Applied Mathematics*, vol. 117, no. 1, pp. 81–86, 2002.

[36] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.

[37] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[38] D. R. Compeau and C. A. Higgins, "Computer self-efficacy: Development of a measure and initial test," *MIS quarterly*, pp. 189–211, 1995.