ReDSOM: Relative Density Visualization of Temporal Changes in Cluster Structures using Self-Organizing Maps

Denny
Department of Computer Science, The Australian National University, Canberra, Australia
Faculty of Computer Science, University of Indonesia, Indonesia
denny@cs.anu.edu.au, denny@cs.ui.ac.id

Graham J. Williams
Australian Taxation Office, Canberra, Australia
graham.williams@ato.gov.au

Peter Christen
Department of Computer Science, The Australian National University, Canberra, Australia
peter.christen@anu.edu.au

Abstract

We introduce a Self-Organizing Map (SOM) based visualization method that compares cluster structures in temporal datasets using Relative Density SOM (ReDSOM) visualization. Our method, combined with a distance matrix-based visualization, is capable of visually identifying emerging clusters, disappearing clusters, enlarging clusters, contracting clusters, the shifting of cluster centroids, and changes in cluster density. For example, when a region in a SOM becomes significantly more dense compared to an earlier SOM, and well separated from other regions, then the new region can be said to represent a new cluster. The capabilities of ReDSOM are demonstrated using synthetic datasets, as well as real-life datasets from the World Bank and the Australian Taxation Office. The results on the real-life datasets demonstrate that changes identified interactively can be related to actual changes. The identification of such cluster changes is important in many contexts, including the exploration of changes in population behavior in the context of compliance and fraud in taxation.

1. Introduction

Businesses and government organizations need knowledge of change in order to adapt their strategies to ever-changing environments. Knowing what has changed can be a major competitive advantage for an organization. To understand what has changed, analysts have to be able to relate new knowledge or models acquired from a newer dataset to those acquired from an earlier dataset. Without this context, it can be difficult to revise existing strategies. This is particularly problematic if an organization has already implemented a strategy based on an earlier model.

In supervised learning, classifier performance often degrades over time, an issue known as concept drift [19, 23]. In many real-life domains, a concept of interest may depend on some *hidden context*, which is not given explicitly in the form of predictive features (i.e. some variables are invisible to the learner). For example, such hidden concepts can be changes in economic policy, disasters, life events, or changes in marketing strategies. Changes in the hidden context can induce more or less radical changes in a target concept. Most research in concept drift only addresses concept drift in a supervised learning context—little has been researched in the context of unsupervised learning.

In data mining of conceptual changes, a number of temporal data mining algorithms have focused on detecting the point in time when something has changed (change detection), rather than understanding or exploring the causes that have made the changes (change analysis). For example, by gradually eliminating the effects of past data, an on-line discounting learning algorithm can detect outliers and the change points in time in a changing data source [25].

To discover changes between two datasets, the resulting data mining models can be compared, given that a data mining model is designed to capture specific characteristics of a dataset. A theoretical framework has been introduced in [7] that allows measuring changes between two models. In this

framework, when the structural components of the models are different, both structures are 'extended' to a greatest common refinement. The deviation between two models is then calculated by aggregating the differences in the measurement components of the models.

This paper focuses on visualizing, identifying, and analyzing changes in cluster structures in a SOM, trained with a newer dataset, compared to a SOM trained with an older dataset. Temporal cluster analysis can be useful to understand changes in datasets, or to review the effectiveness of deployed strategies. For example, if an organization has devised a marketing strategy based on a clustering of the past year's customer data, it is important to know if the current year's clustering differs, in order to understand changes in customer behavior, or to review the effectiveness of the implemented marketing strategy. New strategies can then be devised to encourage or deter the development of new clusters or to slow the demise of clusters, as suits the requirements of the business.

ReDSOM visualization allows users to explore the distinctive features of changes interactively using the hot-spot methodology [6]. Involving the user in the data exploration process is important in ensuring effective data analysis [12].

We propose the use of SOMs for effectively visualizing changes. SOMs have several advantages in temporal cluster analysis. They are able to relate clustering results by linking multiple visualizations, and they can detect various types of cluster changes, including emerging and disappearing clusters [5]. Furthermore, SOMs create a smaller but representative dataset, and they have topology preservation properties [15]. Importantly, SOMs can be used to explore high-dimensional data spaces through a non-linear projection onto a two-dimensional (2-D) plane using visualizations that are easy to understand even by non-analysts [15]. A mathematical analysis of SOM properties can be found in [17]. Applications of SOM for data mining are found in engineering, speech analysis and recognition, finance, and information retrieval [4, 15].

We contrast our ReDSOM visualization methodology to the goals of time series clustering [13]. Time-series clustering aims to cluster entities that have similar time-series patterns, whereas our method clusters entities at points in time (snapshots), and compares the clustering structures of such snapshots.

Research in data stream mining has also provided some insights into the problem described here. Aggarwal et.al. [2] presented a framework for clustering data streams where the aim is to discover changes in the evolving data streams. The basic idea of the approach is to divide clustering into an online and off-line component, with the online component periodically storing summary information of the data stream clusters (so called micro-clusters), and the off-line component generating aggregated clusters according to the needs of an analyst. Our work does not consider data streams but investigates snapshot datasets that were collected at different points in time. While data streams are becoming common in many application areas, static snapshot datasets are still the most commonly used type of data in many organizations. There still remains a need to analyze changes between such snapshots.

The main contribution of this paper is the development of Relative Density SOM (ReDSOM) visualization that can compare and contrast changes in cluster structures in temporal datasets. ReDSOM allows analysts to explore and understand changes interactively. We evaluate our method on synthetic datasets and on real-life datasets published by the World Bank [24]. We have also evaluated our method on large real-life datasets from the Australian Taxation Office. The results on the real-life datasets demonstrate that changes identified interactively can be related to actual real-life changes.

The remainder of the paper is organized as follows. The next section discusses related work in temporal cluster analysis. An overview of Self-Organizing Maps is provided in Section 3. Section 4 introduces our relative density definition and ReDSOM visualization. The results of our experiments are then discussed in Section 5, and conclusions and future work are provided in Section 6.

2. Related Work

Temporal data can be grouped into four broad categories: static, sequences, time-stamped, and fully temporal [18]. In static datasets, temporal context is not included and cannot be inferred. Sequences are basically ordered lists of events, but not time-stamped. Examples of time-stamped datasets are census data, web-based activity, or sales transaction. In fully temporal databases, each tuple in a time-varying relation in the database may have one or more dimensions of time, such as age and treatment time. Our method analyzes changes in two or more static temporal datasets.

Chakrabarti, Kumar, and Tomkins [3] defined evolutionary clustering as the problem of processing time-stamped data to produce a sequence of clusterings; that is, a clustering for each time step. This framework tries to optimize two potentially conflicting criteria: remaining faithful to the current data, and not shifting dramatically from the previous clustering results. Therefore, the user has to define a snapshot quality function $sq(C_t, D_t)$ which measures the quality of a clustering result C_t for dataset D_t at time t, and a history cost function $hc(C_{t-1}, C_t)$ which measures how much a latter clustering result C_t differs from the previous one, C_{t-1}. The optimal cluster sequence can, therefore, be found by determining at each time step t a clustering C_t that optimizes the incremental quality $sq(C_t, D_t) - cp \cdot hc(C_{t-1}, C_t)$, where cp is a non-negative

change parameter. As cp is increased, more weight is placed on matching the historical clusters. Based on this, the authors derived an agglomerative hierarchical and a k-means clustering algorithm. When calculating clustering result C_t using k-means, for example, the previous cluster centroids of C_{t-1} are used as the starting seeds. The new centroids are then calculated based on the closest match of cluster centroids in the previous clustering result, and the cluster centroids of the non-evolutionary (conventional) k-means.

This framework is able to find a balance between remaining faithful and not shifting dramatically in training the subsequent clustering results, in order to smooth the clustering sequence. However, the aim of this framework is not to analyze the changes of the clustering results. It is not easy to understand the actual cluster changes using plots of snapshot quality and historical cost over time. It is not clear how to relate and understand changes in terms of the earlier clustering results. Furthermore, the capability of the framework to detect any rate of changes (abrupt or gradual changes) is questioned as the change parameter cp is constant over time and has to be defined beforehand.

The k-means version of the framework also has some issues related to the discovery of new or lost clusters. With the proposed k-means variant, the previous cluster centroids C_{t-1} are used as starting seeds. A problem arises when there is a larger number of cluster (k) selected in finding the latter clustering result C_t. It is not clear how to initialize the additional cluster centroids in this case. A similar problem occurs when a smaller k is selected. In this case, one or more cluster centroids from the previous clustering result C_{t-1} have to be removed in finding the latter clustering result C_t. Our method, on the other hand, is able to show emerging clusters and lost clusters without having to determine the number of clusters beforehand.

Hido et.al. [9] proposed an approach to explain changes between two datasets by using a decision tree, and labeling one dataset as negative and the other as positive. The trained model is then investigated to understand differences between the datasets. As decision trees divide the data space into hypercubes and try to separate the entities based on their labels, this approach can detect when there is a new hypercube that was more sparsely occupied by the other dataset. However, this method cannot show the separation between the hypercubes and the density of these hypercubes. Therefore, this method cannot differentiate between emerging clusters and cluster enlargements. In our method, on the other hand, separation between prototype vectors can be shown using distance matrix visualization [10]. Furthermore, when correlated attributes exist in the dataset, only one of them will be used to describe new or lost hypercubes, reducing the description of the hypercubes.

Recently, Adomavicius and Bockstedt [1] introduced a graph visualization technique for exploring trends in multi-attribute temporal datasets using a temporal cluster graph. In this technique, transactional dataset D is partitioned into data subsets D_t according to time periods. Each data subset is then clustered. Only clusters that have at least $\alpha|D_t|$ number of entities are shown as nodes, where $\alpha \in [0, 1]$ is a node filter parameter. Nodes between two adjacent time periods are connected with an edge, if the distance between the nodes is less than a threshold η, which is calculated based on an edge filter parameter β and the average between-cluster distances. This graph is visualized interactively, where users have to experiment with the number of clusters for each partition D_t, node filter parameter α, and edge filter parameter β. In our method, users do not have to experiment to find the optimal number of clusters.

For detecting changes between two SOMs, Kaski and Lagus [11] proposed a dissimilarity measure for two maps, which is calculated based on the expected value of distances between pairs of representative data points on both maps. This approach can determine how much two SOMs differ, but it cannot analyze the changes.

Lingras et.al. [16] studied the temporal cluster characteristics of supermarket customers using an interval set based-SOM. The capability of the proposed method to detect new clusters is questioned as the number of clusters was constant for all time periods in their experiments.

Denny and Squire [5] proposed a SOM training method and SOM-based visualization techniques that are capable of explaining the clustering results of a second dataset in terms of the clustering result of a first dataset by using color and position linking. These visualization techniques can relate two clustering results which can show the following structural changes: changes in cluster size, centroid movements, new clusters, cluster splitting, missing clusters, and cluster merging. Such changes have to be analyzed visually by selecting a number of clusters for both datasets. However, these techniques cannot differentiate between cluster enlargement (occupying new space), and increase in cluster density (more entities in the same space).

3. Self-Organizing Maps

A SOM is an artificial neural network that performs unsupervised competitive learning [14]. Artificial neurons are arranged on a low-dimensional grid, commonly a 2-D plane with n_r rows, n_c columns, and a total number of $n_{unit} = n_r \cdot n_c$ units. Each neuron j has an d-dimensional prototype vector, m_j, where d is the dimensionality of dataset D. Each neuron is connected to neighboring neurons, with distances to its neighbours being equidistant in the map space. The lattice structure can be a hexagonal grid, where each neuron is connected to six neighbours. Larger maps generally have higher accuracy and generalization capability [22], but they also have higher computation costs.

Before training a map, the prototype vectors should be initialized using random initial values, or ordered values (linear initialization) [15]. Linear initialization uses ordered values of component vectors based on the first two largest principal components. When using random initialization, the radius of the neighbourhood function should be large enough; otherwise the map will not be globally ordered. However, if linear initialization is used for the initial map, then a smaller radius and a shorter training length could be used [15]. Therefore, linear initialization is preferred over random initialization, because it can speed up the learning process by orders of magnitude [15].

At each training step t, the best matching unit b_i (BMU) for training data vector x_i, i.e. the prototype vector m_j closest to the training data vector x_i, is selected from the map according to Equation 1:

$$\forall j, \quad \|x_i - m_{b_i}(t)\| \leq \|x_i - m_j(t)\| \tag{1}$$

In the batch training algorithm [15], the values of new prototype vectors $m_j(t+1)$ are weighted averages of the training data vectors x_i, where the weight is the neighbourhood kernel value $h_{b_i j}$ centered on the best matching unit b_i. Since the neighbourhood function $h_{b_i,j}$ value is the same for all data vectors mapped to the same unit, the sum S_{V_j} of the Voronoi set V_j of each prototype vector m_j at training step t. So, each training data vector belonging to the Voronoi set of its closest prototype vector (BMU), can be calculated first using Equation 2:

$$S_{V_j}(t) = \sum_{x_i \in V_j} x_i. \tag{2}$$

Then, the prototype vectors are updated with:

$$m_j(t+1) = \frac{\sum_{i=1}^{n_{unit}} h_{ji}(t) \cdot S_{V_i}(t)}{\sum_{i=1}^{n_{unit}} h_{ji}(t) \cdot n_{V_i}}, \tag{3}$$

where h_{ji} is the neighbourhood kernel function centered on unit j (commonly Gaussian) [15], and n_{V_j} is the number of training data vectors in Voronoi set V_j. To handle missing values, S_{V_j} and n_{V_j} only perform summation and counting of non-missing components, respectively. This calculation of new prototype vectors takes into account neighboring prototype vectors which preserve the topological order.

The map is usually trained in two phases: a rough training phase and a fine tuning phase. The rough training phase usually has shorter training length and larger initial radius compared to fine tuning phase [15].

SOMs are popularly used in cluster analysis because they perform vector quantization and preserve topological order. Furthermore, the trained SOMs can be visualized using various methods that allow non-technical users to explore a dataset. Component plane visualizations can be used to show the spread of values of a certain component of all prototype vectors in a SOM [21]. Distance-matrix based visualizations, such as u-matrix visualization [10], show distances between neighboring nodes using a color scale representation on a map grid. This visualization can be used to identify borders between clusters, where long distances show highly dissimilar features between neighboring nodes that divide clusters, i.e. the dense parts of a map with similar features [10].

In [22], the prototype vectors of a trained SOM can be treated as 'proto-clusters' serving as an abstraction of the dataset. The prototype vectors are then clustered using a traditional clustering technique, such as k-means, to form the final clusters. In this two-level clustering, adding an extra layer simplifies the clustering task and reduces noise, but may yield higher distortion [22].

4. Relative Density

Let $D(\tau_1)$ be a dataset at time τ_1, and $D(\tau_2)$ be a dataset at time τ_2, where $\tau_1 < \tau_2$. In order to be able to compare two maps that are trained using datasets $D(\tau_1)$ and $D(\tau_2)$, the orientation of map $M(\tau_1)$ and $M(\tau_2)$ must be the same. Therefore, the following map training procedure, as proposed in [5], is used.

1. Normalize both datasets $D(\tau_1)$ and $D(\tau_2)$ using the same normalization method (e.g. z-score) and parameters (e.g. the same mean and standard deviation values) for the same attributes.
2. Initialize map $M(\tau_1)$ using ordered values.
3. Train map $M(\tau_1)$ using dataset $D(\tau_1)$.
4. Initialize map $M(\tau_2)$ using the prototype vectors of the trained map $M(\tau_1)$.
5. Train map $M(\tau_2)$ using dataset $D(\tau_2)$.

4.1. Relative Density Definition

As a SOM follows the distribution of a dataset it is trained on, more prototype vectors are allocated for dense regions, as shown in Figure 1. Therefore, area density at $m_j(\tau_1)$ on its own map, $M(\tau_1)$, might be different compared to area density at the same location on map $M(\tau_2)$. When the area density at the location of the prototype vector $m_j(\tau_1)$ in $D(\tau_2)$ is more sparse, the area density on map $M(\tau_2)$ is lower, compared to the area density at the same location on map $M(\tau_1)$, and vice-versa. In Figure 1, the area density at the marked area (the center of the Gaussian kernel contour) on the left plot is higher, compared to the area density at the same location on the right plot.

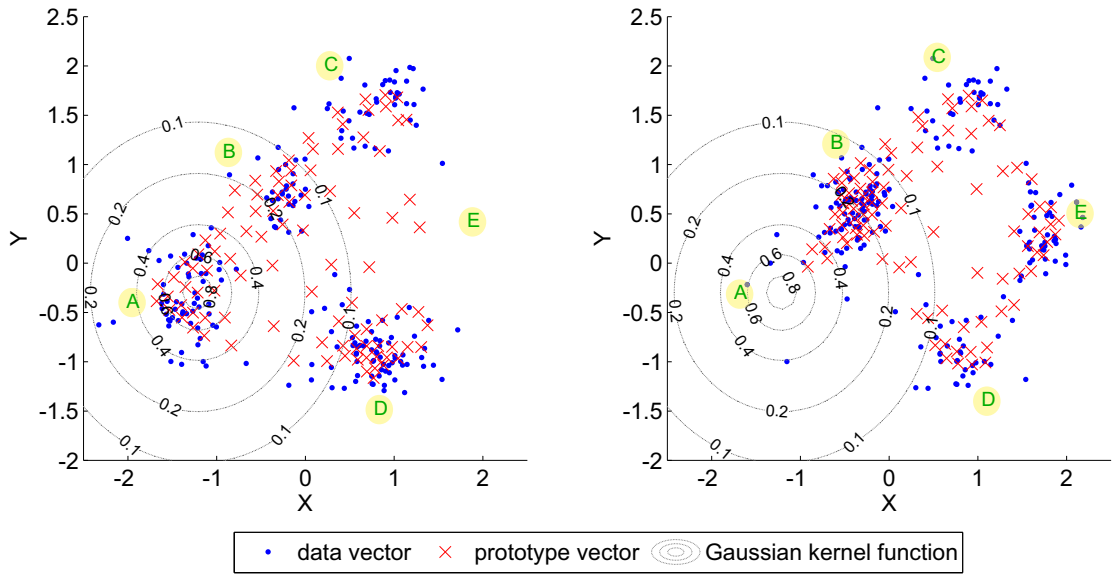

Figure 1. Plots of data vectors and prototype vectors of the maps trained using two synthetic datasets $D(\tau_1)$ (left) and $D(\tau_2)$ (right), where there is an emerging cluster ('E'), a lost cluster ('A'), a more dense cluster ('B'), a less dense cluster ('D'), and an unchanged cluster ('C'), in the dataset $D(\tau_2)$. The contour of the Gaussian kernel function centered on one of the cluster 'A' prototype vectors is shown on both plots.

We define area density $\rho_{M(\tau)}(v)$ at the location of a vector v on map $M(\tau)$ as the weighted sum of similar prototype vectors $m_j(\tau)$ on $M(\tau)$ centered on vector v, where the weight is calculated based on a Gaussian kernel function centered on vector v, as shown in Equation 4:

$$\rho_{M(\tau)}(v) = \sum_{j=1,\ldots,n_{unit}} exp\left(-\frac{\|v - m_j(\tau)\|}{2 \cdot r}\right) \quad (4)$$

As the radius r can be different for different maps (e.g. datasets with large attribute value ranges will need to have a larger radius), the radius should be determined based on "between neighbour distances" on the map. To adapt the radius for different maps, the quartile (e.g. third quartile) of these distances is used as the radius. As the area density is normalized into a relative density (Equation 5), this relative density is not sensitive to the radius. However, the radius should not be too small or too large. If the radius is too small, the relative density will be too sensitive to noise. On the other hand, if the radius is too large, then relative density cannot capture the details of changes.

We define relative density $\text{RD}_{M(\tau_2)/M(\tau_1)}(v)$ as the ratio of the area density at location of vector v on map $M(\tau_2)$ to the area density at the same location on the reference map $M(\tau_1)$, as shown in Equation 5:

$$\text{RD}_{M(\tau_2)/M(\tau_1)}(v) = log_2\left(\frac{\rho_{M(\tau_2)}(v)}{\rho_{M(\tau_1)}(v)}\right) \quad (5)$$

Without using a logarithm function in Equation 5, values between 0 and 1 are interpreted as becoming more sparse, the value of 1 is interpreted as no change, and values above 1 are interpreted as becoming more dense. Therefore, a base two logarithm is used to make it easier to interpret the ratio where positive values are interpreted as becoming more dense, negative values as becoming more sparse, and 0 is interpreted as no change. For example, a value of $+2$ is interpreted as the area centered at the location of vector v in the dataset $D(\tau_2)$ being four times more dense compared to the same area in the reference dataset $D(\tau_1)$.

Based on our observations, when the value of $\text{RD}_{M(\tau_2)/M(\tau_1)}(v)$ is less than -3, then the space at location of vector v is no longer occupied in the next map $M(\tau_2)$ (it is lost). Similarly, when $\text{RD}_{M(\tau_2)/M(\tau_1)}(v)$ is greater than $+3$, then the space at location of vector v was not occupied on the reference map $M(\tau_1)$. In other words, the space is only occupied on map $M(\tau_2)$.

Both Equations 4 and 5 are performed on all the prototype vectors of both maps, but not on the actual data vectors. Therefore, the running time of the calculation for a map is quadratic in the number of map units n_u, not in the number of data vectors $n_{D(\tau)}$, where $n_u \ll n_{D(\tau)}$.

4.2. Relative Density Visualization

As a shorthand, let $rd_1 \leftarrow \text{RD}_{M(\tau_2)/M(\tau_1)}(m_j(\tau_1))$ and $rd_2 \leftarrow \text{RD}_{M(\tau_2)/M(\tau_1)}(m_j(\tau_2))$.

To visualize the relative density of the locations of all prototype vectors $m_j(\tau_1)$, the values of rd_1 are visualized on a map $M(\tau_1)$ in a gradation of blue for positive values and in a gradation of red for negative values[1], as shown in the left map in Figure 2. Values over $+3$ are represented as dark blue, and values under -3 are represented as dark red. A value of 0 is represented as white, as it indicates no change in the density. For example, visualizations of the datasets and prototype vectors from Figure 1 can be seen in Figure 2.

However, rd_1 rarely returns values greater than +3, because all prototype vectors $m_j(\tau_1)$ are always present on the reference map $M(\tau_1)$ (prototype vector $m_j(\tau_1)$ is part of map $M(\tau_1)$). Therefore, $\rho_{M(\tau_1)}(m_j(\tau_1))$ at least returns 1, which makes the denomination part of Equation 5 larger compared to $\rho_{M(\tau_1)}(m_j(\tau_2))$. Therefore, values of rd_2 should be visualized on map $M(\tau_2)$ to detect emerging clusters, as shown in the right map in Figure 2. To detect new clusters, rd_2 is used because the area at the location of prototype vector $m_j(\tau_2)$ might be empty on map $M(\tau_1)$. Similarly, rd_2 cannot be used to detect lost clusters on map $M(\tau_2)$, because the empty space at the location of the prototype vector $m_j(\tau_1)$ is not represented on map $M(\tau_2)$, as $M(\tau_2)$ follows the distribution of dataset $D(\tau_2)$.

Because of SOM's vector quantization property and our definition of relative density, prototype vectors on map $M(\tau_1)$ that have negative rd_1 values will be less represented on map $M(\tau_2)$. For example, there are less prototype vectors in cluster 'D' in the right map in Figure 2. On the other hand, prototype vectors on map $M(\tau_1)$ that have positive rd_1 values will be more represented on map $M(\tau_2)$, as shown in cluster 'B' in the right map in Figure 2.

4.3. Analysis of Changed Regions

There are several types of possible structural changes between two datasets: new clusters, lost clusters, cluster enlargements, cluster contractions, shifting of centroids, and changes in cluster density. All these structural changes can be identified using ReDSOM and distance matrix visualizations. As mentioned before, distance-matrix based visualizations show distances between neighboring nodes using a color scale representation on the map, which can be used to identify cluster borders [10]. For example, in Figure 3, there are four clusters in both datasets (light yellow regions) separated by long distances.

[1]The SOM visualizations presented in this paper unfortunately require a diverging color scheme to illustrate the relative density that is hard to distinguish between positive and negative values using gray scale. We hope the reader has access to the color version, at least can view the PDF file.

Identifying new clusters. New clusters in dataset $D(\tau_2)$ can be identified by dark blue regions (values above +3) on relative density visualization rd_2 of map $M(\tau_2)$, and they have long distances at the border of the regions on map $M(\tau_2)$, for example cluster 'E', as shown in the right maps in Figures 2 and 3. When a new cluster appears inside the distribution of dataset $D(\tau_1)$, in other words between disjoint clusters, the values of rd_1 are close to +3, and the cluster is positioned in a sparse area (long distances in the distance matrix visualization between the clusters). This is due to interpolative units, which appear when the data clusters are disjoint [22], as can be seen in both plots in Figure 1. Therefore, the area density at this gap is higher, compared to the area density at the empty space outside the data distribution.

Identifying cluster enlargements, cluster contractions, movement of cluster centroids. When a new dense area emerges in dataset $D(\tau_2)$, but it does not have a good separation to its neighbour, the changes can be interpreted as cluster enlargements. This can be identified by dark blue regions on the relative density visualization rd_2, and the region has short distances at the border of the regions on map $M(\tau_2)$, as shown in the bottom-right corner of the right maps in Figure 4. When this kind of new region appears at the border of map $M(\tau_2)$, it can be said that the changes move towards the tail of the data distribution. Similarly, cluster contraction can be identified by a lost region, but it does not have a good separation to its neighbours. Movement of cluster centroids can be identified by simultaneous cluster enlargements and cluster contractions.

Identifying lost clusters. Lost clusters in dataset $D(\tau_1)$ can be identified by dark red regions (value below -3) on the relative density visualization rd_1 of map $M(\tau_1)$, and long distances at the border of the regions on map $M(\tau_1)$. An example is cluster 'A' in the left maps in Figures 2 and 3.

Identifying change of cluster density. An increase of cluster density in dataset $D(\tau_2)$ can be identified in the relative density visualization as light blue, for example cluster 'B' as shown in the left map in Figure 2. On the other hand, a decrease of cluster density in dataset $D(\tau_2)$ can be identified in the relative density visualization as light red, for example cluster 'D' as shown in the left map in Figure 2.

Analyzing interesting changes. Once a region of interest is selected interactively by a user, our hot spot methodology [6] can be used to understand distinctive features of these changing regions. In this methodology, the component planes are sorted by the importance of the attributes that distinguish the region from the rest of the population

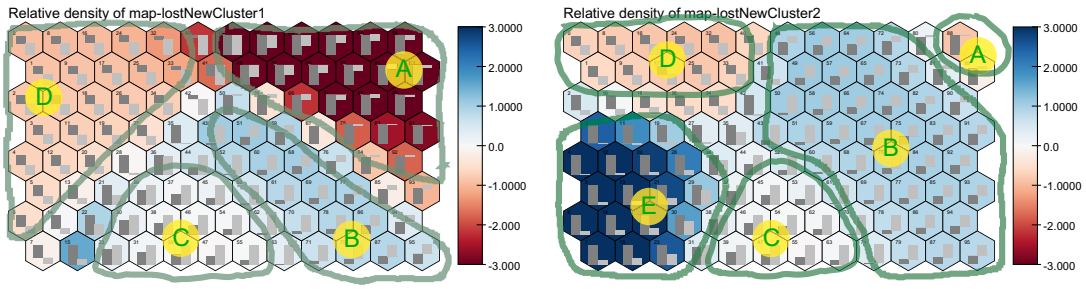

Figure 2. Relative density visualizations rd_1 (left) and rd_2 (right) of the datasets and maps shown in Figure 1. The bar chart inside each node shows the value of components of the prototype vectors (dark gray for component 'X' and light gray for component 'Y').

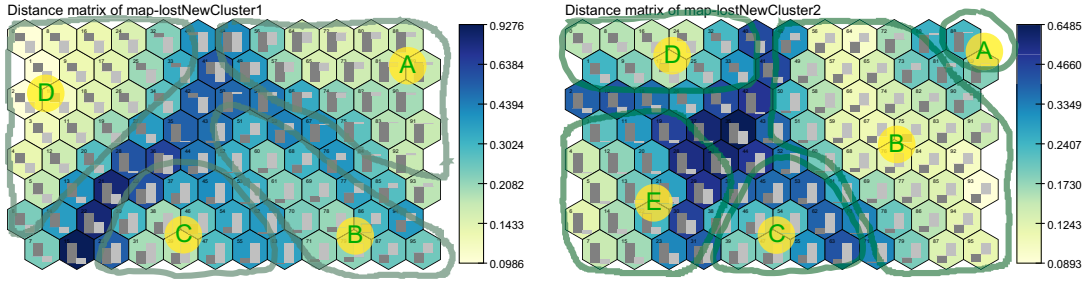

Figure 3. Distance matrix visualizations of the maps shown in Figure 1. These visualizations are linked by position to Figure 2, meaning that the same node in the left two maps refer to the same Voronoi region in the data space, and similarly in the right two maps.

using an attribute selection measure [8], such as information gain or gain ratio, as shown in Figures 5 and 7. As a SOM produces a smaller but representative dataset, the prototype vectors can be used as an approximation of the whole dataset. Efficient computation allows an analyst to explore distinctive features of any region of the map interactively.

5. Results and Discussion

Our method has been tested using our Java SOM Toolbox (JSOM) on both synthetic and real-life datasets. Synthetic datasets were used to evaluate the ability of the proposed method to visualize individual known cluster changes, such as the introduction of new clusters and disappearing clusters. Due to space limitations only one combined scenario has been presented in Section 4. The synthetic dataset $D(\tau_1)$ has been generated using Gaussian distributions, while the dataset $D(\tau_2)$ has been generated using a transition matrix $P = \{p_{ij}\}$ [8], which contains the probability of an entity moving from cluster i in dataset $D(\tau_1)$ to cluster j in dataset $D(\tau_2)$.

5.1. World Development Indicator Data

We evaluated our method using selected indicators from the World Development Indicator (WDI) dataset [24], which is a multi-variate temporal dataset covering 205 countries [5]. Yearly values were grouped for 10-year periods, and the latest available values are used. The experiments compare cluster structures based on the selected indicators that reflect different aspects of welfare, such as population, life expectancy, mortality rate, immunization, illiteracy rate, education, television, and inflation.

The visualizations in Figure 4 reveal several interesting changes. First, the cluster at the bottom-left of the 1980s map is missing in the 1990s map. This cluster consists of four South American countries: Brazil, Argentina, Nicaragua, and Peru. These countries were suffering economic difficulties (e.g. high inflation) due to a debt crisis in the 1980s, which is known as the 'lost decade' [26]. However, South American countries performed rapid reforms in the late 1980s and early 1990s [26]. The welfare of these countries therefore became more similar to other countries, which explains the missing cluster in the 1990s.

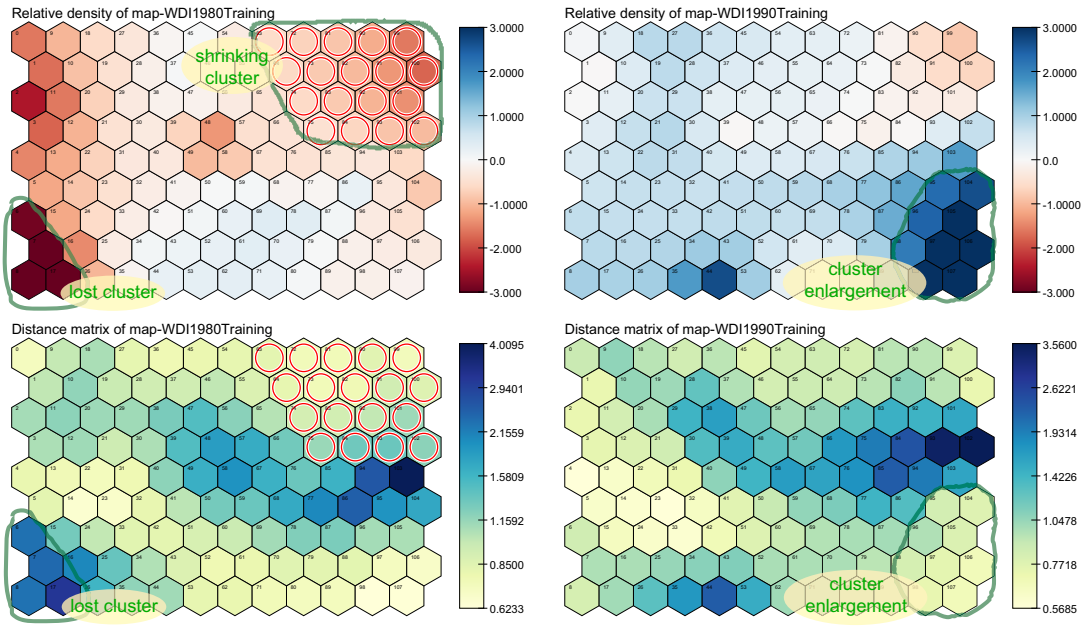

Figure 4. The world's welfare and poverty maps of the 1980s (left) and the 1990s (right): relative density visualizations (top) and distance matrix-based visualizations (bottom).

Another interesting finding is that there is a cluster enlargement towards the tail of the distribution at the bottom-right corner of the 1990s map. This new region consists of OECD (Organization for Economic Co-operation and Development) and other developed countries who achieved a higher standard of living in the 1990s, that have not been achieved in the 1980s. However, this region cannot be considered as a new cluster, as it does not have a good separation from its left neighbours, as shown in the distance matrix visualization of the 1990s map (the bottom-right map in Figure 4).

Finally, the top-right region of the 1980s map experienced a decrease in density compared to the 1990s map. This region consists of several African countries. Generated by applying hot-spot analysis [6], Figure 5 can be used to understand distinctive features of this shrinking region. After selecting this region, the sorted component planes show that this region is characterized by high illiteracy, high mortality rate, high percentage of children in the labor force, low ratio of physicians, and low school enrolment.

5.2. Australian Taxation Data

Our method has been used to explore changes in cluster structures in very large anonymized taxpayer datasets from 2003 to 2007 for the Australian Taxation Office (ATO). Here, we provide aggregate indicative results that demonstrate the effectiveness of our method, without breaching the confidentiality of the data or the specifics of the discoveries made.

The datasets consist of nearly 2.8 million entities, each with 83 numeric attributes, such as income from various sources, work-related expenses, and tax deductions, from 2003 to 2007. The datasets were pre-processed [5] and imported into the embedded database. The map size chosen was 15x20, and a map for each year was trained in around 6 hours on a Debian GNU/Linux machine running on a 64-bit, 2.6 GHz AMD Opteron, dual-core quad processor server, with 32 GB main memory. During the map training, the Java SOM Toolbox only used approximately 4 GB of memory (our JSOM Toolbox avoids loading the whole datasets into memory).

The top map in Figure 6 shows the emergence of a new large region (the dark blue region at the top of the map) in the 2007 dataset, compared to the 2006 dataset. This change is identified as a cluster enlargement as it does not have a good separation with its neighbours, as shown in the distance matrix visualization (the bottom map in Figure 6). This kind of massive change in cluster structure was not found to exist in previous time periods (from 2003 to 2006). Noting that a SOM performs vector quantization, the size of this region reflects the magnitude of the population affected by the change. Thus, this is a sizable change in the behavior of the population.

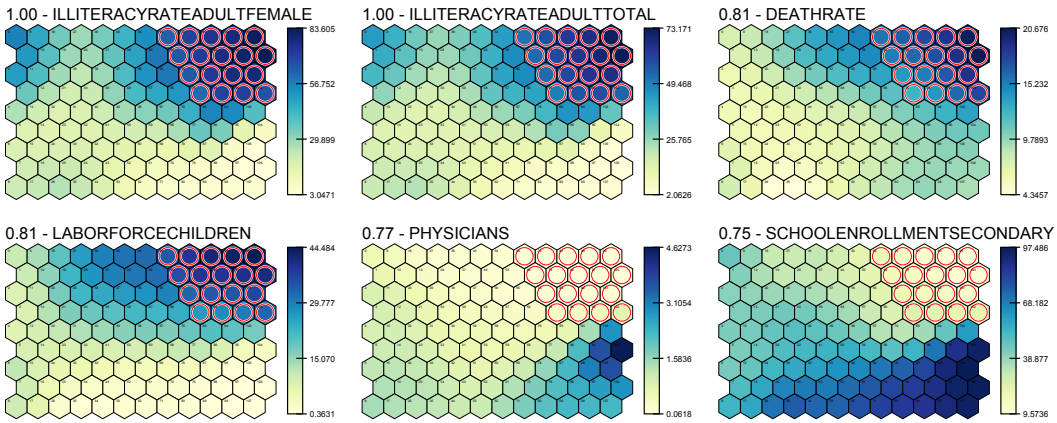

Figure 5. Top distinctive attributes for the shrinking top-right region of the 1980s map in Figure 4.

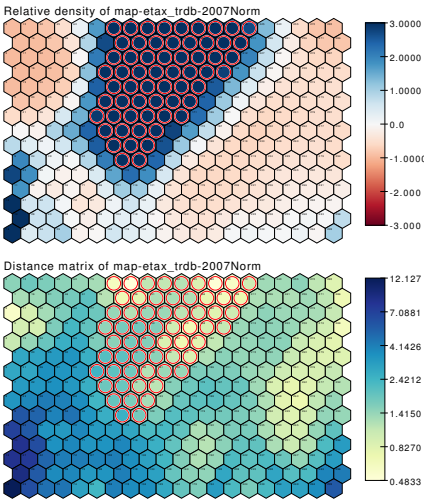

Figure 6. Relative density visualization rd_2 of the 2007 to the 2006 ATO dataset (top) and distance matrix visualization (bottom) of the 2007 ATO dataset.

Further analysis of the discriminating characteristics of this new region lead to insights that are important to the taxation analysts. The top distinctive feature was found to relate to low income rebate amounts, as shown in Figure 7. It was noted, in comparing the values to the 2006 map, that the maximum value of the low income rebate amount had doubled. Without any other knowledge, a change in behavior was identified through the deployment of ReDSOM.

An investigation, conducted after discovering this behavioral change, found that it was caused by a change in government policy. In 2006, the Australian Government increased the Low Income Tax Offset from $235 to $600 per year for the financial year 2006/2007 [20]. This also explains why low values of taxable income was also a distinctive feature, as seen in Figure 7.

6. Conclusions and Future Work

We have introduced a relative density SOM (ReDSOM) visualization that is able to show various changes in cluster structures, such as emerging clusters, disappearing clusters, cluster enlargements, cluster contractions, movement of cluster centroids, and changes in cluster density. ReDSOM has been tested with real-life datasets, including large datasets from the Australian Taxation Office.

Experiments using real-life datasets have shown that ReDSOM is capable of indicating actual changes, such as the change in economic fortunes of South American countries between the 1980s and 1990s, or the change in tax policy for low income earners.

These structural changes can be analyzed further by looking into the migration patterns of the entities. Future work incorporating migration analysis is underway.

Acknowledgement

This research has been supported by the Australian Taxation Office. The authors express their gratitude to Elea Gudgeon for providing data and domain expertise, to AusAID for providing a PhD scholarship to the first author, and to the reviewers for providing useful feedbacks. Map colors are based on `www.ColorBrewer.org`.

References

[1] G. Adomavicius and J. Bockstedt. C-trend: Temporal cluster graphs for identifying and visualizing trends in multi-

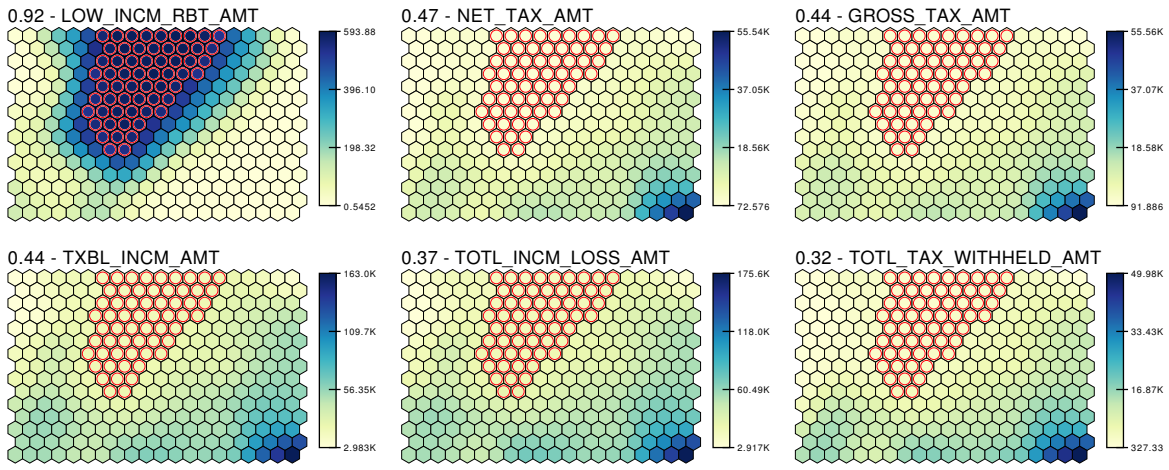

Figure 7. Top distinctive features for the selected region in Figure 6.

attribute transactional data. *IEEE TKDE*, 20(6):721–735, 2008.

[2] C. Aggarwal, J. Han, J. Wang, and P. Yu. A framework for clustering evolving data streams. *VLDB*, 29:81–92, 2003.

[3] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *ACM SIGKDD 2006*, pages 554–560, New York, NY, USA, 2006.

[4] G. Deboeck and T. Kohonen. *Visual Explorations in Finance with Self-Organizing Maps*. Springer-Verlag, London, 1998.

[5] Denny and D. M. Squire. Visualization of cluster changes by comparing Self-Organizing Maps. In *PAKDD 2005*, volume 3518 of *LNCS*, pages 410–419. Springer, 2005.

[6] Denny, G. J. Williams, and P. Christen. Exploratory hot spot profile analysis using interactive visual drill-down self-organizing maps. In *PAKDD 2008*, volume 5012 of *LNCS*, pages 536–543. Springer, 2008.

[7] V. Ganti, J. Gehrke, R. Ramakrishnan, and W.-Y. Loh. A framework for measuring differences in data characteristics. *Journal of Computer and System Sciences*, 64:542–578, May 2002.

[8] J. Han and M. Kamber. *Data Mining: Concepts and Techniques (second edition)*. Morgan Kaufmann, San Francisco, CA, 2006.

[9] S. Hido, T. Idé, H. Kashima, H. Kubo, and H. Matsuzawa. Unsupervised change analysis using supervised learning. In *PAKDD 2008*, volume 5012 of *LNCS*, pages 148–159. Springer, 2008.

[10] J. Iivarinen, T. Kohonen, J. Kangas, and S. Kaski. Visualizing the clusters on the Self-Organizing Map. In *Conference on AI Research in Finland*, volume 12, pages 122–126. Finnish AI Society, 1994.

[11] S. Kaski and K. Lagus. Comparing Self-Organizing Maps. In *ICANN'96, Bochum, Germany*, volume 1112 of *LNCS*, pages 809–814. Springer, Berlin, 1996.

[12] D. A. Keim. Information visualization and visual data mining. *IEEE Trans. Vis. Comput. Graph.*, 8(1):1–8, 2002.

[13] E. Keogh, J. Lin, and W. Truppel. Clustering of time series subsequences is meaningless: Implications for previous and future research. In *IEEE ICDM 2003*, page 115, Washington, DC, USA, 2003.

[14] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[15] T. Kohonen. *Self-Organizing Maps (Third Edition)*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 2001.

[16] P. Lingras, M. Hogo, M. Snorek, and C. West. Temporal analysis of clusters of supermarket customers: conventional vs. interval set approach. *Inf. Sci.*, 172(1-2):215–240, 2005.

[17] H. Ritter, T. Martinetz, and K. Schulten. *Neural Computation and Self-Organizing Maps; An Introduction*. Addison-Wesley Longman Publishing, Boston, USA, 1992.

[18] J. F. Roddick and M. Spiliopoulou. A survey of temporal knowledge discovery paradigms and methods. *IEEE TKDE*, 14(4):750–767, 2002.

[19] J. C. Schlimmer and R. H. Granger. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354, 1986.

[20] The Treasury - Australian Government. Press release no. 066, July 2006. `http://www.treasurer.gov.au/`.

[21] V. Tryba, S. Metzen, and K. Goser. Designing basic integrated circuits by Self-Organizing Feature Maps. In *Neuro-Nîmes'89. Intl. Workshop on Neural Networks and their Applications*, pages 225–235, Nanterre, France, November 1989. ARC; SEE, EC2.

[22] J. Vesanto and E. Alhoniemi. Clustering of the Self-Organizing Map. *IEEE TNN*, 11(3):586–600, May 2000.

[23] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[24] World Bank. *World Development Indicators 2003*. The World Bank, Washington DC, 2003.

[25] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne. Online unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004.

[26] R. Zagha and G. T. Nankani, editors. *Economic Growth in the 1990s: Learning from a Decade of Reform*. World Bank Publications, Washington, DC, 2005.