First publl. in: VisSym 2004: Joint Eurographics/IEEE TCVG Symposium on Visualization; Konstanz, Germany, May 19-21, 2004, pp. 55-64

Shape-Embedded-Histograms for Visual Data Mining

Amihood Amir[1†] and Reuven Kashi[2] and Daniel A. Keim[3‡§] and Nathan S. Netanyahu[4] and Markus Wawryniuk[3]

[1] Department of Computer Science, Bar-Ilan University, 52900 Ramat-Gan, Israel
[2] Rutgers Center for Operations Research, Rutgers, The State University of New Jersey, Piscataway, NJ 08854-8003, USA
[3] Department of Computer & Information Science, University of Konstanz, 78457 Konstanz, Germany
[4] Center for Automation Research, University of Maryland, College Park, MD 20742, USA

Abstract

Scatterplots are widely used in exploratory data analysis and class visualization. The advantages of scatterplots are that they are easy to understand and allow the user to draw conclusions about the attributes which span the projection screen. Unfortunately, scatterplots have the overplotting problem which is especially critical when high-dimensional data are mapped to low-dimensional visualizations. Overplotting makes it hard to detect the structure in the data, such as dependencies or areas of high density.

In this paper we show that by extending the concept of Pixel Validity (1) the problem of overplotting or occlusion can be avoided and (2) the user has the possibility to see information about an additional third variable. In our extension of the Pixel Validity concept, we summarize the data which are projected onto a given region by generating a histogram over the required attribute. This is then embedded in the visualization by a pixel-based technique.

Categories and Subject Descriptors (according to ACM CCS): I.3.m [Computer Graphics]: Miscellaneous–Visualization H.2.8 [Database Management]: Database Applications–Data Mining

1. Introduction

In this paper we propose a new visualization technique which enhances a two-dimensional projection, e.g., a scatterplot, such that the probability distribution of an additional third attribute can be perceived by the user. At a small local region in the projection plane the probability distribution of the additional attribute is shown. This enables the user to recognize dependencies between three variables, i.e., the two attributes which define the projection and the third attribute.

We call our technique *Shape-Embedded-Histograms*, because the histogram of a third attribute is embedded in the plane to which the other two attributes are projected. The embedding is done by a pixel-oriented visualization technique. The main benefit of our technique is to overcome the impact of overplot or occlusion, a well known problem of scatterplots when mapping high-dimensional data sets to low-dimensional data sets. Overplot or occlusion happens when two or more data items are mapped to the same position or when the number of data items exceeds the number of unique positions available for the visualization.

We point out two applications where the method proposed in this paper can be applied and show examples taken from real data sets.

The first application is the task of analyzing the classification of a data set. The classification (or labeling) can be known in advance or can be determined by a data mining algorithm. For instance, clustering methods give the user a set of clusters whether they are meaningful or not. The user needs techniques which allow her/him to verify the validity of the clustering. This can be done by means of statistical tests or by more informal data-oriented methods which

† Partially supported by NSF grant CCR-01-04494, and ISF grant 282/01.

‡ This work was partially funded by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-33058 PANDA project (2001-2004).

§ This work was partially funded by the Deutsche Forschungsgemeinschaft (German Research Foundation) grant Ke 740/3-1.

Konstanzer Online-Publikations-System (KOPS)
URN: http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-69708
URL: http://kops.ub.uni-konstanz.de/volltexte/2009/6970

make fewer assumptions about the data. One such informal method is visualization. A widely used visualization technique is the scatterplot, where the points are colored according to their class. The user can answer questions about the separability and similarity of the classes, and dependencies between attributes (that is, the location of points in the projection plane) and the class label can be determined.

Typically, high-dimensional data items are mapped to a low-dimensional (e.g., in our case a two-dimensional) space and colored according to their class. Due to overplotting the true class distribution can not be perceived. This means that the questions "Which classes are present at a given location?" and "How much is a particular class represented at a given location?" can not be fully answered. The literature proposed jitter and other techniques of repositioning to overcome the overplotting problem. Another technique is to draw the data items in an appropriate ordering such that the most significant items are drawn last. Our technique enables the user to recognize the true class distribution by showing the histogram of the class distribution at any given location.

The second application is the need for exploring large data sets. Analyzing data sets typically starts with analyzing the scatterplot matrix (provided that the dimensionality is not too high). People like scatterplots because they are easy to understand. Assuming that the axes of the plot correspond to the real attributes ("real" means non-transformed attributes by methods of linear combination or dimensionality reduction) one can directly draw conclusions from the scatterplots. Scatterplots give information about the two attributes which span the projection screen. The question arises: "Is there any possibility to show information about an additional third variable?" In [AKN01] the authors propose a method called *Pixel Validity Plots*. This method enables the user to get information about an additional third variable when analyzing 2D projections, making the 2D projections more expressive, particularly with regard to the fact that analyzing low-dimensional projections is very useful in order to generate previously unknown hypotheses.

The method proposed in this paper enhances the expressiveness of *Pixel Validity Plots*. In [AKN01] a pixel is called *valid* if the mass of the data which are projected onto a pixel is close to the median. The color of the pixel represents the value of the median. If the pixel is *not valid* the pixel is colored black. This works well if the probability distribution is unimodal. Bimodal distributions are likely to result in invalid pixels disregarding the fact that the bimodality might be an interesting observation. Therefore, we develop *Shape-Embedded-Histograms*, an extension of *Pixel Validity*, in order to show information of the probability distribution of the third attribute. *Shape-Embedded-Histograms* solve the overplotting problem and enhance the *Pixel Validity* by summarizing the data which are projected onto a small region in the projection plane. This is done by computing the histogram of the class distribution or the probability distribution of the data which are projected onto the particular location. Our technique of embedded histograms visualizes those histograms transferring the essential information to the user. The histogram computation restores the loss of information due to overplotting or occlusion.

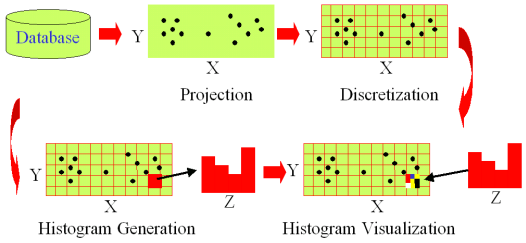

Figure 1: *Overview of the technique presented in this paper.*

An overview of our approach is shown in Figure 1. In this paper we assume that the attributes which span the projection space are selected in advance. We call these attributes X and Y. Furthermore, we assume that an additional attribute, called Z is selected. At first, all data items from the database are projected onto $X \times Y$. In the next step the projected items are discretized in order to get a grid, which is defined in the $X \times Y$ plane. Each cell of the grid corresponds to a region in the final visualization. Next, the histogram of the Z attribute for the data items falling in a particular grid cell is computed, i.e., the data of a cell are summarized and features are extracted. In the last step the histogram for a cell is visualized at the position corresponding to that cell. The histograms visualize some features of the data set, but avoid the overplotting problem which would result from visualizing the individual data items.

In this paper we restrict ourselves to axes-parallel projections. For a given data set arbitrary projections or projections determined by PCA or LDA can be useful to find dependencies which are not visible in axes-parallel projections. For instance, classes can seem to be badly separated in axes-parallel projections, but an appropriate rotation may reveal that they are well separated.

The rest of the paper is organized as follows: Section 2 presents work which is relevant to our technique. In section 3, we formalize our idea and present the essential algorithms. In section 4, we evaluate our technique by visualizing data sets taken from real world applications. We summarize our findings and point out future work in section 5.

2. Related Work

The goal of visual data mining is to combine the domain knowledge, the perceptual abilities and the creativity of the human with the computational power of computers in order to explore large high-dimensional data sets. Visual data

mining methods have been proven to be successful in many areas. An overview of techniques and applications can be found in [KW02]. In this paper we apply the concepts of several visual data mining techniques.

Scatterplots are widely used, but they have the problem of overplotting or occlusion. Several methods have been proposed in order to overcome the overplotting problem. Overplotting or occlusion is a matter of visual scalability [EK00]. The authors of [EK00] give some remarks about the scalability of scatterplots. To overcome the overplotting problem some techniques apply a repositioning of the data items by jitter [CCKT83, Cle93] or self organizing maps [TGC03]. Other techniques summarize the data and visualize the extracted features. Those methods are based on density estimation. Examples include the well-known density plot and the methods proposed in [CLNL87, Hyn96]. In most cases squares are used for binning, whereas [CLNL87] proposed to use a hexagon for binning. Another technique to reduce the overplotting problem is to draw the data items in an appropriate ordering such that the most significant items are drawn last. Also, techniques of panning and zooming can be used to analyze regions of interest.

Several techniques have been emerged for visualizing multi-dimensional data by embedding dimensions within other dimensions. One of these techniques is called Dimensional Stacking [FB90a, FB90b, LWW90, MGTS91]. In [LWW90] the n-dimensional attribute space is partitioned in two-dimensional subspaces which are "stacked" into each other. The technique requires a partitioning of the attribute value ranges into classes. The technique works best when the important attributes are used on the outer levels.

Other techniques which embed a set of information into a shape in a 2D region include Shape Coding [Bed90] and Color Icons [Lev91, KK94]. In [Bed90] the data are visualized using small arrays of fields. Each field represents one attribute value. The arrays are arranged line-by-line according to a given order (e.g., the time attribute for time-series data). Color Icons [Lev91, KK94] are arrays or shapes divided into fields and the color of the fields represents the attribute values. The arrangement of the icons can be query-dependent (e.g., spiral) or can be specified by other attributes.

Our technique can be seen as a combination of Dimensional Stacking and Shape Coding. In contrast to previous work, we stack only one dimension: We embed the histogram of the third dimension in a higher-level image. The embedding results in a dense display, whereas [LWW90] gives a sparse display. Shape coding displays many attributes at a small location. Our method achieves a higher coherence, because only one attribute is displayed.

The method proposed in this paper embeds histograms by a pixel-based technique. The idea of pixel-based techniques is to represent each attribute value by one colored pixel. The value ranges of the attributes are mapped to a fixed colormap and the attribute values for each attribute are presented in separate subwindows. A survey of pixel-based techniques can be found in [Kei00]. There are differences between the work proposed so far and our work. In this paper the bin of a histogram, i.e., "the attribute" is mapped to a number of pixels of the same color. The number of pixels depends on the value (height) of the bin and the color represents the index of the bin. In contrast, the previous approaches map the attribute value to the color and the specific attribute is recognizable by the location of the pixel.

{The database of dimensionality d is defined as}
{$DB = \{x_i : i = 1, \ldots, n \wedge x_i \in \mathbb{R}^d\}$.}
{The attributes X, Y, and Z are selected in advance.}
{Let f, g, and h be appropriate functions}
{which discretize X, Y, and Z.}

{Project and discretize the data.}
{M is a matrix of size $N_X \times N_Y$.}
{The elements of M are sets.}
for $i = 1$ to N_X **do**
  for $j = 1$ to N_Y **do**
    $M_{ij} \leftarrow \{x : x \in DB, f(x_X) = i, g(x_Y) = j\}$
  end for
end for
{Construct the histograms.}
{H is a matrix of size $N_X \times N_Y$.}
{The elements of H are histograms.}
for $i = 1$ to N_X **do**
  for $j = 1$ to N_Y **do**
    $H_{ij} \leftarrow$ histogram of the set $\{x_Z : x \in M_{ij}\}$ with respect to the discretization given by h
  end for
end for
{Visualize the histograms.}
{I is the image matrix of size $N_X \cdot S \times N_Y \cdot S$.}
for $i = 1$ to N_X **do**
  for $j = 1$ to N_Y **do**
    Histogram H_{ij} is visualized in the image region given by $[(i-1) \cdot S + 1, i \cdot S] \times [(j-1) \cdot S + 1, j \cdot S]$
  end for
end for

Figure 2: *The basic algorithm of our visualization technique.*

3. Shape-Embedded-Histograms

In this section we explain the essential algorithms. The basic algorithm which outlines our method is given in Figure 2.

3.1. The Basic Algorithm

Suppose we have a database DB and we would like to analyze the data with regard to attributes X, Y, and Z. For now

we assume that these attributes are selected in advance. Attributes X and Y correspond to the axes of the image. The goal is to visualize the probability distribution of Z depending on X and Y.

First, we have to discretize X and Y in order to get the coordinates of a data item in the image. The most natural way is to use a linear mapping of the original range to the range of the image coordinates, because this mapping preserves the distances within an attribute. Other techniques such as non-linear mappings might also be used, depending on the nature of the distribution of the attribute. For instance, if a specific attribute has a Zipf or some multimodal distribution we would give more bins to the high populated areas of the distribution. The discretization defines a grid in the X and Y space. In the final image (the image of the visualization) this grid is mapped to an equi-distant grid.

In the next step we have to generate the histograms over the Z attribute. For each cell of the grid a histogram is generated. The overall task is to visualize those histograms. If Z is a categorical attribute, e.g., if the class distribution has to be analyzed, the histogram is given by the categories. If Z is a continuous attribute, i.e., the probability distribution of Z with respect to X and Y has to be analyzed, we discretize Z in order to get an approximation of the probability distribution of Z by histograms. The discretization of Z influences the visualization. Later, in section 4, we discuss different discretization techniques including equi-width or equi-depth binning.

The attributes X and Y are mapped to $[1, N_X]$ and $[1, N_Y]$ respectively, i.e., the generated grid has a size of $N_X \times N_Y$. Note that this is not the size of the image in pixels. The histogram of the Z attribute for a specific position is represented by a small square of the size $S \times S$ pixels. Therefore the image has a size of $(S \cdot N_X) \times (S \cdot N_Y)$ pixels.

Let h be a histogram with n bins h_i ($1 \leq i \leq n$) describing the probability distribution of the Z attribute. For simplicity we assume that $\sum_{i=1}^{n} h_i = 1$. The task is to visualize the histogram in a small square of $S \times S$ pixels. We use a pixel-based technique in order to achieve this. Each bin of the histogram has to be represented by a number of pixels proportional to the height of the bin. The color of the pixels is determined by the index of the bin. We now explain the details.

The index of a bin is encoded by the color. For instance low values of a continuous variable are shown by dark colors whereas high values of a continuous variable are represented by light colors. Useful colormaps are discussed in the literature, e.g., [Lev97]. Typically, a continuous colormap is used for numerical data, whereas a set of well-distinguishable colors is used for categorical data.

The height of a bin is encoded by the number of pixels. Suppose the histogram is allowed to occupy $N_P = S \cdot S$ pixels on the screen. Bin h_i should occupy approximately $h_i \cdot N_P$ pixels on the screen. In general this is a real number and not an integer. Therefore we must round this value to get the exact number of pixels p_i for bin h_i. Here we note that $\sum_{i=1}^{n} p_i = N_P$ must be satisfied. This implies that the rounding is not as trivial a task as it first seems.

Example 1: $n = 3$, $h = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, and $N_P = 10$. After rounding each bin occupies 3 pixels. One pixel is still unoccupied.

Example 2: $n = 3$, $h = (\frac{1}{4}, \frac{1}{4}, \frac{1}{2})$, and $N_P = 10$. After rounding two bins occupy 3 pixels, and one bin occupies 5 bins. There is not enough space for 11 pixels.

A "good" rounding should satisfy

$$\sum_{i=1}^{n} p_i = N_P. \tag{1}$$

In order to specify the quality of the "good" rounding we use a least-square approach which corresponds to the Euclidean Distance between the required amount and the actual amount of pixels:

$$\sum_{i=1}^{n} \left(h_i - \frac{p_i}{N_P} \right)^2 \rightarrow \min \tag{2}$$

Furthermore it must be satisfied that each non-zero bin gets at least one pixel:

$$p_i > 0 \iff h_i > 0 \tag{3}$$

Summarizing: The equations (1), (2), and (3) require that the percentage of pixels for bin i should represent the actual value of h_i to the best possible extent.

The equation (2) is known as the *integer least-squares problem* as well as the *short vector problem*. These problems are known to be NP-hard [Ajt98, HV02]. Equation (2) has the form

$$\min_{p \in \mathbb{Z}^n} \|h - Hp\|_2 \tag{4}$$

where $H = (\frac{1}{N_P}, \ldots, \frac{1}{N_P})$ is diagonal. To solve equation (4) it is sufficient to solve the unconstrained least-squares problem and then round p_i to the nearest integer, because H is diagonal. But in our case p is constrained by equation (1) and (3). Therefore we developed a simple greedy algorithm in order to determine the best number of pixels for each bin. This algorithm is given in Figure 3. Basically the algorithm assigns an initial number of bins. If there is still an unoccupied pixel, the pixel is assigned to the bin which would yield the minimal value for (2). We note that it is possible that two bins with the same height may be represented by numbers of pixels which differ by 1.

Until now we have done the following: We mapped each data item to its grid cell. For each grid cell the histogram is computed, and for each bin the number of pixels required to visualize a particular bin is computed. What remains to be

$\{h$ is the histogram as defined in the text$\}$
$\{p$ is the vector of dimensionality $n\}$
$\{p_i$ is the number of pixels for bin $i\}$
$counter \leftarrow 0$ {number of bins i with $h_i > 0$}
for all bins i **do**
  if $h_i > 0$ **then**
    $p_i \leftarrow 1$
    $counter \leftarrow counter + 1$
  else
    $p_i \leftarrow 0$
  end if
end for
for all bins i with $h_i > 0$ **do**
  $p_i \leftarrow p_i + \lfloor (N_P - counter) \cdot h_i \rfloor$
end for
while $\sum_{i=1}^{n} p_i < N_P$ **do**
  $j \leftarrow \arg\min_{k \in \{1,..,n\}, h_k > 0}$
    $\left((h_k - \frac{p_k+1}{N_P})^2 + \sum_{i \in \{1,\ldots,n\} \setminus \{k\}, h_i > 0} (h_i - \frac{p_i}{N_P})^2 \right)$
  $p_j \leftarrow p_j + 1$
end while

Figure 3: *The algorithm to determine the number of pixel for every bin.*

done is to specify how the histogram and the bins are visualized, i.e., we need to discuss how the pixels are mapped to the square of size $S \times S$.

We use the concept of space-filling curves in order to map the bins to the pixels of the image. The idea is to draw the p_1 pixels of the first bin at the first p_1 locations on the curve, to draw the next p_2 pixels of the second bin on the locations $p_1 + 1, \ldots, p_1 + p_2$ of the curve and so on. Examples for space filling curves [Sag94] include the Hilbert Curve, the Peano (Z) Curve as well as the Column-wise Scan or the Column-wise Snake Scan. For our purpose we decided to use the Column-wise Snake Scan (a to-and-fro method). It is an intuitive mapping, easy to understand for the analyst and it avoids jumps between pixels which correspond to the same bin, i.e., regions of pixels of the same color are not interrupted.

3.2. Incorporating the Support

Until now we did not mention that some grid cells, while having the same or a similar histogram might have different support. The support is defined as the number of data items which belong to a given cell. We use the term support in order to avoid that the term probability distribution refers to the $X \times Y$ plane as well as the Z attribute. For a successful data analysis it is important to be able to distinguish between cells of high and cells of low support - and, at the same time, perceive the probability distribution of the Z attribute. The following two possibilities can be used to give the user information about the support of a particular cell:

The first possibility is to adapt the number of pixels N_P which have to be drawn in order to visualize the histogram. The value of N_P should vary in an interval $[N_{\min}, N_{\max}]$. A minimum value is needed in order to ensure that cells with a very low support still have enough pixels to draw the histogram as well as that the user is able to perceive those histograms. The support of a grid cell is mapped to this range. A linear mapping or other transformations which are better adapted to a given data set can be applied. The advantage of this approach is that the method proposed so far remains unchanged. The only change is that the value N_P must be computed for each grid cell.

The second possibility is to let the color represent the index of a bin and, at the same time, the support of the grid cell. When using the HSV color model this can be achieved as follows:

For all colors we assume full saturation. The index of a bin is mapped to hue, and the support is mapped to the lightness. An example of a colormap can be found in Figure 8. We successfully analyzed data sets with this colormap. We decided to use this colormap because the colors *aqua (cyan)*, *green*, *yellow*, *orange*, and *red* are easy to distinguish - in all cases: The difference in hue can be recognized (for the same lightness), the difference in lightness can be recognized (for the same hue) as well as different combinations of hue and lightness (for different combinations of index and support) are distinguishable. At this point we note that there is still need for research to answer the question "How can we map two parameters to color?" This subject is discussed in [Lev97].

3.3. Special Case $N_Y = 1$

The case $N_Y = 1$ can be interpreted as follows: The data items are not projected onto a 2D plane, but rather they are projected onto a single axis, namely the X attribute. This opens a new way to represent the support. So far a histogram is visualized in a square of $S \times S$ pixels. Now the histogram can be visualized in a rectangle of width S and height corresponding to the support (where an appropriate minimum height is used such that cells with a very low support are still recognizable). The algorithm implementing the space-filling curves (here Row-wise Snake Scan) needs only minor changes to handle different heights. The resulting visualization has similarities to *Bar Charts* or *Pixel Bar Charts* [KHDH02]. But the idea to show a probability distribution inside a bar is a new contribution. An example of this special case is given in Figure 9.

4. Evaluation

We tested our method with data sets taken from the *National Health Interview Survey 1993*, the *Current Popula-*

tion Survey of 1993 and the *UCI Machine Learning Repository*. To demonstrate the visualization of a third variable we used two data sets. The first one is the nhis93ac data set (National Health Interview Survey 1993), available at `http://ferret.bls.census.gov`. The second one is the cpsm93p data set (Current Population Survey of 1993 of personal records), available via the Data Extraction System (DES) on `http://www.census.gov`. The pendigits data set, available at `http://www.ics.uci.edu/~mlearn/MLRepository.html`, is used to exemplify how to analyse class distributions. For each of those data sets we use an individual colormap which is adapted to the task at hand.

4.1. Visualizing a 3rd Dimension

The first example of our visualization is taken from the NHIS data set and is shown in Figure 5. The selected attributes are *Age* (X), *Weight* (Y), and *Doctor Visits in Past 12 Month* (Z). The X and Y attributes are mapped linearly to grid coordinates. For the sake of small resolution images in papers we mapped X and Y to the range of $[1,20]$. We note that one can increase the number of grid cells. Our experiments show that a grid size of 50 by 50 gives good results. Further on we note that changing the grid size only slightly changes the results perceived from the visualization. The range of the Z attribute is from 0 to 997. We chose a nonlinear discretization in order to reflect that many people visited the doctor only a few times per year, but for the other people the number doctor visits is widespread. The resulting histogram is shown in Figure 4.

The first visualization is shown in Figure 5. The amount of different colors in a given cell represents the histogram of doctor visits for that cell. Therefore one can recognize dependencies between age, weight and the number of doctor visits. For instance, this visualization tells that increasing age as well as increasing weight results in a higher number of doctor visits per year. This is because the amount of red and orange is increasing in the upper right region of the image. For comparison the well-known scatterplot is shown in Figure 6.

One disadvantage is, that the support of the cells in a grid defined by age and weight is not represented in the visualization. Therefore the user can not know where and how the data items are distributed in the grid. In this example we show how to encode the support into the lightness of the color. The corresponding visualization is shown in Figure 7. Altogether three levels of support are encoded. The bottom left corner is the region with the highest support. One can recognize two very light cells (corresponding to the highest support) and one cell of medium lightness (corresponding to medium support). When verifying this isolated location we found that this region corresponds to children, and that this region covers about a quarter of the data. The large area in the center of the image contains cells with low support, but

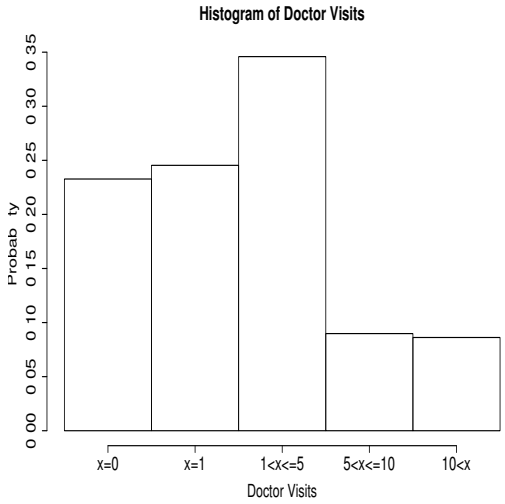

Figure 4: *The histogram shows the discretization of the* Doctor Visits in Past 12 Month *attribute into 5 bins. A nonlinear mapping is applied. The histogram is used in the experiments of Figure 5 and Figure 7. The original range is mapped to* $[0,4]$*, and the mapping is given at the x-axis.*

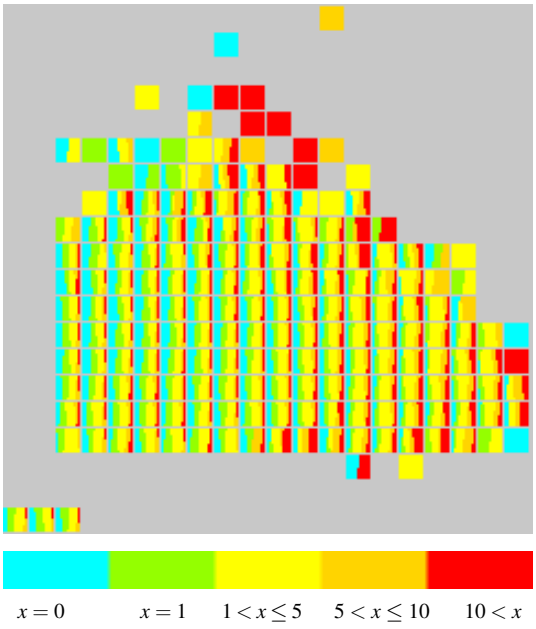

Figure 5: *An example taken from the NHIS data set. X corresponds to age and Y corresponds to weight. The embedded histogram shows the distribution of doctor visits. The visualization shows that increasing age as well as increasing weight results in a higher number of doctor visits per year. The colormap used is shown at the bottom, where aqua (red) colors correspond to a low (high) number of doctor visits.*

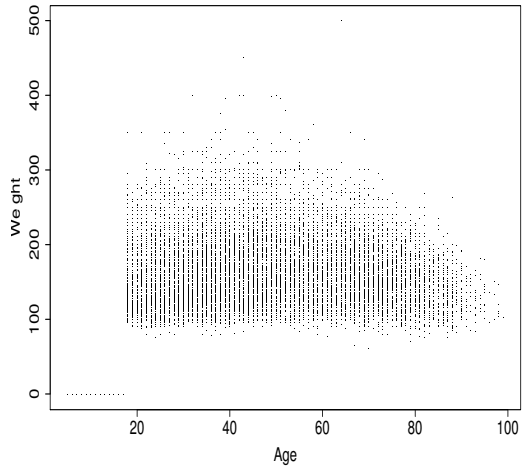

Figure 6: *The scatterplot corresponding to the data shown in Figure 5 and 7. The overplotting is 85%. With respect to* Shape-Embedded-Histograms *there is no information about a third variable or the density. Note that a weight of zero means children.*

it represents the majority of the data. There are two cells of medium support. The level of support is simply computed by dividing the range of all support values into 3 equally sized intervals.

The colormap used for Figure 7 is shown in Figure 8. For the highest level of support we use maximum lightness, and the number of doctor visits (the index of the bin) is represented by hue. The hue ranges from *aqua (cyan)* for 0 doctor visits, over *green*, *yellow*, *orange* to *red* for the highest number of doctor visits. The colormaps for the remaining two levels of support are obtained by reducing the lightness: The darker the color the lower the support.

4.2. Special Case $N_Y = 1$

The next example is taken from the CPSM data set and is shown in Figure 9. Here we demonstrate the special case where $N_Y = 1$. In this case we can intuitively represent the amount of data inside each grid cell by the size (that is, the height) of the region where the histogram is embedded.

The selected attributes are *Age* for X and *AGI (Adjusted Gross Income)* for Z. That means, the distribution of *AGI* for the different intervals of *Age* is shown. The range of *Age* is $[0,99]$ and we divided the range in ten intervals of length 10. The range of *AGI* is $[-9.999,99.999]$. We mapped all values smaller than 0 to zero (approximately 0.1% of the data), the value 0 to one (approximately 50% of the data), and the interval $[1,99.999]$ is divided into eight equi-width buckets.

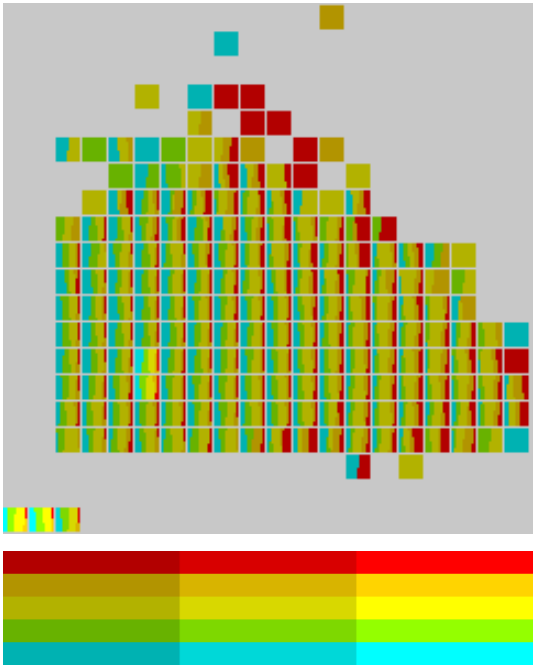

Figure 7: *This visualization corresponds to Figure 5 which does not show any information about the support of a specific cell. In this figure this is achieved by using different values for lightness. Three levels of support are shown. The colormap is explained in Figure 8.*

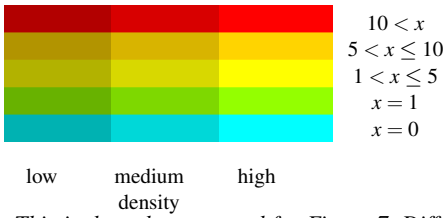

Figure 8: *This is the colormap used for Figure 7. Different levels of lightness correspond to different levels of support. Hue represents the index of the bin (that is, the number of doctor visits).*

Figure 9 shows that for middle-age people the fraction of high *AGI* is larger than for other people. It also shows that young people have a higher risk to have negative *AGI*, which can be seen by the red pixels. From the height of the different blocks it is obviously that middle-age and young people represent the biggest portion of the data set.

4.3. Analyzing Class Distributions

One data set from the UCI Machine Learning Repository is the pendigits data set. This data set contains 7494 data items and 16 dimensions which describe handwritten digits. For every attribute the range is $[0,100]$, therefore we would

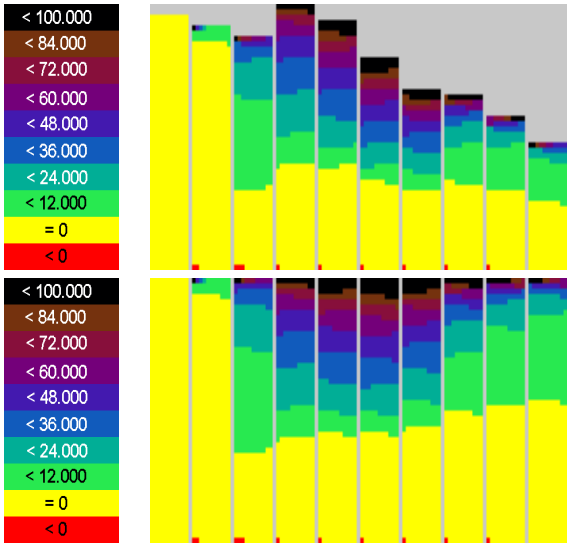

Figure 9: *An example taken from the CPSM data set. The horizontal axis corresponds to age. The height of different bins (upper part) reflects the amount of data in each bin. Inside each bin the distribution of AGI is shown. The colormap shown on the left side maps low (high) AGI to yellow (black) colors, whereas AGI below zero is mapped to red. The lower part does not map the support to the height of the bin. This allows easier comparison of different bins.*

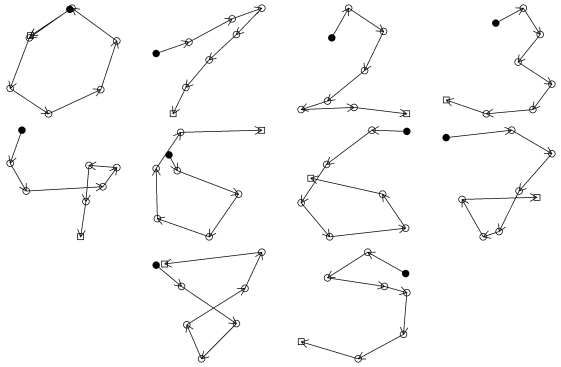

Figure 10: *Some digits reconstructed from the pendigits data set. The solid circle identifies the point where a person starts to write a digit, and the square identifies the point where a person stops writing the digit. Those points can be easily and reliable reconstructed from the pendigits data. Intermediate points are identified by circles and the direction of writing is shown by arrows. Note that points where writing is interrupted (e.g., when writing "5") can not be identified.*

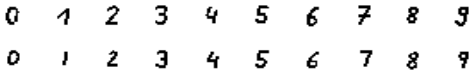

Figure 11: *Writings of the digits. The upper row represents the European style, the lower row represents the US style.*

not apply any transformation for the selected X, Y, and Z attributes. But for the purpose of the paper we mapped the variables to the interval $[1, 10]$ in order to improve the readability of the figures.

The data items of the pendigits data set can be interpreted as follows: The movement of the pen when writing a digit was recorded via a writing tablet. For each digit 8 points were obtained by spatial resampling which yields a feature vector of length 16. The first two attributes correspond to the x- and y-coordinates where a person starts to write a digit, whereas the last two attributes correspond to the x- and y-coordinates where a person stops writing the digit. Some reconstructed digits are shown in Figure 10. We present examples for every digit and note, that there are major differences in writing digits between Europe (where the data set comes from) and the US. This is shown in Figure 11. In particular this applies for the digits 1 and 7. Further more we note that some digits can not be written without lifting the pen. In the US there are 2 such digits (digit "4" and "5") and in Europe 3 such digits (digit "4", "5", and "7"). For this reason the digit "5" looks like the digit "6" in Figure 10.

Because the data set is labeled, we would like to see if there are similarities in how different people tend to write digits. In particular we would like to know where (on the writing tablet) different persons tend to start to write a specific digit as well as stopping writing the digit. We are able to accomplish this by plotting the first two attributes (the x- and y-coordinates where the pen hits the tablet the first time) as well as the last two attributes (the x- and y-coordinates where the pen is lifted) and to color the plotted data items corresponding to their class label which represents the digit.

Because there is considerable overplotting in the projections it is impossible to see any trustworthy patterns. This is shown in Figure 14. Our technique helps to overcome this situation. At a given location the histogram of the class distribution is drawn. Figures 12 and 13 show the resulting visualizations.

The long vertical yellow region in Figure 12 corresponds to the digit 1. That means, usually a person starts to write the digit 1 at the left border of the image. Here we have to note that the digits are normalized in order to make the representation invariant to translations and scale distortions. The places where a person starts to write the other digits can be identified with the colormap given in Figure 12.

Analogous is Figure 13 where one can identify the position where a person ends the writing of a digit. For instance the digit 2 (represented by red) typically ends at the bottom right corner of the image. This figure additionally gives an

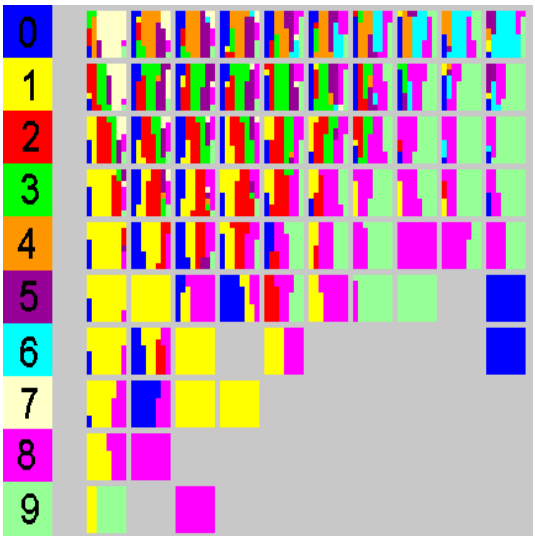

Figure 12: *This visualization shows the locations where people tend to start to write different digits, i.e., the location where the pen hits the paper is shown. The coordinates in the visualization correspond to coordinates of the tablet. The amount of digits which start at a given location (grid cell) is represented by the embedded histogram. The colormap is shown at the left side.*

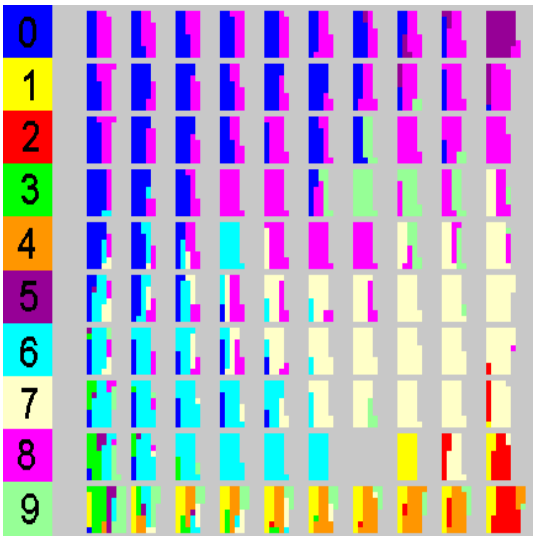

Figure 13: *This visualization corresponds to Figure 12, but instead of the first point the last point (the point where the pen is lifted) is in focus. The visualization takes care of the support as explained in the text. The colormap is shown at the left side.*

example of how to incorporate the support. Narrower blocks correspond to cells with a low support, but square or almost square-like blocks correspond to cells with a high support. The decision to represent different levels of support by the histogram size histogram is more appropriate than using lightness, because changing the lightness would result in too many colors.

One can verify those observations with Figure 10 where we reconstructed some digits. In order to compare our visualization technique, we present a "normal" scatterplot in Figure 14. While the user gets a first impression of the data the overplotting of points is fairly high.

Some of the colors in Figure 12, 13, and 14 are hard to distinguish, i.e., it is hard to distinguish the classes. Here we mention that 10 classes is a fairly high number of classes to visualize.

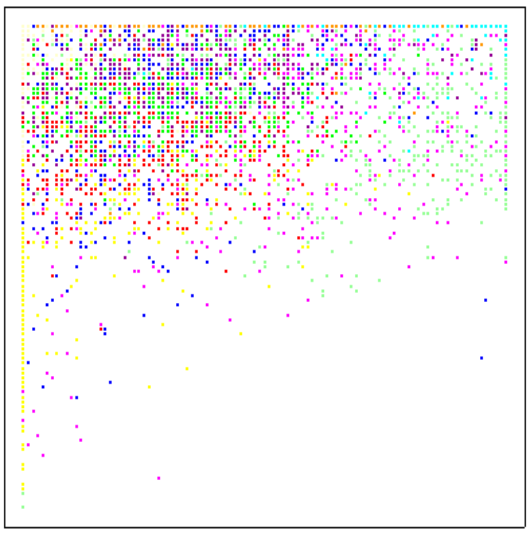

Figure 14: *This scatterplot corresponds to Figure 12. The overplotting is 40%. The same colormap as in Figure 12 and 13 is used.*

5. Conclusions

In this paper we presented a technique which makes it possible to recognize the distribution of a third attribute in the 2D projection screen, spanned by two attributes. Our experiments show that this is useful for analyzing the class distribution as well as dependencies between three variables. We have shown that by extending the concept of Pixel Validity, we are able to overcome the problems of overplotting or occlusion.

Our future research is directed by the fact that there are many combinations of three attributes, namely $\frac{d\cdot(d-1)}{2}\cdot(d-2)$ where d is the dimensionality of the data set. Not all of them have to be useful for an exploratory analysis. Therefore we want to develop criteria which can help to select the most interesting visualizations.

An important question is, how to map more than one parameter to color. In our case this means: "How can we map the combination of histogram bin and support to the color?" That means, we need a colormap which makes it possible to recognize the histogram information as well as the support of the corresponding cell. Increasing the number of support levels and/or the number of bins and, at the same time, having well-distinguishable colors is a challenging task.

The integration of *Shape-Embedded-Histograms* into systems for an exploratory data analysis is a goal of our future work. The availability of information about an additional attribute in two-dimensional plots should lead to a better quality of the exploration. Experiments with potential users are needed in order to show the success of the integration.

References

[Ajt98] AJTAI M.: The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (1998), ACM Press, pp. 10–19.

[AKN01] AMIR A., KASHI R., NETANYAHU N. S.: Analyzing quantitative databases: Image is everything. In *Proc. of 27th International Conference on Very Large Data Bases* (2001), pp. 89–98.

[Bed90] BEDDOW J.: Shape coding of multidimensional data on a mircocomputer display. In *Visualization 1990, San Francisco, CA* (1990), pp. 238–246.

[CCKT83] CHAMBERS J. M., CLEVELAND W. S., KLEINER B., TUKEY P. A.: *Graphical Methods for Data Analysis*. Wadsworth International Group, Belmont, California, 1983.

[Cle93] CLEVELAND W. S.: *Visualizing Data*, 1st ed. Hobart Press, Summit, New Jersey, U.S.A, 1993.

[CLNL87] CARR D. B., LITTLEFIELD R. J., NICHOLSON W. L., LITTLEFIELD J. S.: Scatterplot matrix techniques for large n. *Journal of the American Statistical Association 82* (1987), 424–436.

[EK00] EICK S. G., KARR A. F.: *Visual Scalability*. Technical Report 106, National Institute of Statistical Sciences, 2000.

[FB90a] FEINER S., BESHERS C.: Visualizing n-dimensional virtual worlds with n-vision. *Computer Graphics 24*, 2 (1990), 37–38.

[FB90b] FEINER S., BESHERS C.: World within world: Metaphors for exploring n-dimensional virtual worlds. In *Proc. UIST* (1990), pp. 76–83.

[HV02] HASSIBI B., VIKALO H.: On the expected complexity of integer least-squares problems. In *IEEE ICASSP* (2002).

[Hyn96] HYNDMAN R. J.: Computing and graphing highest density regions. *American Statistician 50* (1996), 120–126.

[Kei00] KEIM D. A.: Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 6*, 1 (January–March 2000), 59–78.

[KHDH02] KEIM D. A., HAO M. C., DAYAL U., HSU M.: Pixel bar charts: A visualization technique for very large multi-attribute data sets. *Information Visualization 1*, 1 (2002), 20–34.

[KK94] KEIM D. A., KRIEGEL H.-P.: VisDB: Database exploration using multidimensional visualization. *IEEE Computer Graphics & Application Journal* (September 1994), 40–49.

[KW02] KEIM D., WARD M.: *Visual Data Mining Techniques*, 2 ed. 2002.

[Lev91] LEVKOWITZ H.: Color icons: Merging color and texture perception for integrated visualization of multiple parameters. In *Visualization 1991, San Diego, CA* (1991).

[Lev97] LEVKOWITZ H.: *Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications*. Kluwer Academic Publishers, 1997.

[LWW90] LEBLANC J., WARD M. O., WITTELS N.: Exploring n-dimensional databases. In *Proceedings of Visualization 90* (1990), pp. 230–237.

[MGTS91] MIHALISIN T., GAWLINSKI E., TIMLIN J., SCHWEGLER J.: Visualizing multivariate functions, data, and distributions. *IEEE Computer Graphics and Applications 11* (1991), 28–37.

[Sag94] SAGAN H.: *Space-Filling Curves*. Springer-Verlag, New York, 1994.

[TGC03] TRUTSCHL M., GRINSTEIN G. G., CVEK U.: Intelligently resolving point occlusion. In *InfoVis 2003, IEEE Visualization Conference, Seattle, Washington, USA* (2003).