Anomaly Handling in Visual Analytics

by

Quyen Do Nguyen

A thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

_____

January 2008

APPROVED:

_____

Professor Matthew O. Ward, Thesis Advisor

_____

Professor David Brown, Thesis Reader

_____

Professor Michael Gennert, Head of Department

Abstract

Visual analytics is an emerging field which uses visual techniques to interact with users in the analytical reasoning process. Users can choose the most appropriate representation that conveys the important content of their data by acting upon different visual displays. A visual analytics application helps users to formulate and view the patterns in the datasets by means of data visualization and methods to adjust the parameters of the algorithms or the technologies provided behind the visual displays. In the analytical discourse, users can combine their information or expertise in the domain to guide the exploration to save time and to produce more satisfactory results.

The data itself has many features of interest, including clusters, trends (commonalities) and anomalies. Most visualization techniques currently focus on the discovery of trends and other relations, where uncommon phenomena are treated as outliers and are either removed from the datasets or de-emphasized on the visual displays. Much less work has been done on the visual analysis of outliers, or anomalies. In this thesis, we introduce a method to identify the different levels of "outlierness" by using interactive selection. We implemented a density-based outlier detection algorithm, where users have control over the density input parameter and the dimensions used in calculations through a graphical interface. For the visual analytics process, we developed two techniques for interacting with data regions of different outlier degrees. To compare the effectiveness of these approaches, we performed user studies on the usability of the two visual methods. The tools were developed based on XmdvTool version 7.0 and Xmdv-lite.

ACKNOWLEDGEMENTS

I would like to give sincere thanks to my advisor, Professor Matthew Ward for his instructions and valuable advice, not only about the research and course work, but also about how to get to success after a failure. He has always given strong support throughout my academic life at WPI. I also deeply appreciate the valuable suggestions from Professor David Brown, my thesis reader, which have improved the quality of the writing as well as the application of this thesis.

I would like to thank Professor Elke Rundensteiner and my team members, Zaixian Xie, Di Yang, and other students who have worked in the XMDV group, for their cooperation and the work that they have done which has inspired me in my thesis.

Last but not least, I would like to thank my family for their love and endless support, and my friend, Abraao Lourenco, for all the things he has done, which have enlightened me in my work and my life.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. MOTIVATION

Information visualization has been a field of study for decades, while visual analytics is fairly new. While information visualization focuses on the display of data to provide an overview of the datasets and the details on demand, together with the techniques such as zooming and filtering, visual analytics focuses on the interaction with users, and getting users to be involved in the analytical process to help guide the exploration. Visual analytics is the formation of abstract visual metaphors in combination with a human information discourse (interaction) to directly perceive patterns and derive knowledge and insight from them [1]. Visual analytics is the combination of many technical fields, including data visualization, statistical analysis, human-computer interaction, cognitive science, decision science, and many more. Visual analytics has been introduced to meet the critical needs of national

security by the Department of Homeland Security in 2004, and now has been used widely in bioinformatics [2], network monitoring [3], business intelligence [4], web performance [5] and many more applications.

Interactive visual interfaces are the means that facilitate the analytical reasoning in visual analytics. Visual representations and interactive displays are intuitive ways to constantly convey the abstract information to human eyes. However, no single form of visual display or data processing is superior in all applications. Providing analysts with alternatives to tailor the algorithm parameters and to process information on different types of graphs help analytical understanding evolve as the abstract data representations reveal more features about the nature of the data.

Anomaly detection is an important area in visual analytics. There is an increasing number of application areas where detecting anomalies is extremely useful: detecting fraudulent credit cards or online transactions in e-commerce and banking [6], identifying the spending behavior of customers with extremely low or extremely high incomes in marketing [7], or in medical analysis for finding unusual responses to a medical treatment [8]. Another area where detecting anomalies plays a critical role and has been investigated by a large number of researchers is network intrusion detection [9] [10]. In these cases, the rare events can be even more interesting than the regularly occurring ones. In other applications, it is very common for an occasional error to occur during the data entering process, either a mistake caused by the data entry person or an error with a sensor having difficulty reading the data. Finding and fixing these errors are important to the analysis process.

There is a need for interactive systems that allow analysts to interactively explore different regions of data points while detecting the regions with outliers and providing flexibility in viewing and adjusting the parameters of the outlier detection algorithm in the analytical discourse.

1.2. OUR GOALS

Our goals in this thesis are:

- To implement an algorithm that can be used in general-domain datasets to effectively detect anomalies.
- To investigate different visualization techniques that help users navigate through data regions and display the anomaly degrees of the data objects at the same time.
- To allow analysts to get involved in the anomaly detection process by changing the parameters of the outlier detection algorithm, thus refining the results on the visual displays and hence generating a better understanding of the data itself with the focus on its anomalous characteristics.
- To provide fast updates on the visual displays whenever there is a change in algorithm parameters.
- To study the effects of the different interaction and visualization methods on the human analytic and cognitive process.

1.3. OUR APPROACH

In this thesis, we implemented a density-based algorithm [11] which detects anomalies and identifies the level of "outlierness" for each data point in the dataset. The strong point of this algorithm is that it can detect local outliers, as it calculates the outlier degree of one data point based on the density of the neighborhood of the point itself as well as the density of the neighborhoods around all of its neighbors. In order to reduce the complexity of the calculations for big datasets in terms of the number of dimensions, we provide users with the option to specify the dimensions that they believe are more important in detecting anomalies than the others, together with the weights of each dimension to specify the degree to which that dimension contributes to the anomaly attribute of the data records. Another technique that we implemented to reduce the number of recalculations for each change in the parameters of the anomaly detection algorithm was to perform preprocessing for a certain number of frequently used parameters, thus producing better performance for the visual updates of the graphs in which data is displayed and classified based on its anomaly attribute.

For the visualization process, there are two methods for users to interact with different areas of the datasets that are mapped to different ranges of outlier degrees. In the first method, we integrated the outlier degree as one dimension in the graph. This method gives analysts a view of the dataset together with its anomaly attribute as a whole and users can navigate through the dataset in all dimensions, including the anomaly degrees. In the second method, we separated the outlier degree attribute in an anomaly-based brush

toolbox, which creates a mapping between the outlier degree range chosen in the toolbox and the data regions in the dataset. This method makes the cognition of the anomaly degree of the data objects clearer for users while providing a mapping between the data space and the anomaly attribute space. To compare the advantages and disadvantages of the interaction in each method and to evaluate the correctness of the results computed for anomaly degrees, we performed in-person user study sessions with twelve people of different levels of expertise with the visualization techniques. The user study sessions aimed at getting users involved in the exploration, tracking their activities during the interaction process with the tools in terms of time required and correctness of results, taking note of the feedback from users about the usability of the tools and their suggestions for improvement. Results and lessons learned from this user study will be described in the last chapter of the thesis.

The main contributions of this thesis are:

- We have introduced different methods to visualize anomalies in datasets and also to provide interaction techniques to assist users in the analytical discourse.

- We have implemented a density-based algorithm and optimized it with preprocessing to allow users to change the density parameter and get faster response time.

- We have developed a brushing tool for users to select the ranges of outlier degree and get updates of the dataset display on a parallel coordinates graph.

- We have integrated the anomaly detection and visualization tool inside the Xmdv-lite version.

- We have performed user evaluation on the effectiveness of the algorithm and the interaction techniques and visualization of anomalies. The results have shown that the algorithm was able to detect the expected anomalies and users were able to distinguish these anomalous points on the graph. Ninety percent of the users rates one or both of the methods to be easy or very easy to use.

The organization of the rest of the thesis is as follows: in Chapter 2 we review related work, and in Chapter 3 we give an introduction to anomalies and anomaly detection algorithms. Chapter 4 describes the density-based algorithm and our implementation. Chapter 5 gives a background on XmdvTool – the helpful features that inspired the anomaly visualization and interaction process, and introduces the anomaly-based brush and two methods to visualize and interact with the datasets and its anomalies. Chapter 6 presents our user study, the results and lessons learned from the user evaluations. Chapter 7 gives conclusions of this thesis and describes future work.

CHAPTER 2

RELATED WORK

2.1. ANOMALY VISUALIZATION

Most of the techniques in anomaly or outlier visualization try to identify the outlying values and then either remove or visualize them on the graphs [6] [12]. Ming Hao et al. [6] introduced a new visualization technology called VisImpact for analysis and anomaly detection in business operations. Important factors in the business process are presented as nodes on a bipartite flow graph in the form of cause and effect, with edges representing the relationships between two attributes. The anomaly detection is based on the correlation relationship of each pair of the two nodes that can be viewed on the graph. This type of application is very domain specific, thus requiring a particular design for the algorithm and visualization of the data.

Novotny and Hauser [12] integrated an outlier treatment in their work, in which datasets are divided into bins as in a histogram and, based on the frequency of occurrence of the data records, outliers are detected. The outliers are then visualized on parallel coordinates as one category, together with focuses (the portion of data to be highlighted on the display) and trends (the rest of the data) (Figure 1, 2). In this approach, users have no interaction with the process, making the method inflexible; the performance of the program depends on a predefined set of parameters such as the number of bins or the filter threshold. Several experiments need to be conducted to determine an effective set of parameters for which the discovery of outliers seems to be reasonable.
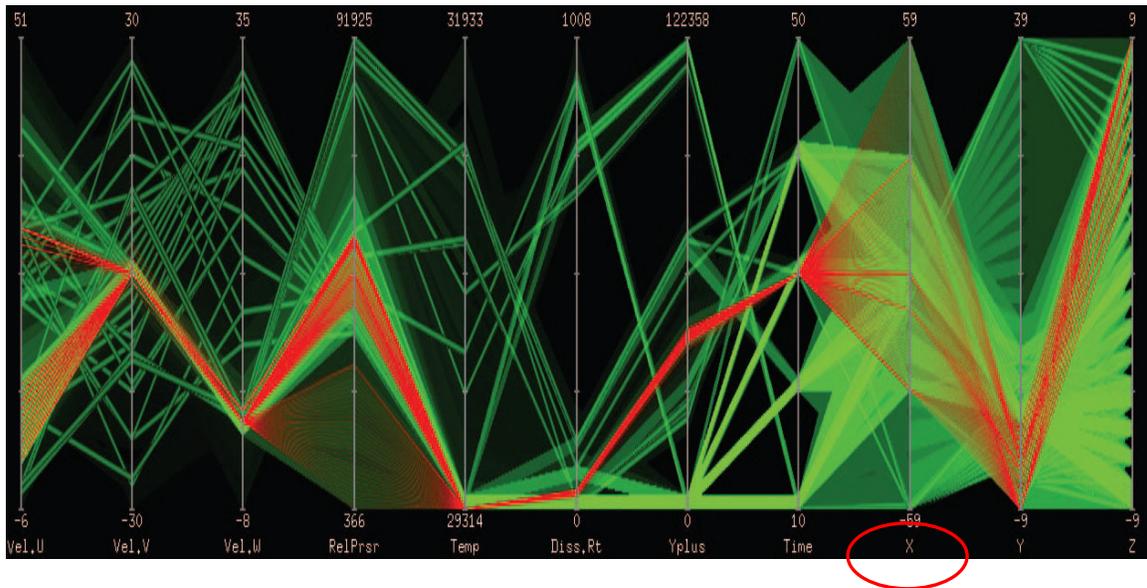
Figure 1. A flow simulation dataset with the focus displayed in red, the context displayed in green (the blended area) and some outliers on the X-axis (the sharp lines in the same green color, the X-axis is circled)

(Image from [11], used without permission)
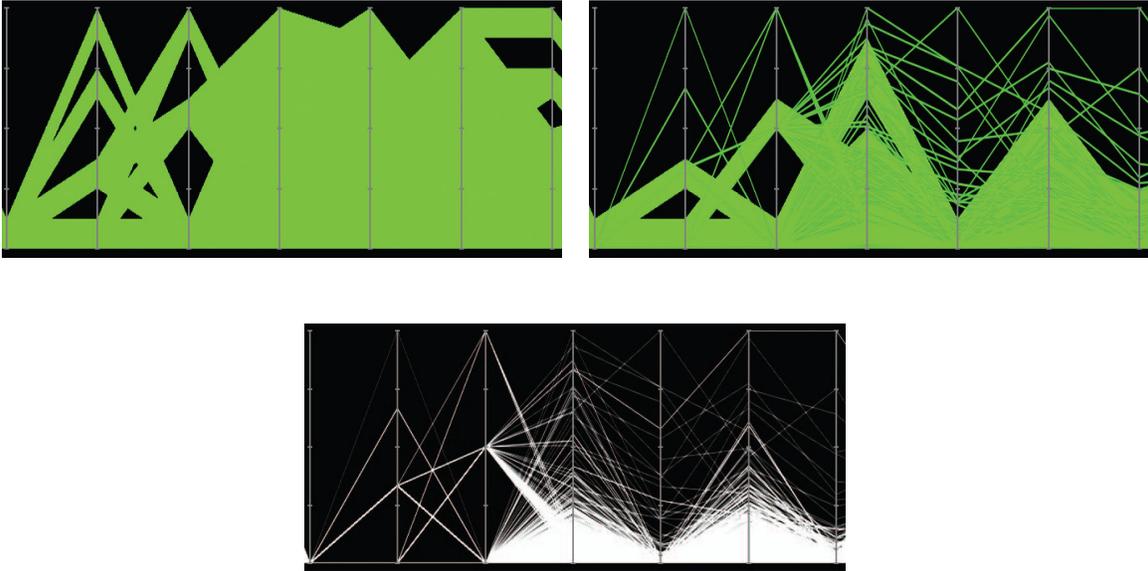
Figure 2. A dataset with outlier treatment (right) and without outlier treatment (left) – the clusters were distorted. The original dataset is on the bottom

(Image from [11], used without permission)

Many anomaly detection techniques only provide a binary categorization for the anomaly attribute in the dataset. This means that any data object in the dataset is classified as either anomalous or not. Fabio González et al. [13] introduced a model for discriminating and visualizing anomalies. The model is trained with only normal samples and will learn from encounters with new anomalies. It is combined with a negative selection algorithm and a self-organizing map (SOM) inspired from the human's immune architecture to detect anomalies and produce the visual representation to discriminate among normal, known abnormal, and unknown abnormal regions.

Davidson and Ward [14] proposed a clustering-based anomaly detection framework which was originally used for visualizing clustering results by

representing clusters as affected by gravitational forces. The cluster centers are placed as particles in a three-dimensional space, with gravitational effect of the cluster centers on a particle given by the degree to which the particle belongs to its cluster. In this method, the observations that do not belong to any clusters with a degree greater than some threshold are identified as anomalies. In this context, anomalies are the points that do not belong strongly to any clusters, or they are similar to the data points of more than one cluster (Figure 3).
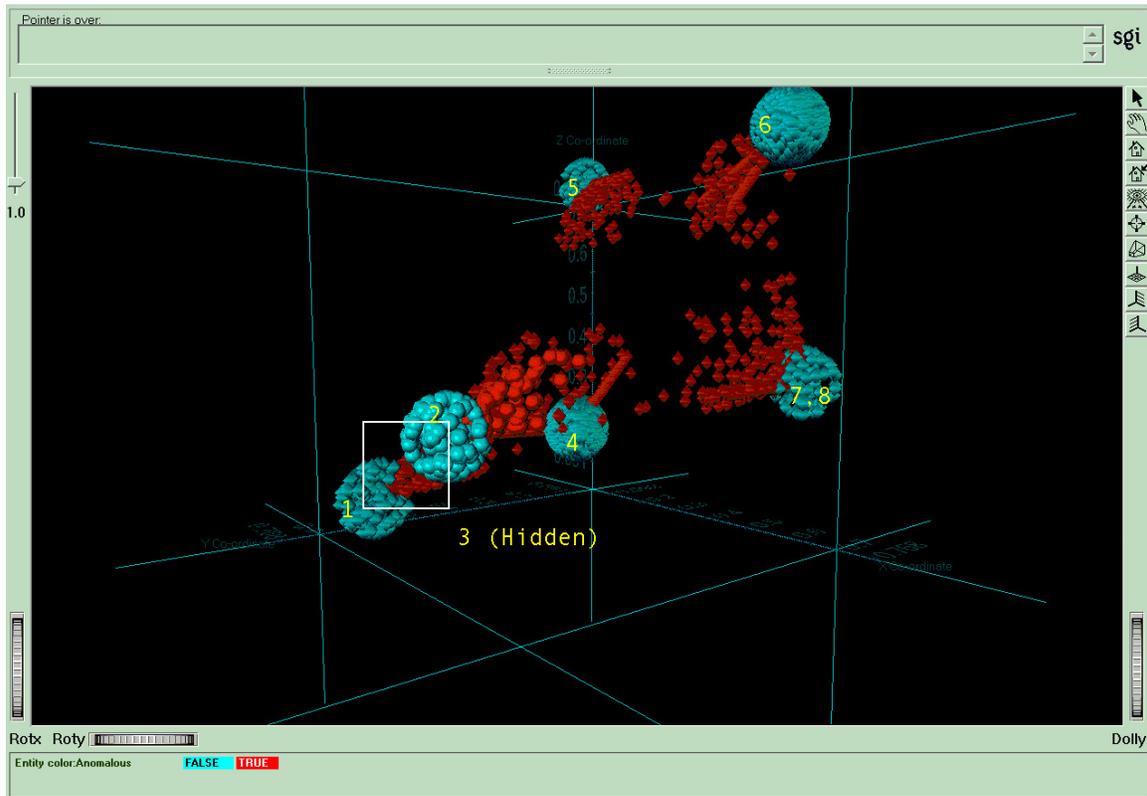
Figure 3. A visualization of the UCI Churn dataset, with clusters shown in blue and are the spheres of particles strongly tight together. Anomalies are marked in red. For example, those points that lie in between cluster one, two and three represent voice plans that have the same amount of international calls as other clusters but also have high usage of other types of calls such as daytime calls.
(Image from [14], used with permission)

2.2. INTERACTIVE BRUSHING

There have been several other systems that provide analysts with sliding tools to brush through different regions of the datasets [15] [16] [17] [18]. Pin Ren et al. [15] built an interactive system that allowed users to brush on a correlation matrix view to highlight traces of unsuccessful connections with similar patterns. David DesJardins [18] built "live" graphs which use EDA+ (Exploratory Data Analysis Plus) techniques such as brushing and animation to brush across outlier points.

Andreas Buja et al. [17] performed data visualization for high-dimensional datasets through interactive view manipulations: focusing, linking and arranging views with the use of brushing as a method to perform queries with the database visually. They implemented the techniques in XGobi - a multivariate visualization system which uses real-time controls to tune the views and give visual feedback. XGobi is not used for detecting anomalies particularly, but it introduced a concept of "linked scatterplot brushing", where the actions in one window are immediately reflected in another window displaying the same data.

Ying-Huey Fua et al. [16] also introduced a structure-based brushing technique, which allows users to navigate data by choosing the focal extents and level of detail parameters on a window that display the hierarchical structure of the data. Their brush used proximity-based coloring as a means to map data that is closely related in the structure to similar colors. This coloring scheme helps convey the relationships among data, as well as the anomalies.

CHAPTER 3

BACKGROUND

3.1. ANOMALY CONCEPT AND THE ANOMALY DETECTION PROBLEM

Anomalies are data objects that do not comply with the general behavior or model of data. Such data objects are grossly different from or inconsistent with the remaining set of data (adaptation from the definition of "outliers" from the book "Data Mining: Concepts and Techniques" [19]).
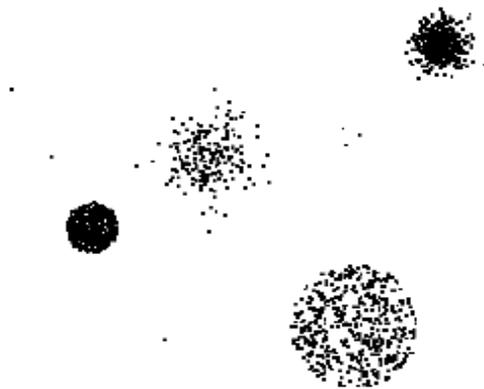
Figure 4. An example of a dataset with anomalies ([19], used without permission)

Anomalies can be caused by the collecting of data from different classes. For example, in a dataset that stores the weights of oranges, there might be a mixture of a few grapefruits. The data itself can be erroneous, for example, a human body measure dataset might specify an entry for a two year old baby who weighs 200 pounds. However, the data itself can indicate a natural variation itself; for example, there could be an exception for unusually tall people. It is the knowledge of analysts that helps interpret the meaning of these exceptional phenomena and determine if the results found indicate outliers – the noise in the datasets, or they are anomalies – data that may be of interest to users [19].

There can be several variants to an anomaly/outlier detection problem [19]. The first one is, given a database D, find all the data points x ∈ D with anomaly scores greater than some threshold t. Another one is to find the data points in a database D having the top-n largest anomaly scores f(x). The third approach deals more with relative anomaly scores and pattern matching in the dataset; it starts with a database that contains mostly normal (but unlabeled) data points, and a test point x. The requirement is to compute the anomaly score of x with respect to D.

Finding anomalies can be challenging, as in many cases we do not know how many outliers or anomalies exist in the datasets. Assigning thresholds to anomaly scores in order to flag data objects of high anomaly degrees can be difficult as the thresholds vary among datasets and it may require many testing in order to find out the correct ones. With the working assumption that there are considerably more "normal" observations than "abnormal" observations (outliers/anomalies) in the data [19], the problem of detecting

outliers in the dataset may become a "finding the needle in the haystack"
problem.

3.2. ANOMALY DETECTION ALGORITHMS

3.2.1. STATISTICAL APPROACH

The probabilistic definition of an outlier is that it is an object that has a low
probability with respect to a probability distribution model of the data [11].
This is a traditional approach in detecting outliers using a probabilistic data
model and a discordance test, a procedure to determine whether a particular
object is an outlier or not. The statistical test verifies the basic hypothesis – a
statement about an object fitting in a probabilistic model of the system or
being generated by a distribution law, against the alternative hypotheses
[19]. A test depends on the data distribution, the parameters of distribution
(e.g., mean, variance) and the number of expected outliers (confidence
limit)(see Figure 5).
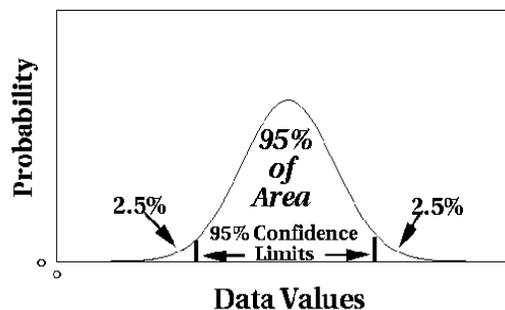
*Figure 5. The one dimensional Gaussian distribution of the dataset with the statistical
confidence interval of 95% ([19], used without permission)*

A simple example that uses a statistical approach for anomaly detection is illustrated in Figure 6. Assuming we have a system and its parameters are modeled as independent, Gaussian random variables. We define a range of normal values for each variable; each time there is a feature of an observation in the data that falls out of the defined range, we increase it score. As the variables are independent, the probability distribution of the scores was calculated. The probability density of this distribution is shown in Figure 6 below.
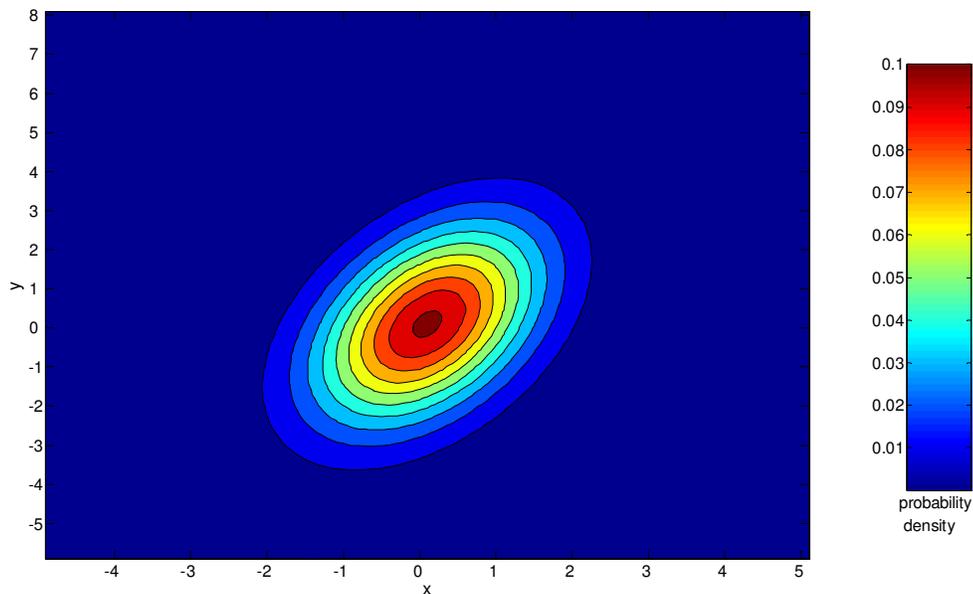
Figure 6. Two dimensional Gaussian distribution with a probability density scale score

([19], used without permission)

The statistical-based approach possesses a number of strengths inherited from its base in mathematical statistics [9]. For example, the verification of

competing hypotheses is a conventional problem of mathematical statistics and can be applied to the statistical model used in outlier detection. The statistical method can be very efficient and yield good results once the probabilistic model is known. Furthermore, as the data itself is not needed any more once the data model is generated, the method is also space-efficient as just a minimal amount of storage is needed for the data model. However, in many cases, there may not exist a statistical distribution model for the data given, or the process of constructing this model is rather complex, hence the computational procedure for finding the parameters and conducting tests for hypothesis verification can be complex, especially in the case of high-dimensional data. This process becomes extremely difficult when the percentage of outliers in the data is high, distorting the parameters of the distribution.

3.2. 2. DISTANCE-BASED APPROACH

A definition for outliers in the distance-based approach is "an object is an outlier if a specified fraction of the objects is more than a specified distance away" [20]. This is the most popular approach in detecting outliers in data, which is based on the calculation of distance between pairs of objects in the dataset. The basic method is the one that defines DB(p,d), which states that an object o is an outlier if at least p percent of all objects of the database are at a distance greater than d from o. However this method has a big disadvantage, as it loses globality and it requires the specification of the parameters p and d in advance. Various algorithms have been designed to implement this method; the purpose is to facilitate different models of data

storage and different numbers of dimensions. Among these methods are the index-based algorithm, the nested-loop algorithm and the cell-based algorithm. All of these algorithms are described in detail in [21].

Another algorithm that uses a distance-based approach but has some advantages over the DB(p,d) method is the k-nearest neighbor algorithm [22]. This method defines the k-neighborhood of an object o to be the set of k nearest objects. The outlier score of an object is the k-distance, which is the maximum distance from that object to its neighbors in the k-neighborhood. The k-nearest neighbor algorithm does not depend on the parameter d and it is also able to detect outliers among objects that lie deep in the data, not only the ones belonging to the boundary of the dataset.

Outlier score
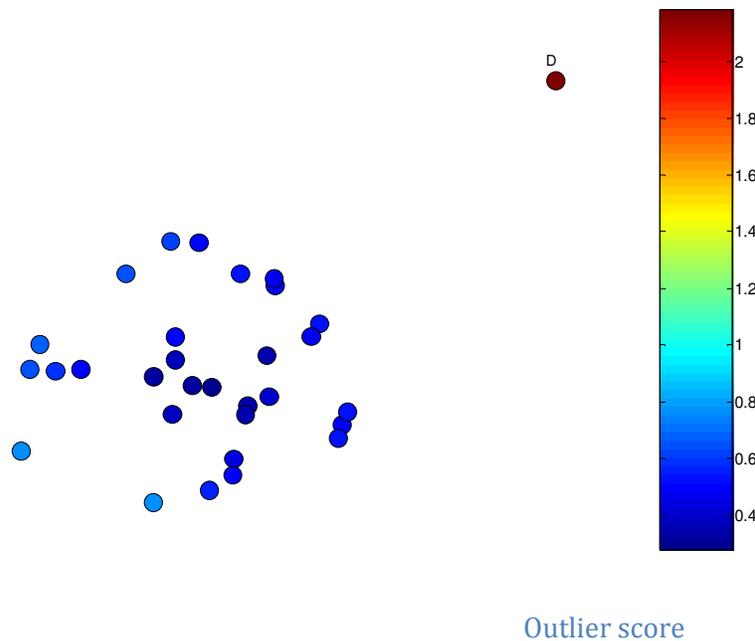
Figure 7. Dataset processed with 1-nearest neighbor algorithm, one outlier detected

([19], used without permission)
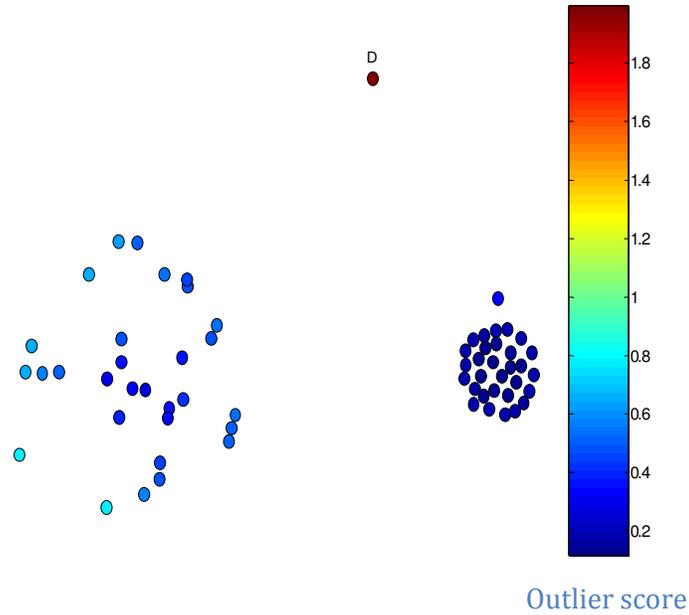
Figure 8. Dataset processed with 5-nearest neighbor algorithm and differing density

([19], used without permission)

The biggest advantage of the distance-based approach is its simplicity; all of the parameters have clear meaning, and it is able to detect local outliers. However, the complexity of this algorithm is high (quadratic). Secondly, the model is sensitive to parameters, such as the number of neighbors k and the distance d. If these parameters are changed, a new model needs to be reconstructed. The model is also sensitive to variations in density; modern information systems may contain heterogeneous data of complex structure, or data objects may have discrete or nominal values, making the distance definition difficult. Moreover, when the number of dimensions in the dataset becomes high, it is less meaningful to define the distance across all of those dimensions.

3.2.3. MODEL-BASED ANOMALY DETECTION

This approach builds a model of the data and checks for data points that do not fit well in the model, or those that distort the model, and define these as anomalies. This approach is similar to the statistical-based approach, but using a data model trained with "normal" data instead of using a statistical model [11].

3.2.4. CONVEX HULL METHOD

This is a very simple method, where extreme points are considered to be outliers. It uses a convex hull to cover all the data point region and detect extreme values. The data points that lie around the boundary are indentified as outliers. However, this method fails to detect outliers that lie in the middle of the data [11].
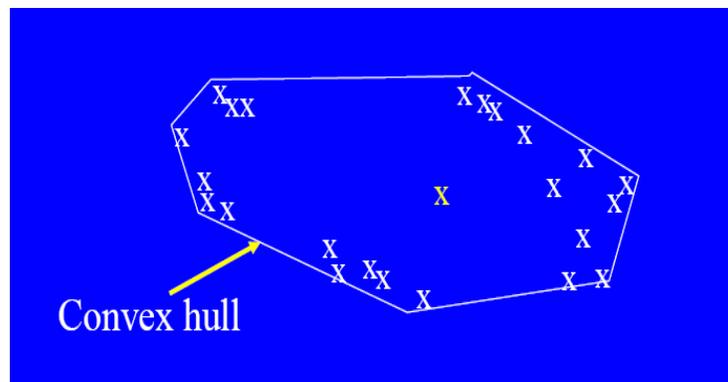
Figure 9. Convex hull method with an undetected outlier (marked as yellow)

([19], used without permission)

CHAPTER 4

DENSITY-BASED ANOMALY DETECTION AND IMPLEMENTATION

4.1. OVERVIEW

The density-based algorithm is an efficient method to detect anomalies; especially when the data is not uniformly distributed [19]. This method uses the number of "neighbors" that a point has in order to determine if the point is an outlier. Intuitively, a data point that does not have many "neighbors" will be considered isolated from other points and thus is an outlier or an anomalous point. How many neighbors a point should have to be counted as an "inlier" can vary and depends on the nature of the problem and the data characteristics. Interaction by users can help determine the level of deviation by which a point can be considered as an outlier.

Based on the number_of_neighborss input parameter, this algorithm calculates the distance of the k-neighborhood, which is similar to the process in the k-nearest neighbor algorithm described earlier in chapter 3. The density value for each data point is acquired by inverting the value of the average distance of the neighborhood. After the density values are calculated for all data points, the level of outlierness, or the outlier score is computed as the average of the ratios of the density of a data point and the density of its nearest neighbors. Outliers are the points with the largest local outlier factor (LOF) value. This factor indicates the outlier degree, thus hereafter, we will use the terms "LOF", "outlier degree", "outlier score", "anomaly degree", or "anomaly score" interchangeably.

The density-based algorithm is able to detect local outliers, as it uses not only the density of the data point itself to calculate the outlier score, but also considers the density of the neighborhood around that point. If we have a non-uniformly distributed dataset, where there is a mixture of dense and sparse regions of data, the density-based method can identify a data point that is relatively close to its neighbors, but the density of the data points in its neighborhood is much higher than the density of the data point itself. That point will have a high degree of outlierness, or a high chance of being an outlier. The idea is illustrated in Figure 10.

4.2. FORMULA REPRESENTATION OF THE DENSITY-BASED ANOMALY DETECTION ALGORITHM

In order to determine whether a point is an outlier or not we need to specify the degree to which an object is an outlier, or degree of "outlierness", denoted by the **local outlier factor (LOF)**.
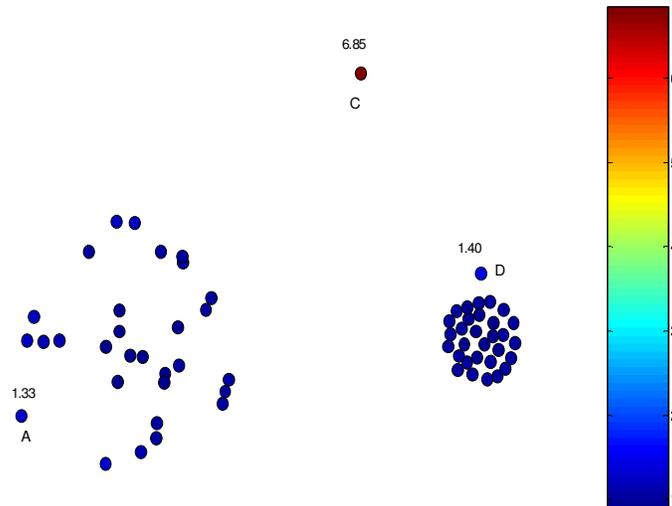
Figure 10. A distribution of a dataset which is a mixture of sparse and dense regions, where C is detected as a global outlier and D is detected as a local outlier by the density-based algorithm. The numbers beside A, C and D are the outlier scores

([19], used without permission)

In order to define the local outlier factor of a data point, we need to introduce the related concepts of k-distance, k-distance neighborhood (or k-neighborhood), reachability distance and local reachability density [19]:

- k-distance of an object p: the maximum distance between p and its k nearest neighbors. This distance is denoted as k-distance(p) such that for every object o that belongs to the k-nearest neighbors of p, the distance between p and o:

  d(p,o) ≤ k-distance(p)

30

Here the notation of k can also be used interchangeably with the MinPoints notation

- k-distance neighborhood of an object p: $N_k(p)$ contains at least MinPts nearest neighbors of p (MinPts = k)

- reachability distance of an object p with respect to object o:

reach_dist$_{MinPts}$(p,o) = max (MinPts-distance(o), d(p,o))

Consequently, the reachability distance between p and o is the actual distance if o is beyond the MinPts neighborhood of p, and it will be the MinPts-distance if o is within the region.

- local reachability density of p (lrd) is the inverse of the average reachability density based on the MinPts-nearest neighbor of p:

$$lrd_{MinPts}(p) = 1 \Big/ \left(\frac{\sum\limits_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)}{|N_{MinPts}(p)|} \right)$$

And now, the local outlier factor (outlier degree) of p is defined as:

$$LOF_{MinPts}(p) = \frac{\sum\limits_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

The local outlier factor of object p captures the degree to which we consider p to be an outlier. The lower p's local reachability density and the higher the local reachability density of p's MinPts-nearest neighbors are, the higher the

LOF(p) is. Ideally, when LOF(p) = 1, p is not an outlier. The higher LOF (p) is, the higher its degree of "oulierness".

4.3. OUR IMPLEMENTATION

The algorithm is briefly described as follows [11]:

1: {k is the number of nearest neighbors}
2: **for all** objects **x do**
3:    Determine $N(\mathbf{x}, k)$, the k-nearest neighbors of **x**.
4:    Determine $density(\mathbf{x}, k)$, the density of **x**, using its nearest neighbors, i.e., the objects in $N(\mathbf{x}, k)$.
5: **end for**
6: **for all** objects **x do**
7:    Set the $outlier\ score(\mathbf{x}, k) = average\ relative\ density(\mathbf{x}, k)$

8: **end for**

The average relative density in the algorithm above is calculated as:

$$average\ relative\ density(\mathbf{x}, k) = \frac{density(\mathbf{x}, k)}{\sum_{\mathbf{y} \in N(\mathbf{x},k)} density(\mathbf{y}, k) / |N(\mathbf{x}, k)|}.$$

As the algorithm calculates the distance between each pair of data points, the complexity of this algorithm is n square, where n is the number of data points.

To determine the k-nearest neighbors for each data point, we store the calculated distances between each pair of objects in an ordered list so that for

a given k (number of neighbors input parameter), we just have to retrieve from the list the first k data objects that are connected to one data point. These data objects are ensured to be the k-nearest neighbors of that point. The construction of this distance list is part of preprocessing, so that when the input parameter k is changed, we just need to look up this distance list and extract the first k objects for one data point and recalculate the outlier score for that data point.

The weights of the dimensions chosen to be considered for outlier detection are assigned in a dialog box and are passed to the algorithm (see Figure 11). If a dimension is chosen, it needs to be assigned a weight from zero to one, but the sum of all selected dimensions should add up to one. The unselected dimensions have weight zero. If no dimension specification is made, all the dimensions are treated equally and taken into the anomaly detection calculation.
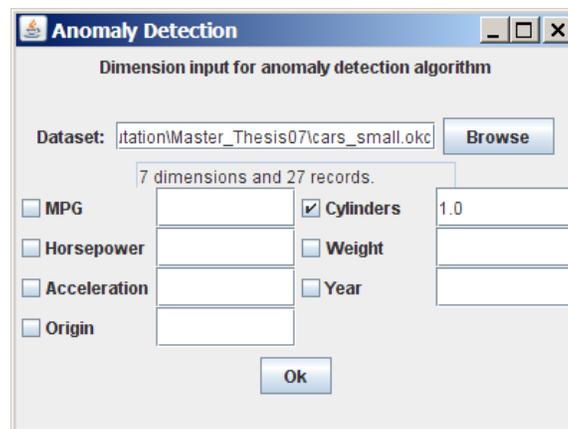

Figure 11. A dialog box where users can choose the dimensions for anomaly calculation and assign weights to them. In this case, a user chose to detect anomalies on the cylinder dimension, and the weight assigned is one.

Figure 12 shows a dataset about cars, where the algorithm detects outliers on one dimension, the "cylinders". In this dataset, there are cars that have three, four, five, six or eight cylinders. The graph shows that the cars that have high outlier degree (the LOF is close to 1) are the ones that have three and five cylinders.

When the number of dimensions to be calculated in the algorithm is greater than one, the relationships among dimensions are also the factors that affect the outlier degrees of data objects. In Figure 13, the dataset is extracted from the cars dataset of the StatLib dataset archive [23]. Anomalies were detected on two dimensions: mpg (miles per gallon) and cylinders. As shown on the graph, cars that have high outlier degree (LOF values lie in the upper half of the column) are the ones that have six cylinders and the ones that have eight cylinders but the miles per gallon is very low. This is because although the minimum number of neighbors required here is pretty small, there are not so many cars that have six cylinders in the dataset (4) in comparison with other cars that have four cylinders (7) and eight cylinders (14). For the cars that have eight cylinders, most of them have a moderate number of miles per gallon, thus the ones that have a very low number of miles per gallon are considered to be outliers.

For a dataset, the algorithm is run for the number_of_neighborsss parameters of one to twenty percent, with a difference of one percent, and thirty to one hundred, with a difference of ten percent, of the dataset size. The purpose of this pre-calculation is to provide faster response time whenever users change the input parameter. The parameter range is non-linear, as it is biased towards the smaller range (one to twenty). This is based on the reasoning

that if this parameter is greater than twenty percent, a big portion of the dataset may be identified as outliers, which does not conform to the definition of outliers which states that these should be the "rare events" in the dataset. This preprocessing step in the algorithm is to prepare for the visual display of the dataset, which will be described later in the thesis.

Figure 12 . An example of a dataset about cars; anomalies were detected in the cylinders dimension.
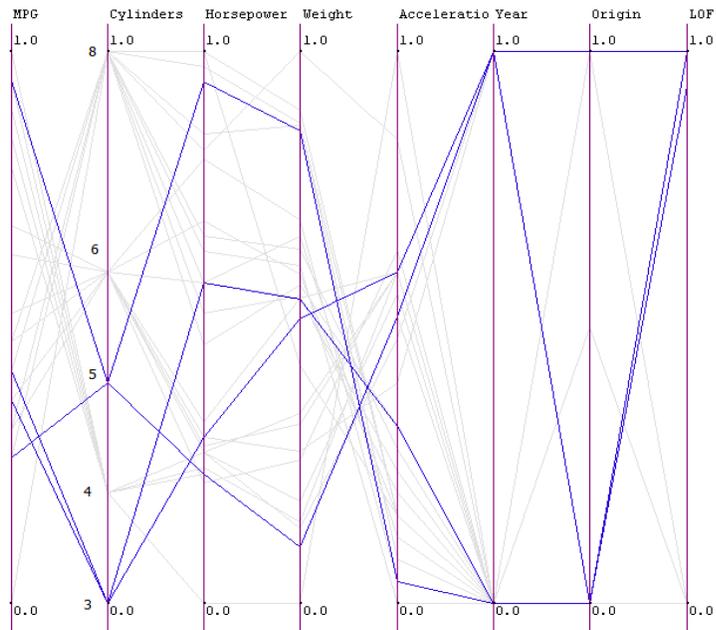
Figure 13 . A dataset extracted from the cars dataset with anomalies in two dimensions: mpg and cylinders.
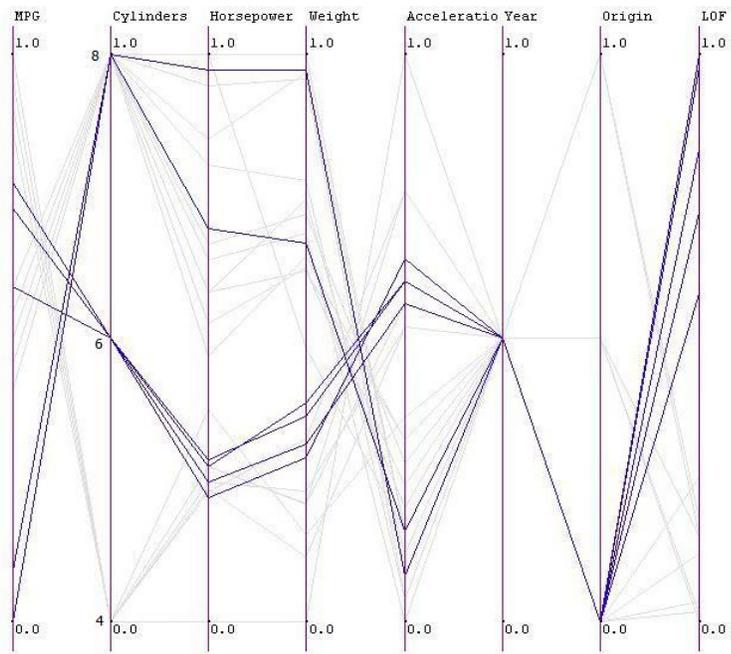
CHAPTER 5

ANOMALY VISUALIZATION

5.1. XMDVTOOL

XmdvTool is a visual exploration environment where the viewing process of data is supported with five classes of techniques to display flat (non-hierarchical) and hierarchical data, namely parallel coordinates, scatterplots matrices, glyphs, dimensional stacking and pixel-oriented displays [24]. Among these, the parallel coordinates graph is a very powerful display technique; it is a geometric projection technique used for multidimensional visualization and automatic classification. Each of the dimensions of the dataset is displayed in one vertical axis, and the data record is represented by a multi-line, which traverses across all of the vertical axes and connects the value projected in each dimension (Figure 14).
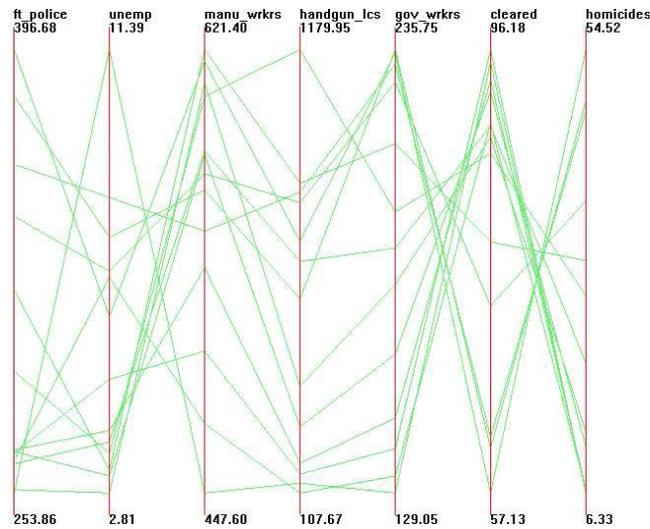
Figure 144. Parallel Coordinates visualization of Detroit crime dataset

(7 dimensions, 13 data items)

The following features have been provided for different types of graphs in general, and for parallel coordinates in particular, within XmdvTool:

➢ Brushing
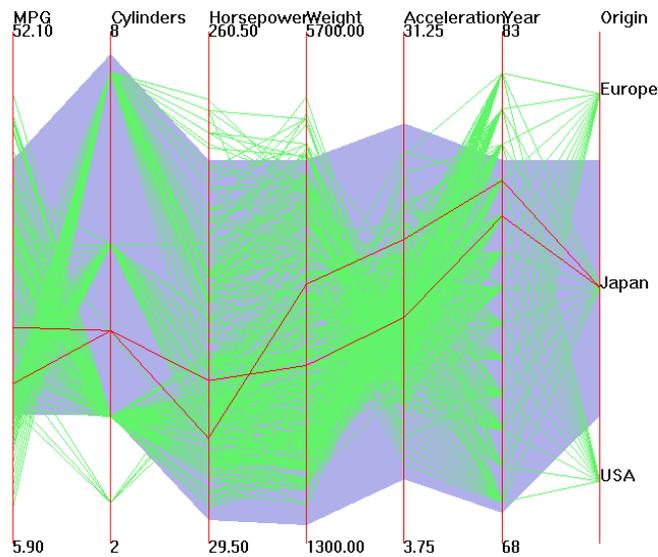
Figure 15. Brushing in a Parallel Coordinates graph

A brush marks the data records that fall entirely in the highlighted (blue) region. The selected points are drawn in red (Figure 15).

➢ Structure-based brush

The structure-based brush allows interactive navigation within a data hierarchy and produces real-time mapping from the selected region on the brush to the data on the graph. A structure-based brush also allows dynamic masking, which creates a fade-in, fade-out effect for the brushed/unbrushed regions (Figure 16).
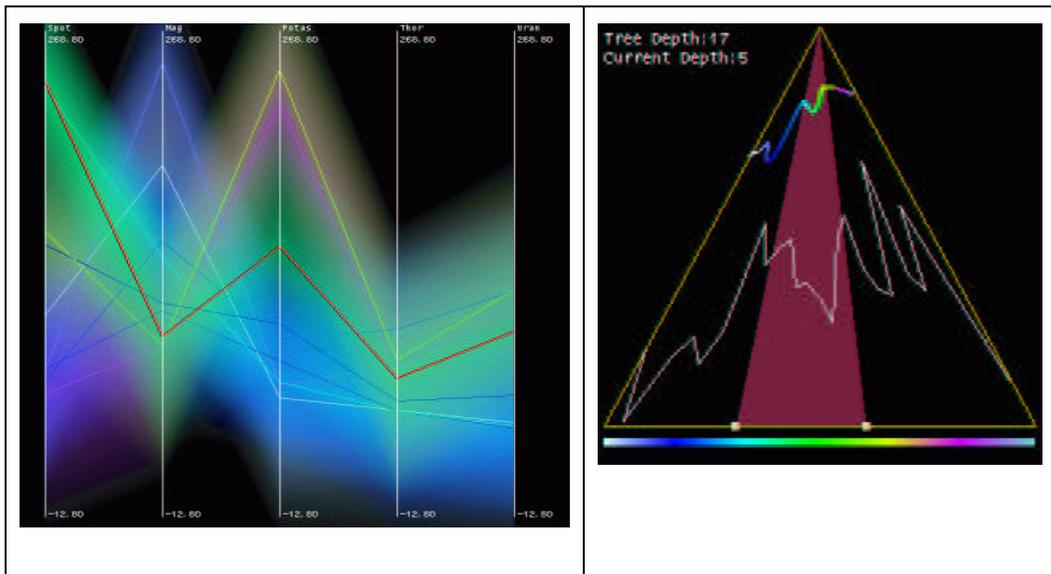
Figure 156. A structure-based brush for a hierarchical view of a dataset. The toolbox on the right has a brush handle for users to move, enlarge or shrink the focus; the highlighted region on the graph to the left is also changed correspondingly (shown in orange).

The idea of interactive brushing and highlighting based on some feature of the dataset can be used in anomaly detection and visualization. Users can interactively select and change the parameter of the anomaly detection

algorithm and navigate through different regions that are mapped to different anomaly scores. XmdvTool also provides many other tools, such as dynamic masking, zooming, and changing the brush radius [25] to make a clearer or more detailed view of the selected data regions.

5.2. DIMENSION AUGMENTATION IN PARALLEL COORDINATES

In this method, we perform anomaly brushing with the built-in brush in XmdvTool. A brush, in this context, is a method that allows users to select the regions of data that they consider more interesting or more important than the others. After applying the density-based algorithm with a specified number_of_neighbors parameter, the outlier degree value of each data point is appended at the end of the data record as a new attribute. On the parallel co-ordinates graph, this attribute is displayed as an additional dimension of the dataset, allowing it to be treated as other dimensions in the exploration process of the data.

5.2.1. BACKGROUND

The dimension augmenting method that we present was originally inspired by a technique introduced by Z. Xie et al. [26] where an interactive brush is formed between data space and quality space. This linkage was created by calculating the quality for all data points, leading to the aggregated quality for data columns and records in the dataset. This calculation constructs a quality

matrix that maps each data value to a quality value in the new matrix. The number of rows and columns in the quality space are augmented by one, which means N+ 1 dimensions and M+ 1 records in the new matrix, where M and N are the number of records and attributes in the original matrix (dataset); the additional dimension and record are for the record quality and dimension quality respectively. The quality information is integrated in the new dataset as an additional column (which are mapped to record quality) and an additional record (which is mapped to the column quality). The results were then visualized on different graphs provided by XmdvTool. Figure 17 shows the data that was brushed together with the information about its record and column quality. Xie also built an interactive brushing toolbox for the new quality space, with a rectangular slider for each dimension, and the data points that fall into a selected quality range are highlighted on the graph (Figure 18).
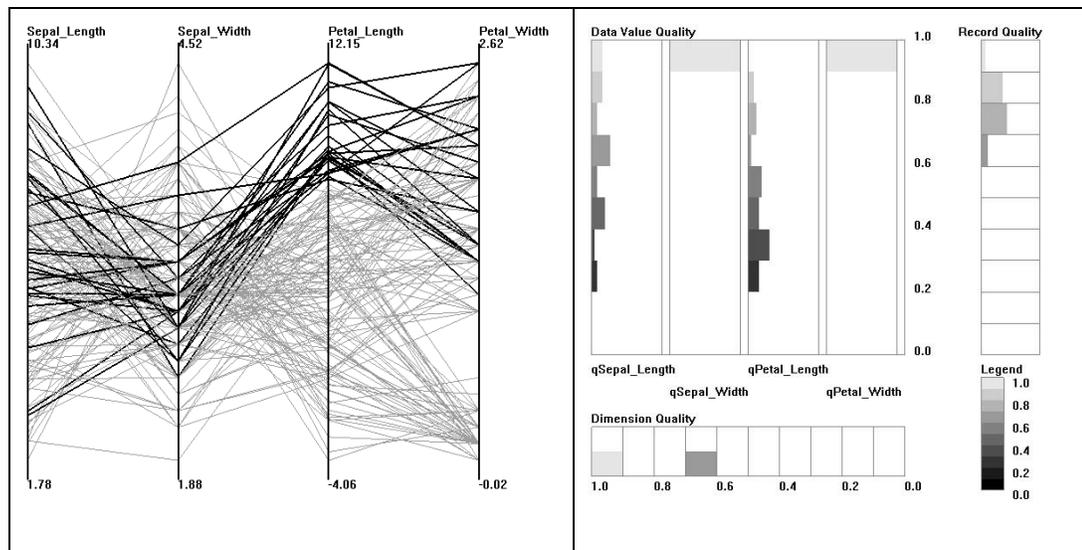
Figure 167. A visualization of the iris dataset, where the high values of Petal_Length value were chosen on the left, and the linked quality space was shown on the right.
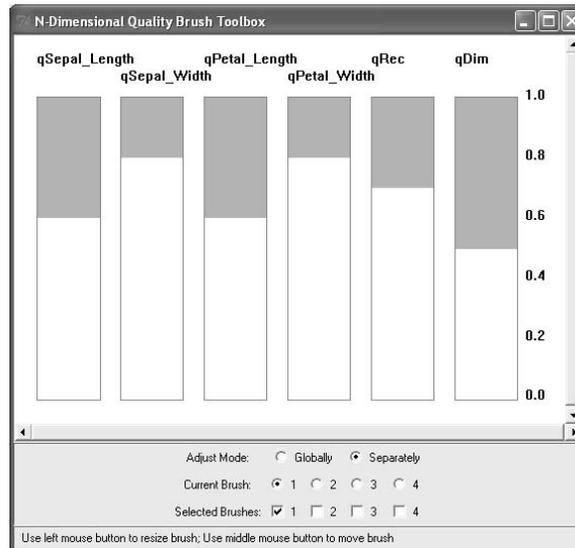
Figure 18. Quality brushing definition toolbox from Xie's program; shaded areas correspond to selected quality ranges for each dimension and record.

5.2.2. ANOMALY VISUALIZATION WITH DIMENSION AUGMENTATION

In our method for visualizing anomalies, we display the outlier degree attribute as the last dimension in the parallel coordinates graph. Users can see the mapping of each data record (a data point) to an outlier degree value, which specifies how anomalous the data point is. Users can interact with the graph by selecting a subset of the dataset in any of the dimensions. There are two types of interactions supported in this graph:

- Selecting a region with high/low outlier degree: users can paint over a subrange of outlier degrees (LOF) and the matched data points will be marked on the graph.

- Selecting a region which is the combination of conditions on multiple attributes: users may be interested in only a subset of the dataset, thus they can use the N-dimensional brush to choose the ranges of values on the dimensions they want to set the value limits. Then the anomaly scores (outlier degrees) of those chosen points will be shown on the last dimension.
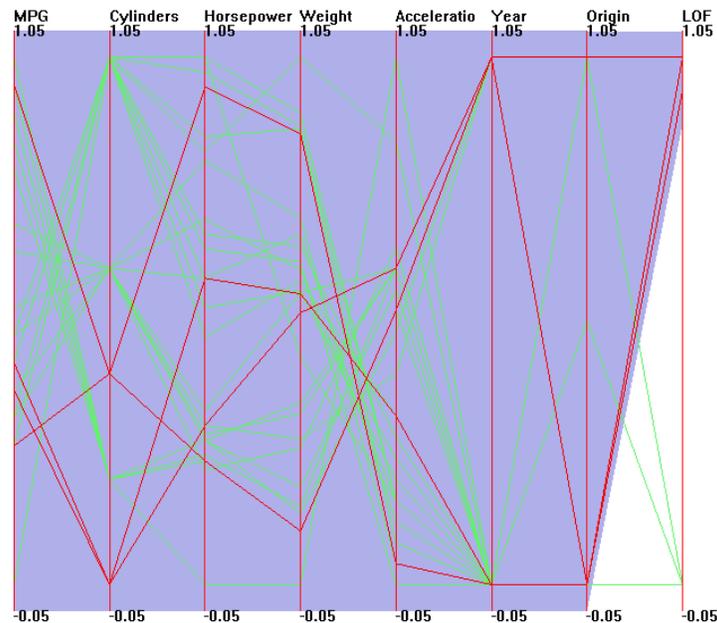
Figure 179. A Parallel Coordinate graph display of the cars dataset, where the last dimension (LOF) denotes the outlier degree. Here the high LOF region is selected, and anomalies are detected on the "Cylinder" dimension. As we can see, cars that have the lowest and third lowest values in the cylinder dimension (which are three and five cylinder cars) are detected as outliers.

5.2. ANOMALY-BASED BRUSH

The conventional visualization which displays data in static graphs provides limited capabilities for analysts to review and understand the data. Nowadays, when the size of data can grow rapidly, the characteristics of the data, such as patterns, clusters or anomalies, can change according to the new data coming in. There needs to be new methods that communicate with users interactively, regarding either the changes in the data themselves, or in the results produced by the system that need to be evaluated and adjusted by users. The interactive graphic forms create a "live" display of the data, not only to give users an insight into the data and the relationships existing among its dimensions and records, but also to help users review their conclusions about the data and assure the results yielded from the graphs are satisfactory.

The anomaly-based brush is a navigation tool where users can choose the number_of_neighbors input parameter for the density-based anomaly detection algorithm and explore data regions mapped to different outlier degrees according to this input parameter. The brush consists of a control box and the graph itself; here we chose parallel coordinates as, for a modest number of dimensions, users can see clearly the relationship among all of the dimensions. We assume that the number of dimensions needed to be displayed on the graph is small, because in our preprocessing phase, we

already allow users to choose the most important attributes to use in anomaly detection. We do not provide any limit on the number of dimensions that can be specified, but it is an inherent characteristic of the parallel coordinates graph that, as the number of dimensions in the graph grows, it becomes more difficult to track the relationships among data, hence resulting in a less effective display. For high dimensional datasets or datasets with millions of records, we can still apply this brushing technique on other types of graphs, such as scatterplot matrices and/or in combination with other visualization techniques, such as hierarchical data displays [27].

5.2.2. THE IMPLEMENTATION OF THE ANOMALY-BASED BRUSH

Our purpose for creating the anomaly-based brush is to show the mapping between the anomaly score space and the data space. Instead of adding the outlier degree as an additional dimension to the dataset, we built a separate anomaly-based brush toolbox. With this toolbox, analysts can adjust the number_of_neighbors parameter and choose the range of outlier degree. Shown in Figure 20 is a snapshot of the toolbox.
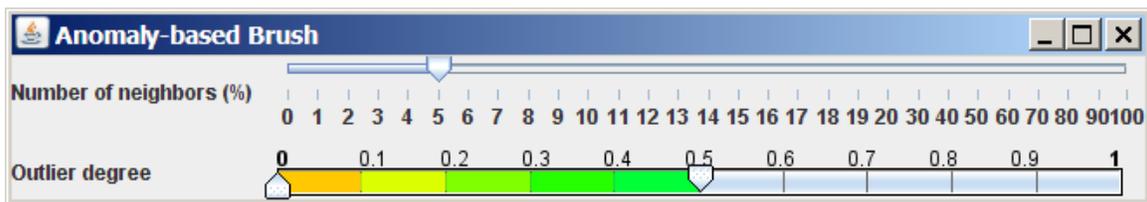
Figure 20. A snapshot of the anomaly-based brush toolbox.

We can see that the slider reflects the non-linear scale for the number_of_neighbors input parameter as mentioned in the algorithm implantation in chapter 4.

An illustration of the data display with the anomaly-based brush toolbox is shown in Figure 21. The AAUP dataset from the StatLib dataset archive [23] is displayed using the parallel coordinates graph. The input parameter for the number_of_neighbors is five percent of the dataset size, and the data range brushed is the one that is mapped to the higher outlier degrees (the upper half of the range, from 0.5 to 1). The region of the data that falls within this brush is highlighted in dark blue color, versus the light grey color for data points that are not highlighted. This color scheme has been tested to ensure the two colors chosen are visible for color-blind people [28]. With the help of this brush, unusual patterns in the data become more evident. In this case, the anomalous points are the ones that have extreme values from the second to the fifth dimension of the dataset. (Note that these are the dimensions that were chosen to detect anomalies on).
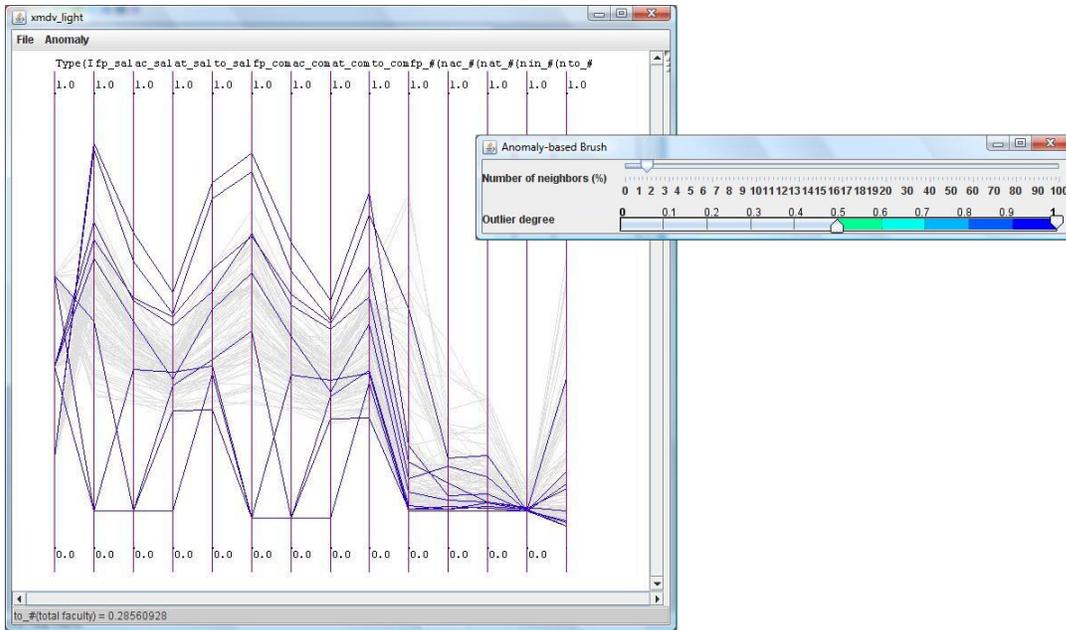
Figure 21. The anomaly-based brush on the aaup dataset, with anomalous region highlighted

As described earlier, the density-based anomaly detection algorithm is quite expensive in computation time, thus a preprocessing phase is performed to produce the outlier degree values of all the data points with the number_of_neighbors parameters of one to twenty, and then thirty, fourty, fifty, … up to one hundred percent. The results are updated on the graph in real time when users choose a value that falls in this pre-calculated range (three seconds on average). The amount of time that these updates take is independent of the datasets as there are no calculations involved. When users choose a value greater than twenty percent, the outlier degree value for each data point is recalculated and it takes longer for the process to show the changes on the graph (two minutes for a dataset of one thousand records).

With this interaction capability, the graph is able to reveal more potentially helpful data. As analysts set the number_of_neighbors input parameter to be higher, there will be fewer data points that meet this criterion, hence a bigger number of outliers. Thus if the input parameter is set too small, the algorithm will be less likely to produce the expected outliers, whereas if this number is too big, most of the data points will be identified as outliers and thus the algorithm does not return the true outliers. When analysts have control over this parameter, the results can be adjusted and evaluated on the graph each time the parameter is changed. Figures 22 and 23 show the two results generated with different input parameters. The first one uses a number_of_neighbors parameter of five percent, when using the brush to select the anomalous points, only the lower-bound extreme data points are highlighted. When we increase this parameter to twenty percent, the algorithm is able to identify the outliers on both the upper bound and lower bound of the selected dimensions (which are the second, third, fourth and fifth ones in the dataset).

Figure 22. A visualization of the AAUP dataset, with outliers highlighted, number of neighbor = 5%
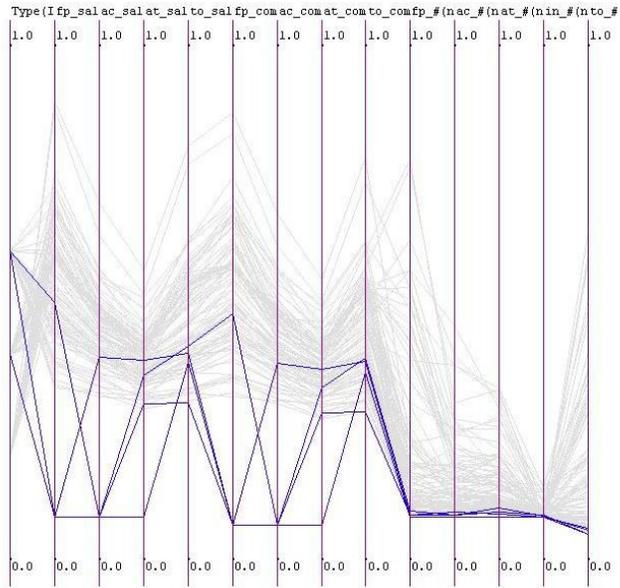
Figure 183. A visualization of the AAUP dataset, with outliers highlighted, number of neighbor = 20%
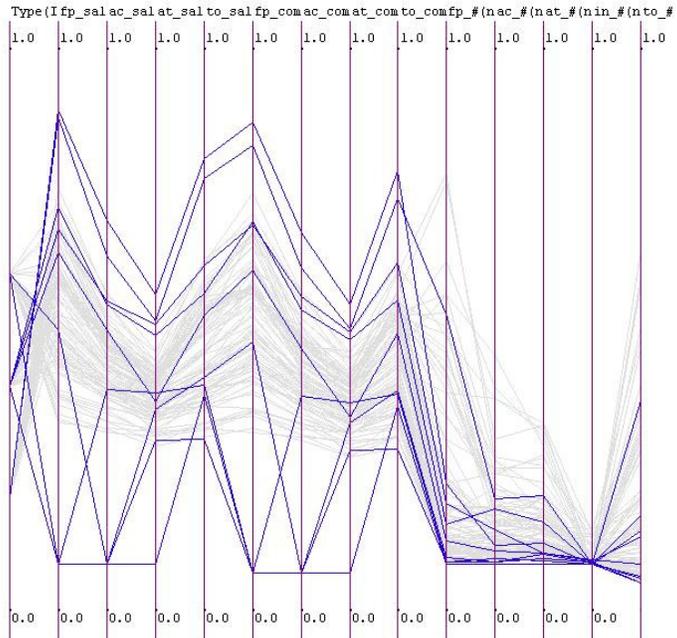
Markus Breunig et al. [29] has done intensive study about the effect of the number_of_neighbors parameter on the outlier degree of data. They suggested picking this parameter from ten to twenty as this is the range that worked well with most of the datasets that they did experiments on.

However, it depends on the nature of the datasets that this parameter should be chosen. For instance, for a dataset that has millions of data records but only a few outliers, we may just need to require a small number of neighbors for each data point to be able to detect these outlying values. On the other hand, if the dataset is a mixture of dense and sparse regions, where many outliers lie in the sparse regions, we may need to increase the density requirement so that the average distance of the neighborhood around the anomalous point is increased, resulting in a higher outlier degree for those points.

CHAPTER 6

USER EVALUATION

In this chapter, we will evaluate the usability and accuracy of the two visual interaction methods that we proposed earlier, the dimension augmentation method and the anomaly-based brushing method, using parallel coordinates graphs as the visual display. The former provides brushing capability on all dimensions, with the last dimension being the anomaly degree. The latter separates the brush for anomaly degree in a separate toolbox, adding another function for users to choose the number_of_neighbors input parameter for the anomaly detection algorithm.

6.1. DESCRIPTION OF EVALUATION

We conducted usability evaluation sessions with a group of twelve people. The users were classified into two groups: novice users (who are not familiar

with data visualization and never used XmdvTool before), and expert users (who have the domain knowledge about computer graphics, visualization and have seen/used XmdvTool). Users were taken from different areas of domain knowledge: economics (3), biology (1), physics (1), and computer science (7).

There are two datasets employed in this user study: an adaption of the cars dataset (7 dimensions, 27 data objects) and the aaup dataset (14 dimensions, 1161 records). The purpose is for users to start with a small dataset and learn how to use and evaluate the tools, and then proceed to evaluate the results of a bigger one. The evaluation process is iterative; we started the process with our original design for two visual methods, the dimension augmentation and the anomaly-based brushing, each with a certain number of functionalities. After an evaluation, we collected the feedback from the user and analyzed it to see his (her) level of satisfaction with the results produced by each method and to identify any problems that (s)he had during the interaction process with the visual displays. Based on this analysis, together with the recommendations for improvements from users, we may make some changes to the system. For example, we have removed the blue brush from the data display of the anomaly-based method as it makes the colors of the data regions on the graph easier to see; we also have changed the default outlier range chosen from the full range to just the higher range from 0.5 to 1, so that users can see the data points that have high outlier degrees right in the first glance. After a change was made, further evaluations were conducted to confirm the effectiveness of this change on the analytical discourse of users.

In order to make the perception of users about the tools objective, our strategy was to switch the order of the two methods after the tests with each

dataset, and apply the same set of questions for each method. The steps in the evaluation process are described in the Figure 24.
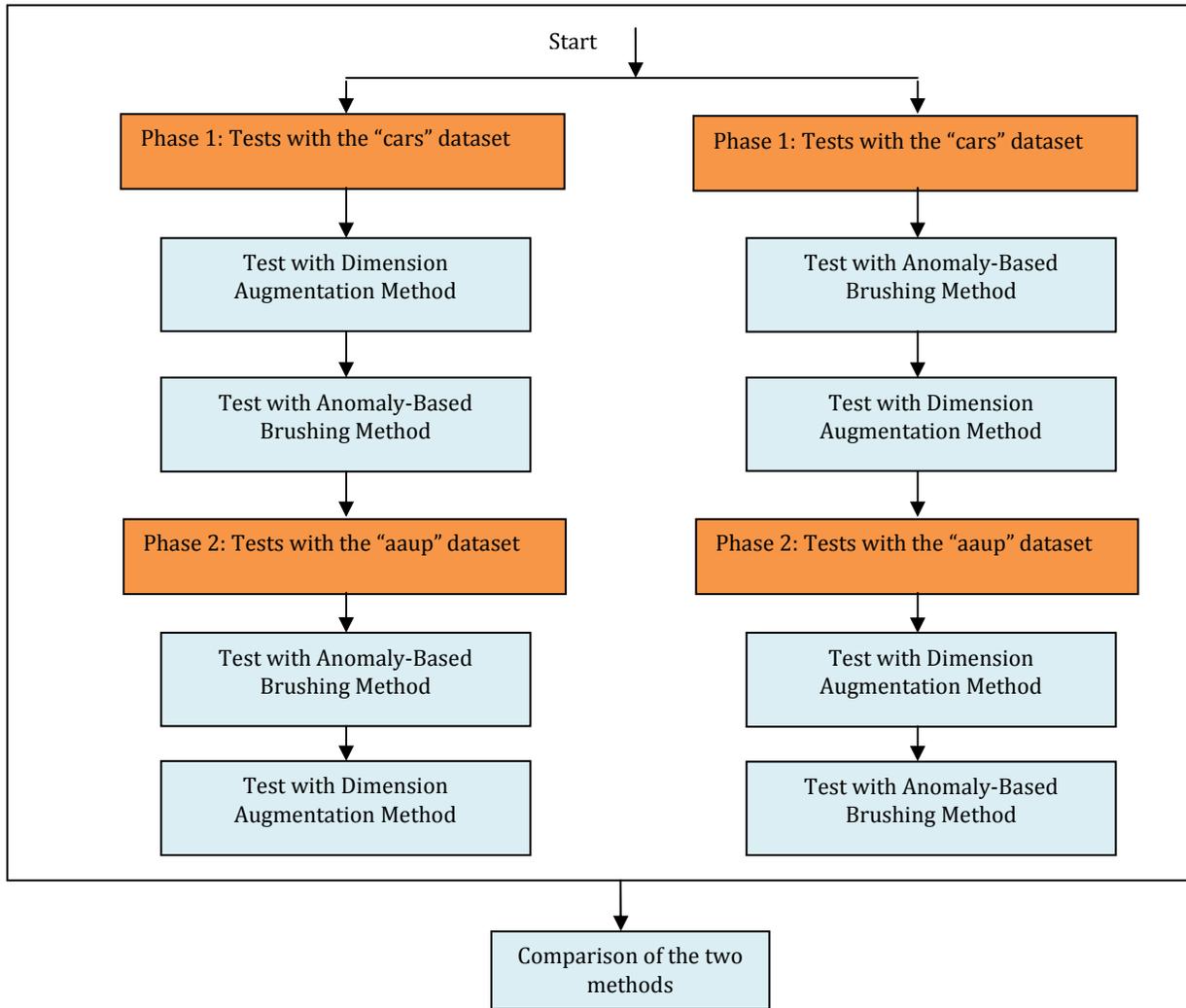
Figure 194. Illustration of the user evaluation process.

The set of tasks for users to perform on each method of a dataset includes selecting ranges in the dataset that are mapped to low/high anomaly degree for both of the methods. For the anomaly-based brushing method, users were

also asked to change the number_of_neighbors parameter in order to evaluate and adjust the results. At the end of each task, the time for accomplishing that task was recorded.

As we can see in Figure 24, an experiment with a user consisted of four sessions, each tested with a particular dataset and a particular method. After a session was completed, users were asked to evaluate the correctness of the results, the efficiency of the tool in terms of how easy the interaction was to select regions of different anomaly degrees, and the ability of the parameter adjusting functionality in helping them improve the anomaly detection results. After two sessions with a dataset, users were asked which method they would prefer to use for anomaly detection and visualization and at the end of all sessions, there was a question to get suggestions from users for each method. Evaluation from users was made in the form of ratings based on a five-point scale (for example, to rate how easy a tool is, there are five levels of ratings: "Very easy to use", "Easy to use", "Not so easy to use", "Difficult to use" and "Very difficult to use").

6.2. USABILITY REQUIREMENTS

The requirements we set for both of the anomaly visualization methods are:

- It should take a novice user less than fifteen minutes to learn how to use the tools.
- It should take less than one minute for an experienced user to find the anomalies of a level range.

- At least seventy five percent of the novice users must rate either of the tools as "Easy to use" or "Very easy to use".
- The update for visual displays with a set of parameters for the visual input (number of neighbors, anomaly degree) should take less than one minute on average to be accomplished.

6.3. USABILITY VALIDATION

6.3.1. ACCURACY

In this experiment we measured the subjective accuracy, meaning the judgment made by the subjects in the experiments about the correctness of the anomaly results. The purpose of this validation was to evaluate the satisfaction of users about the results displayed by either the dimension augmentation method or the anomaly-based brushing method, and to evaluate the effectiveness of our visual methods in conveying the anomaly attribute of data. For a specified input parameter, the brushed data regions shown by the two methods are identical as they use the same anomaly detection algorithm. The difference is only in the interaction techniques with the anomaly degree dimension. For that reason, the accuracy of the results for anomalies that we describe here is for both of the methods.

For the first dataset, to assist the participants in their process of evaluating the correctness of the displayed results, a distribution sheet of the data in the dataset was given. The table below shows the data distribution provided to users:

Number of Cylinders	Number of Data Points	Level of Outlierness
3	2	0.94
4	5	0.00
5	2	0.94
6	8	0.00
8	10	0.00

The level of "outlierness" calculated in this example was based on the number of neighbor input parameter equal to four. The dimension that we are interested in detecting anomalies in is the number of cylinders of the cars. This is a quite straightforward example; users can see that the cars having three and five cylinders seem to be outliers as the number of cars in these criteria is relatively small (two) in comparison with other cars (five, eight and ten). However in later experiments with the "aaup" dataset, where the number of dimensions was doubled and the number of data objects was much larger, the task of evaluating the results could be a lot more difficult. In this dataset we detected anomalies in four dimensions: the salaries of full, associate, assistant professors in a certain type of school and the total salary of all professors in that type of school. It is difficult to create a data distribution for this dataset as we did for the "cars" dataset described earlier, because there are many distinct data values existing in those four dimensions. Furthermore, when the number of dimensions used in the anomaly calculations is greater than two, the relationships among dimensions makes the formation of the distribution rules impossible. However the results

showed that users were still able to identify data objects that are anomalous on the graph in order to evaluate the results of the anomaly detection process. In most cases, users looked at the data objects that had one of the four dimensions falling in an extreme. Eleven participants in the experiments rated the results displayed on the graphs for anomalies were satisfactory (level one and two in a five-point scale) for both of the cars and aaup datasets. There was only one user who was not sure about the results as this user found it hard to identify the data points on the graph as the number of data lines displayed is high. Figure 25 shows the distribution of the ratings collected from users about the accuracy of the anomaly detection results for the two datasets.
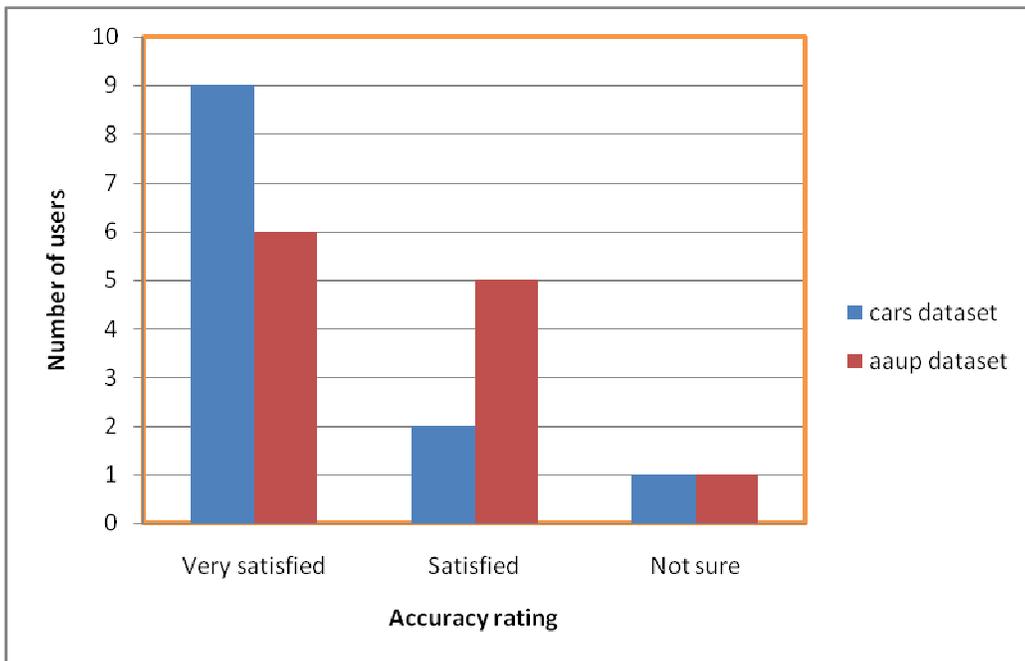

Figure 205. Distribution of user ratings for accuracy of the anomaly results in two datasets, cars and aaup.

As we can see from this chart, nine out of twelve participants rated the results to be very satisfying for the first dataset (cars). The second dataset (aaup) also has a majority of positive feedbacks from users, with six users being very satisfied with the results and five being satisfied. With this dataset, users agreed that the anomalies found were correct, as they can see visually that the anomalous data rows have at least one dimension that falls on an extreme of that dimension. The results complied with their expectations for anomalies in the dataset. The big number of data lines in the dataset was the factor that caused less certainty in the results as users were unsure about the density around one data point when it is hard to identify an individual line on the graph.

6.3.2. TIME EFFICIENCY

There are two aspects that we measured for time efficiency: the amount of time for a novice user to learn how to use the system, and for a user to accomplish the tasks of navigating data regions of different outlier degree and adjusting the results if necessary. The learning time for a novice user also indicates whether the interfaces of the two methods are easy to learn or not. It was measured from the time we began our training process about a tool (dimension augmentation or anomaly-based brush) until the time a novice user was able to start using it. For the time to accomplish the tasks, we just considered the time taken for an expert user as they are the ones that already know how to use the tools well. This strategy ensures that we separate the learning factor from the capability of users to detect and navigate different anomalous regions while interacting with the interfaces.

Figure 26 shows the learning time for the nine novice users that participated in the experiments. The red line describes the learning for users who started with the dimension augmentation method (five users); the red line shows that with the anomaly-based brushing method (four users).
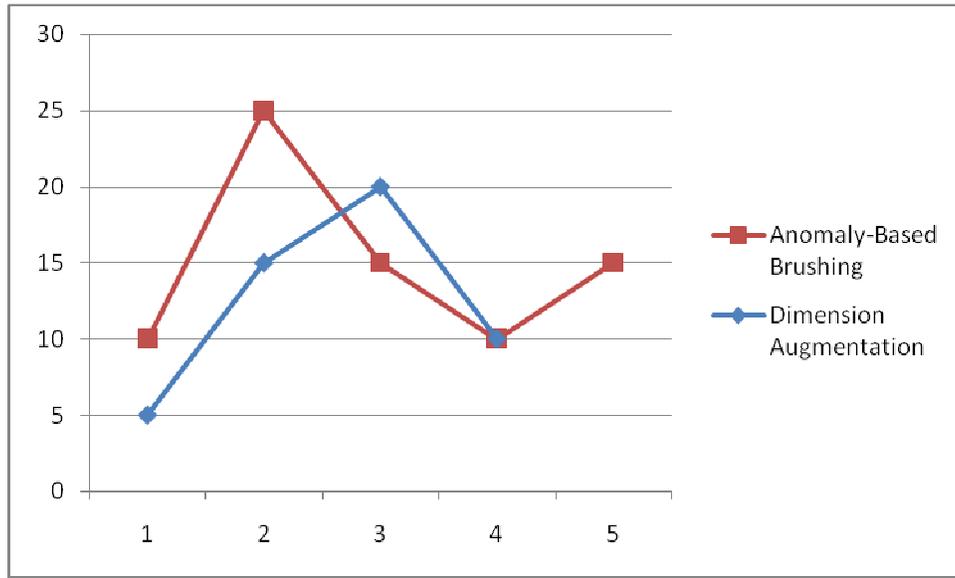
Figure 26. Comparision of the learning time of the dimension augmentation method and the anomaly-based brushing method. The horizontal axis five novice users for the first method, or four for the second one. The vertical axis represents the learning time, measured in minutes.

In the graph, the learning time shown for the first method is pretty high in user two; for the second method, the learning time is high in the third user. Generally, the second method took less time for users to learn, because of its clarity between the data space and its anomaly attribute. It was also easier for users to learn that the unbrushed region of the data was in grey and the brushed one was in dark blue. The average time for all nine users is 13.88 minutes, which is close to the expected learning time in the usability requirements (15). We also noticed that once the users learn how to use the

first graph (dimension augmentation), it took them much less time to understand the features of the second graph (anomaly-based brushing). This was because both of the methods used parallel coordinate graph as the visualization displays in this experiment, thus there were many shared features between the visual displays of the two methods.

For the anomaly navigation and detection tasks, the amount of time spent for the four professional users is displayed in Figure 27. We evaluated the accomplishing times for the dimension augmentation method and the anomaly-based brushing method based on the tasks performed with the aaup dataset. The tasks here were to highlight the data regions that were mapped to high anomaly degrees (from 0.75 to 1) and then to find the ones that are mapped to low anomaly degrees (from 0 to 0.5). The time displayed in Figure 27 is for to complete the task of complete either tasks, as the results showed the same amount for both tasks.
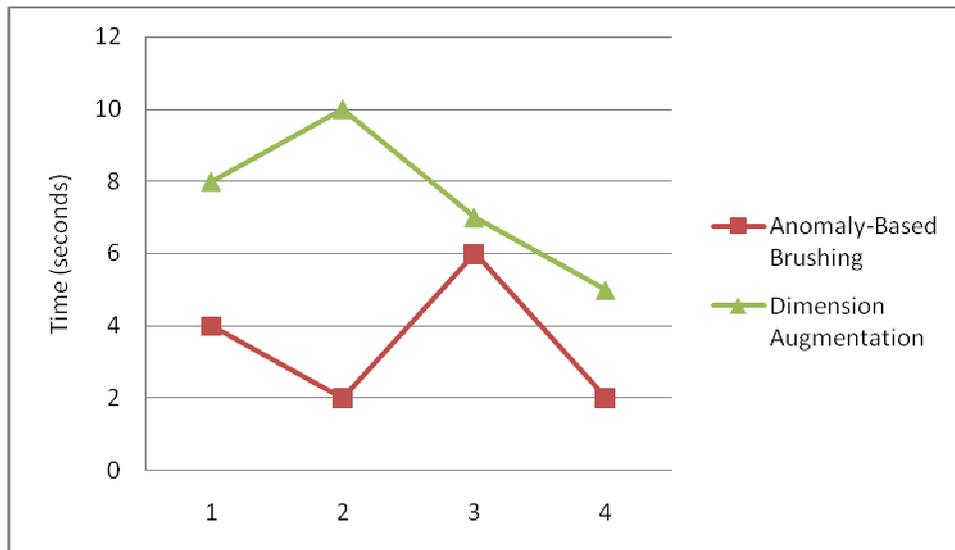
Figure 217. Comparison of the time to achieve tasks of the dimension augmentation method and the anomaly-based brushing method. Four professional users (represented along the horizontal axis), time measured in seconds.

The accomplishing time for the dimension augmentation method is noticeably bigger than for the anomaly-based brush method. In the dimension augmentation method, in order to select the data that were mapped to some anomaly degree range, first users had to choose the whole dataset, and then they could select the anomaly range by clicking to choose in the last dimension. In this method, users need to make a precise selection for a point on the graph with mouse selections. For the second method, a range of anomaly degrees was selected on a slider separated from the graph display of the data, which made the selection easier. This accounts for the big difference in the time to accomplish a task such as anomaly brushing, selecting between the two methods.

6.3.3. OTHER USABILITY CRITERIA

Figure 28 shows feedback from users when asked how easy to use each tool was. This question was asked for each dataset tested as well.
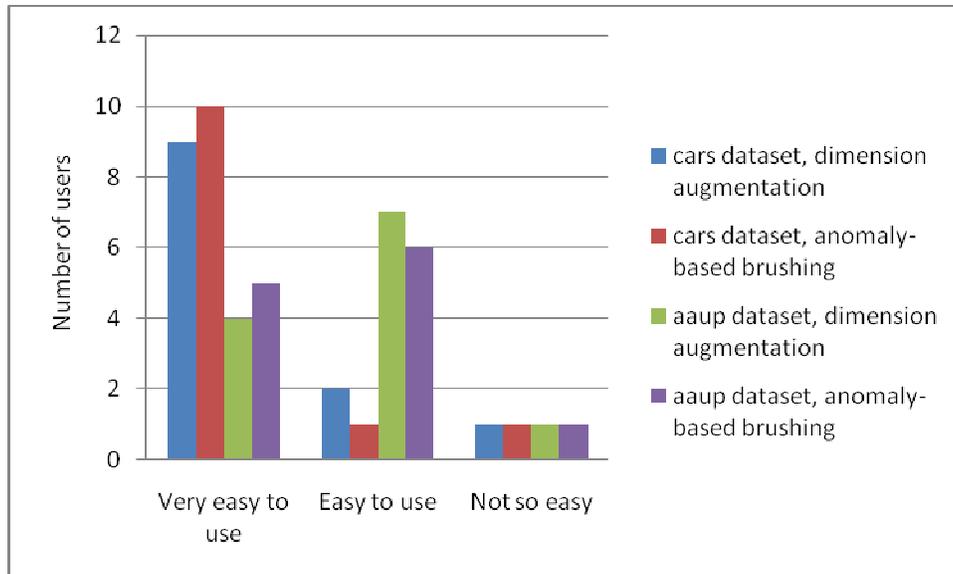
Figure 228. User ratings about how easy it is to use the visual display for each method, the dimension augmentation and the anomaly-based brushing. Tested with two datasets, cars and aaup.

As we can see on the graph, as the size of the dataset grew, there were fewer users giving the highest rate to the dimension augmentation method (4 ratings for "Very easy to use" in the aaup dataset in comparison with 9 in the cars dataset). This number was higher for the anomaly-based brushing method (6 ratings for "Very easy to use"). This data shows that the anomaly-based method was able to scale well with the size of the dataset. Overall, both of the methods received more than 90% of positive feedback from users about the ease of use of the interfaces.

Another aspect that may affect the time to accomplish a task described in the previous section (time efficiency) is the time it takes the system to reflect the changes when user choose a value for the number of neighbor. As mentioned in chapter 4, to make these changes reflected real-time, there is a

preprocessing phase that calculates the anomaly degrees for all data points with the input parameter from one to twenty percent. However, when this parameter is chosen outside the pre-calculated range, it may take one to several minutes for recalculations of the anomaly degrees for all data points, depending on the size of the dataset. In the experiments for time efficiency for accomplishing a task, the number of neighbors chosen was five percent. As the algorithm to calculate the anomaly degrees has quadratic complexity in nature, we can improve the performance either by implementing a less expensive algorithm, such as nearest neighbor algorithm with indexing the data first, and considering the trade-off between time and accuracy, or by removing the options for users to choose the number of neighbors to be bigger than the pre-calculated range.

When asked about the preference for a tool for the anomaly detection and visualization tasks, ten out of twelve participants in the experiments chose the anomaly-based brushing technique. The reason was it is clearer to view data on the graph because the anomaly degree is separated in a different window. It is also more intuitive to choose a range over the slider rather than selecting points on the graph. The visual interaction with the anomaly-based brushing is rated to be more efficient, however all of the users said they liked the feature of bi-directional mapping between data and its anomaly attribute on the dimension augmentation graph. In this approach, users can choose a range of anomaly degrees on the last dimension, and the data that matches with this criterion is highlighted. On the other hand, users can also highlight the data with some criteria specified on any dimensions of the dataset and the anomaly values for the selected data will be displayed in the last dimension. On the anomaly-based brushing graph, users can just select a range for the

anomaly degrees and the data with the anomaly attribute falling into this range will be brushed. There is no reverse mapping from the data space to the anomaly attribute space. However, this can be added to the future development of this technique, such as showing a histogram of anomaly values for the selected data.

Users also commented that the color choice of dark blue for selected data and grey for unselected one made data on this graph easier to distinguish than the red and green color choice on the dimension augmentation graph, especially on the dimension augmentation graph, the blue brush in the background made the colors of data lines harder to identify.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1. CONCLUSIONS

In this thesis, we introduce two visual techniques to interactively select and refine the data regions according to some anomaly criteria. The first technique, dimension augmentation, creates bi-directional mapping between the data space and its anomaly attribute space. The second method, anomaly-based brushing, allows users to have control over the input parameter of the density-based anomaly detection algorithm and to highlight a data range mapped to a selected range of anomaly degrees. We have done some preprocessing while implementing this algorithm to produce real-time updates in the visual process. In order to compare the effectiveness and efficiency of these two techniques, we have conducted a user evaluation on the usability of the interfaces generated for the techniques. Our preliminary results have shown that the visual display for the anomaly-based brushing

saves more time for users to achieve the tasks of brushing and evaluating anomalies. It is also commented to be more appealing to users, because of the clarity of the graphs and the color choice on the interface. These promising results encourage us to conduct more research on the technique in the future.

7.2. FUTURE WORK

One of our directions in the future is to apply hierarchical display techniques to our visual analytic approach. Displaying data in hierarchical structures is a powerful technique to handle big datasets, where the number of data records displayed can be reduced to the number of clusters [25] [30]. Besides, the anomaly detection process may reuse some of the calculations in the clustering process, or the framework for the clustering process can be adjusted to find outliers [14].

To reduce the calculation complexity of the anomaly detection algorithm, we may consider other types of methods to detect anomalies, such as using histograms [31] or distance-based outlier detection using randomization and a simple pruning rule [32] where the complexity can be reduced to nearly linear time.

For visual techniques, as we mentioned earlier in the user study section, one of the useful features that we can add to the anomaly-based brushing displays is the capability to do mapping from the data space to the anomaly attribute space. This feature will make the analysis process more flexible, as users can

choose different criteria on the data and see its anomaly degree attribute values. Also from the user study that we conducted, we see that to make data lines easier to recognize, we can add a different coloring schema for a data line that users hover over to make it stand out from the rest of the data. This feature will help users identify an individual data object when the size of the dataset grows and there are dense data regions on the graph.

REFERENCES

[1]. Pak Chung Wong, Jim Thomas. Visual Analytics, IEEE Computer Graphics and Applications, 2004, Volume 24, Number 5, page 20-21.

[2]. Michael C Schatz, Adam M Phillippy, Ben Shneiderman, Steven L Salzberg. Hawkeye: An interactive visual analytics tool for genome assemblies, Genome Biology 2007, 8(3):R34.

[3]. Florian Mansmann, Daniel A. Keim, Stephen C. North, Brian Rexroad, Daniel Sheleheda. Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring, and Interpretation of Security Threats, IEEE Transactions on Visualization and Computer Graphics, 2007, Volume 13, Issue 6, page: 1105-1112.

[4]. Ben Shneiderman. Discovering Business Intelligence Using Treemap Visualizations, B-Eye: Business Intelligence Network, 2006.

[5]. The New Google Analytics, http://www.google.com/analytics, date of download: 12/15/2007.

[6]. Ming C. Hao, Daniel A. Keim, Umeshwar Dayal, Jorn Schneidewind. Business process impact visualization and anomaly detection, Information Visualization, 2006, Volume 5 , Issue 1, page 15-27.

[7]. Tom Fawcett, Foster Provost. Adaptive Fraud Detection, Data Mining and Knowledge Discovery, 1997, Volume 1, Issue 3, page 91-316.

[8]. Marc Buyse, Stephen L. George, Stephen Evans, Nancy L. Geller, Jonas Ranstam, Bruno Scherrer, Emmanuel Lesaffre, Gordon Murray, Lutz Edler, Jane Hutton,

Theodore Colton, Peter Lachenbruch, Babu L. Verma. The role of biostatistics in the prevention, detection and treatment of fraud in clinical trials, Statistics in Medicine, 1999, Volume 18, Issue 24, page 3435-3451.

[9]. Jiong Zhang, Mohammad Zulkernine. Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection. IEEE International Conference on Communications, 2006, Volume 5, page 2388-2393.

[10]. Ke Wang, Salvatore J. Stolfo. Anomalous Payload-Based Network Intrusion Detection, Recent Advances in Intrusion Detection, 2004, Volume 3224, Pages 203-222

[11]. P. Tan, M. Steinbach, V. Kumar. Introduction to Data Mining, Addison Wesley, 2006, Chapter 10.

[12]. Matej Novotn´y, Helwig Hauser. Outlier-preserving Focus+Context Visualization in Parallel Coordinates, IEEE Transactions on Visualization and Computer Graphics, 2006, Volume: 12, Issue: 5, page 893-900.

[13]. Fabio A. González, Juan Carlos Galeano, Diego Alexander Rojas, Angélica Veloza-Suan. A neuro-immune model for discriminating and visualizing anomalies, Natural Computing Journal, 2006, Volume 5, Number 3, pages 285-304.

[14]. Ian Davidson, Matthew Ward. A Particle Visualization Framework for Clustering and Anomaly Detection, KDD Workshop on Visual Data Mining, 2001.

[15]. Pin Ren, Yan Gao, Zhichun Li, Yan Chen, Benjamin Watson. IDGraphs: NetFlow Visualization for Intrusion Detection and Analysis. 2005 Graphics and Visualization Student Symposium, IBM Research.

[16]. Ying-Huey Fua, Matthew O. Ward, Elke A. Rundensteiner. Structure-Based Brushes: A Mechanism for Navigating Hierarchically Organized Data and Information Spaces, IEEE Transactions on Visualization and Computer Graphics, 2000, pp 150-159.

[17]. Andreas Buja, Dianne Cook, Deborah F. Swayne. Interactive High-Dimensional Data Visualization. Journal of Computational and Graphical Statistics, Vol. 5, No. 1. (Mar., 1996), pp. 78-99.

[18]. David DesJardins. Outliers, Inliers, and Just Plain Liars -- New Graphical EDA+ (EDA Plus) Techniques for Understanding Data. SUGI 26' Proceedings, Paper 169-26, 2001.

[19]. Jiawei Han and Micheline Kamber, Data Mining: Concepts and Techniques, Second Edition, Chapter 7, pages 451-458. The Morgan Koffman Publishers, 2006.

[20]. Edwin M. Knorr, Raymond T. Ng. Algorithms for Mining Distance-Based Outliers in Large Datasets, Proceedings of the 24rd International Conference on Very Large Data Bases, page 392 – 403.

[21]. Edwin M. Knorr, Raymond T. Ng, Vladimir Tucakov. Distance-based outliers: Algorithms and Applications, The VLDB Journal, 2000, Volumn 8, page 237–253.

[22]. Ramaswamy, S., Rastogi, R., and Shim, K., Efficient Algorithms for Mining Outliers from Large Data Sets, Proc. of ACM SIGMOD Int. Conf. on Management of Data, 2000, pp. 427–438

[23]. The StatLib dataset library, http://lib.stat.cmu.edu/datasets/, last download: December 19th, 2007.

[24]. M. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate

data. Proc. IEEE Visualization, pages 326–333, 1994.

[25]. Nishant K. Mehta, Elke A. Rundensteiner, Matthew O. Ward, "A Hierarchy Navigation Framework: Supporting Scalable Interactive Exploration over Large Databases", Ninth International Database Engineering and Applications Symposium (IDEAS 2005), pp 425-434 July 2005.

[26]. Zaixian Xie, Matthew O. Ward, Elke A. Rundensteiner, Shiping Huang, "Integrating Data and Quality Space Interactions in Exploratory Visualizations", The Fifth International Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV 2007), pp47-60, July 2007.

[27]. Fua, Y.-H., Ward, M. O., and Rundensteiner, E. A., "Hierarchical Parallel Coordinates for Visualizing Large Multivariate Data Sets," IEEE Conf. on Visualization '99, pp 43 - 50, Oct. 1999.

[28]. Guidance for color scheme sections for color blind people, http://www.wellstyled.com/tools/colorscheme2/index-en.html, last download: Dec 15th, 2007.

[29]. Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, Jörg Sander. LOF: identifying density-based local outliers, Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, page 93-104.

[30]. Jing Yang, Matthew O. Ward and Elke A. Rundensteiner. InterRing: An Interactive Tool for Visually Navigating and Manipulating Hierarchical Structures, 2002, IEEE Symposium on Information Visualization 2002 (InfoVis 2002), page 77-84.

[31]. Matthew Gebski, Raymond K. Wong. An Efficient Histogram Method for Outlier Detection, Advances in Databases: Concepts, Systems and Applications, 2007, Volume 4443, page 176-187.

[32]. Stephen D. Bay, Mark Schwabacher. Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule, Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, page 29-38.

[33]. Dongmei Ren , Imad Rahal , William Perrizo , Kirk Scott. A vertical distance-based outlier detection method with local pruning, Proceedings of the thirteenth ACM international conference on Information and knowledge management, November 08-13, 2004.

[34]. Ji Zhang , Meng Lou , Tok Wang Ling , Hai Wang. Hos-Miner: a system for detecting outlyting subspaces of high-dimensional data, Proceedings of the Thirtieth international conference on Very large data bases, p.1265-1268, August 31-September 03, 2004.

[35]. Sridhar Ramaswamy , Rajeev Rastogi , Kyuseok Shim. Efficient algorithms for mining outliers from large data sets, ACM SIGMOD Record, v.29 n.2, p.427-438, June 2000.

[36]. J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays, In Proc. of the IEEE Symposium on Information Visualization, pages 125–132, 2005.

[37]. J. J. Miller, E. J. Wegman. Construction of line densities for parallel coordinate plots, Computing and graphics in statistics, pages 107–123. Springer-Verlag New York, Inc., 1991.

[38]. F. González, D. Dasgupta, R. Kozma. Combining negative selection and classification techniques for anomaly detection, Proceedings of the 2002 Congress on Evolutionary Computation CEC2002, page 705--710.

[39]. Paul D. Williams , Kevin P. Anchor, John L. Bebo, Gregg H. Gunsch, Gary D. Lamont. CDIS: Towards a Computer Immune System for Detecting Network Intrusions, Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection, p.117-133, 2001.

[40]. Wen Jin , Anthony K. H. Tung , Jiawei Han, Mining top-n local outliers in large databases, Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, p.293-298, August 26-29, 2001.