Visualization by Example

Oliver Niggemann and Benno Stein
Dept. of Mathematics and Computer Science—Knowledge-based Systems,
University of Paderborn, D–33095 Paderborn, Germany
{murray, stein}@uni-paderborn.de

Abstract

This paper presents an interaction technique for the concept analysis of large sets of objects. Given is a set of objects, O, where each object $o \in O$ is characterized by a set of features. The user is requested to select a subset O' of O, $|O'| << |O|$, and to manually arrange points, which represent the objects in O', on a plane.

This manual layout is analyzed in the following respect: Objects that are located closely to each other are interpreted as similar, and a similarity function respecting the objects's features is learned from the sample drawing. This similarity function captures the user's mental model of object similarities.

Based on the learned similarity function, the entire set O is represented as a graph G_O: (i) The objects $o \in O$ are identified with nodes, and (ii) each pair of nodes is connected by an edge weighted by their associated objects' similarity. Now, graph drawing methods are applied to G_O, and the similarity function, which was implicitly defined by O', is extrapolated to the entire set O, unveiling the interesting concepts.

Note that existing methods to similarity assessment rely on *explicit* functions given by the user. The approach of this paper is novel: It provides a high-level interface to the definition of similarity functions and allows, among other things, for the use of graph drawing as a tool for information visualisation in the field of data mining.

Keywords: User Adaption, Machine Learning, Visualization, Graph-Drawing

1 Introduction

Data mining or knowledge discovery tries to find coherencies and unknown relations between objects in large sets of data. Objects are described by means of features, e.g. people could be identified by their name, education, income, occupation, citizenship, and sex. Our approach to data mining is to combine the human ability to recognize abstract structures in visualized data and the capability of computers to handle large datasets.

For many data mining tasks the general goal of the analysis is known, e.g. when investigating the influence of people's youth on their later life. In order to support

Name	education	income	occupation	citizenship	sex
Meyer	Ms. CS.	70.000$	Sales	German	f
Smith	Ba. CS.	55.000$	Development	British	m
Wagner	High school	45.000$	Training	Dutch	f
...	...	...	...	...	...

Table 1: A typical Dataset

the human understanding of the data, the visualization should express the similarity between objects by means of spatial relations in the visualization, e.g. people with similar education, citizenship and sex should be placed closely to each other, since their youth was probably similar. Using such a visualization, the user is often able to recognize structures in the data. An overview over visualization techniques for such data can be found in [KK96].

Humans normally find it hard to define object similarities explicitly in form of a function for most datasets and analysis tasks. In simple cases, e.g. when the similarity between people equals the difference between their income, a similarity function can still be stated easily. But for most cases the process of transferring a fuzzy understanding of "similar" into an explicit mathematical function overtaxes the possibilities of most users, hence making the application of graph-based visualization methods as graph drawing impossible.

Remarkably, most people have relatively few problems to find a good layout for small datasets. This manual layout has some important features: (*i*) The spatial closeness of objects implicitly defines their similarity. (*ii*) These object distances take the general goal of the analysis task into consideration. (*iii*) By placing n objects, n^2 distances, i.e. similarities, are defined. The main idea of this paper is to use such manual layouts, analyze them, and learn automatically a similarity function, that can be used for graph drawing purposes.

When the similarity function is known, the corresponding graph can be constructed: Objects become nodes. All nodes are connected by edges weighted by their similarity[1]. This graph can be visualized using well-known visualization techniques, especially graph drawing. For example by applying a spring-embedder graph drawing algorithm (see [Ead84, FR91]), the spatial closeness of objects resembles their similarity.

This paper addresses the problem of making the user's implicit understanding of the domain explicit. For this an interface is introduced that does not rely on any explicit knowledge but that exploits the human talent for spatial thinking.

2 The High-level Interface

Datasets are normally given as a table. Every row forms one object and the columns correspond to the features. Table 1 shows a typical dataset.

The user now manually finds a layout for a small, but typical subset of the data. A typical layout can be seen in figure 1 (left-hand side). Users often position object by using abstract categories. These categories are marked in figure 1 (right-hand side). Normally these categories are less obvious and not marked explicitly.

Note that by placing one object the user defines similarities to $n-1$ other objects. This interface exploits therefore the human talent for geometric patterns.

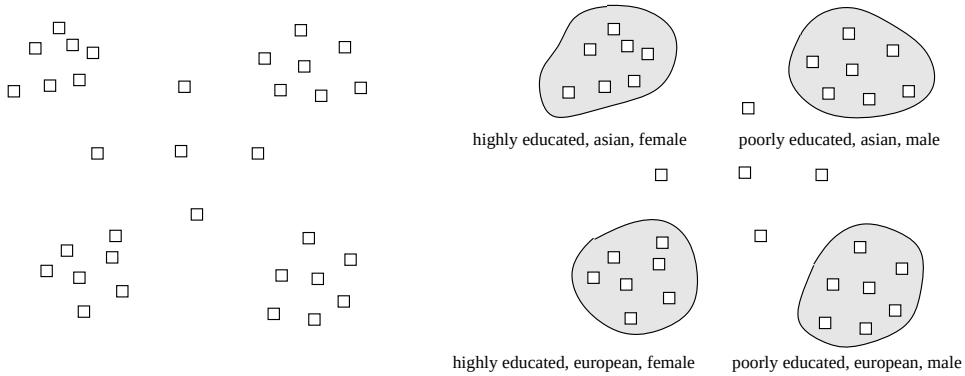

Figure 1: *A manual placement of objects.*

3 Learning a Similarity Function

Let $S = \{o^{(1)}, \ldots, o^{(n)}\}$ denote the set of objects and $o_1^{(i)}, \ldots, o_p^{(i)}$ denote the features of $o^{(i)}$. The features are presumed to be numerical. Non-numerical features can be transfered into numerical features, e.g. a feature <occupation>$\in$ {lawyer, teacher, clerk} can be turned into 3 binary features <occupation$_{\text{lawyer}}$>$\in \{0,1\}$, <occupation$_{\text{teacher}}$>$\in \{0,1\}$, and <occupation$_{\text{clerk}}$>$\in \{0,1\}$.

A function $sim(o^{(i)}, o^{(j)})$ is wanted. This function provides the missing edge weights. After the manual placement of the objects in $S' \subseteq S$ (see also section 2), each object's position is known. From this layout the similarity *sim* can be computed as follows:

$$sim(o^{(i)}, o^{(j)}) = \frac{1}{dist(o^{(i)}, o^{(j)})} \; \forall o^{(i)}, o^{(j)} \in S', \tag{1}$$

where *dist* denotes the Euclidean distance.

[1] Therefore the graph is always totally connected.

In order to learn an explicit function *sim*, a function template has to be given. The following function is a linear similarity function:

$$sim(o^{(i)}, o^{(j)}) = \sum_{k=1}^{p} w_k \cdot d_k, \text{ where } d_k = o_k^{(j)} - o_k^{(i)} \quad (2)$$

More complex function are discussed in section 6. Since for the objects in the manually layouted graph the results for the function *sim* are known (equation 1), a supervised learning strategy can be applied. In this paper, regression and neural networks are used. These methods parameterize the weights w_i. Details concerning regression and neural networks can be found in [WW81, BJ90, HL89].

4 Visualizing new Data

Now the set of objects S can be transfered into a graph $G = (V, E)$: Each object $o^{(i)} \in S$ becomes a node $v_i \in V$. V does not contain other nodes. Each pair of nodes (v_i, v_j) is connected by an edge $e \in E$ weighted with $sim(o^{(i)}, o^{(j)})$. This graph can be visualised using graph drawing methods:

For this paper two graph drawing methods have been implemented, a hierarchical drawing method according to [KST81] and a force-directed approach (see [FR91, Ead84] for details). The force-directed algorithm can be seen as a multidimensinal scaling method (see [dM99] for details). Our visualizing tool also applies clustering to support the understanding of the graph's structure and to improve the run-time behaviour. Details can be seen in section 6.

5 Realization and Application

We have applied our new method to the visualization of visitors of the Microsoft web pages. This data can be down-loaded from the Internet[2]. Every visitor is identified by an id; for every visitor a list of visited Microsoft web pages is given. So every visitor forms an object and every web page a feature. The features are boolean, i.e. the feature <MS Office> is true for visitor x iff x has visited the MS Office web page.

Figure 2 shows manually placed web users. Criteria as whether the user has visited Internet-related web pages, developer pages, or Office pages were used to place similar users closely together. The layout has been analyzed and the function *sim* according to equation 2 was learned. The average learning error, i.e. the difference between the desired similarity (as defined by the manual layout) and the learned similarity has been 7% (applying regression). Using the function *sim*, new objects, i.e. objects not used for the learning process, can be layouted. Figure 3

[2]ftp://ftp.ics.uci.edu/pub/machine-learning-databases/anonymous/

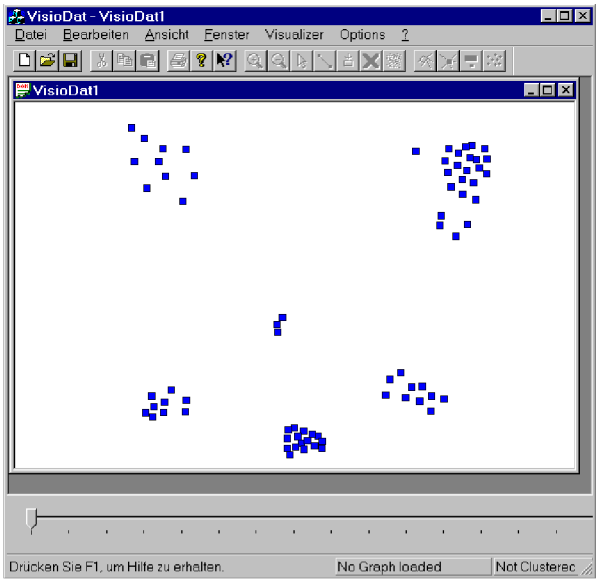

Figure 2: *A manual placement of web users.*

shows a set of new objects layouted by a spring-embedder algorithms (see [Ead84, FR91] for details). The specially outlined subgraphs $1-4$ correspond to general categories as MS Office-related pages (1), Developer pages (2), MS Explorer pages (3), and users, that only visited the start pages (4).

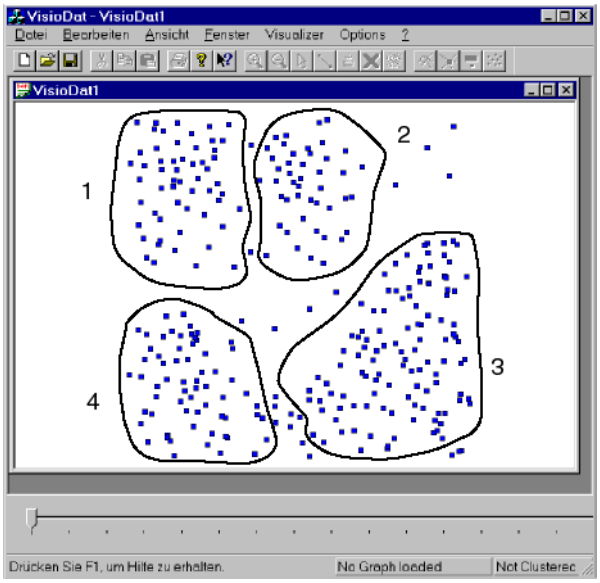

Figure 3: *An automatic placement of web users.*

While yielding interesting insights into the structure of the visitors of the MS web pages, this example may also be used to outline the limits of the approach presented in this paper: (i) The sample has to represent the whole set of objects. If this does not hold, the user has to choose another, probably bigger, sample set.

(*ii*) The user must have a vague idea of the similarity between objects and has to be able express this idea by placing objects onto a plane. (*iii*) The template of the similarity function (see section 3) has to be complex enough to capture the user's idea of similarity.

The reader may notice that while these points limit the applicability of the approach, the only alternative is a manual definition of an explicit similarity function, which will lead in almost all non trivial cases to much worse results.

Our experiments have also shown that the restriction to two dimensions, i.e. object placements on a plane, does not reduce the applicability of the approach: Users tend to arrange the objects into clusters, they hardly place objects between clusters and they rarely differentiate between objects within the same cluster. Therefore the authors have the impression that more complex definition spaces are not necessary.

6 Further Extensions

Some extension to the method described above proved helpful:

1. Using Clustering for the Learning Process

After the manual placement of objects o in the sample S' (see section 2), it is helpful to cluster the nodes. The result of the function $sim(o^{(i)}, o^{(j)}), o^{(i)}, o^{(j)} \in S'$ is then not defined by the Euclidean distance between the node position, but as follows:

$$sim(o^{(i)}, o^{(j)}) = \begin{cases} 1, & \text{if } o^{(i)}, o^{(j)} \text{ are in the same cluster} \\ 0, & \text{otherwise} \end{cases}$$

This approach emphasizes the structure of the layout. For clustering the algorithm MajorClust has been used (details can be found in [SN99]).

2. Using Clustering for the Visualization Process

By clustering new sets of data first, and then visualizing each cluster separately, the structure of the graph is outlined. When using MajorClust for this step too, similar cluster as defined by the manual layout step are found.

3. Using complex Similarity Functions

By using more complex similarity functions, the learning error can be reduced. The following function is able to capture more complex layout patterns:

$$sim(o^{(i)}, o^{(j)}) = \sum_{k=1}^{p} w_k \cdot d_k + \sum_{k_1=1}^{p} \sum_{k_2=1}^{p} w_{k_1,k_2} d_{k_1} d_{k_2}, \text{ where } d_l = o_l^{(j)} - o_l^{(i)} \qquad (3)$$

7 Summary

The new method presented here allows for the application of graph-based clustering and visualizing methods, especially graph drawing algorithms, to the problem of

visualizing and analyzing data mining data. It solves the key problem, the definition of similarities between objects, by using a graphical interface. These similarities are needed in order to find reasonable edge-weights for the corresponding graph. Manual object placements by the user are analyzed and used to learn a similarity measure.

Several problems still need to be solved: (*i*) How can a good random sample of objects be chosen? (*ii*) How much must be known about the similarity function in order to capture the user's idea of similarity? (*iii*) The effect of different clustering approaches should be evaluated. (*iv*) Can this approach be applied to the problem of learning similarity measures in the field of Case-Based Reasoning?
These problems are subject of our on-going research.

References

[BJ90] R. Beale and T. Jackson. Neural computing. Institute of Physics Publishing, Bristol, Philadephia, 1990.

[dM99] Jan de Leeuw and George Michailides. Graph layout techniques and multidimensional data analysis. Statistics Series 248, UCLA, 1999.

[Ead84] P. Eades. A heuristic for graph-drawing. *Congressus Numerantium*, 42:149–160, 1984.

[FR91] T. Fruchterman and E. Reingold. Graph-drawing by force-directed placement. *Software-Practice and Experience*, 21(11):1129–1164, 1991.

[HL89] D. W. Hosmer and S. Lemeshow. *Applied Logistic Regression.* John Wiley & Sons, New York, 1989.

[KK96] Daniel A. Keim and Hans-Peter Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[KST81] S. Tagawa K. Sugiyama and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernectics*, 11(2), 1981.

[SN99] Benno Stein and Oliver Niggemann. *25. Workshop on Graph Theory*, chapter On the Nature of Structure and its Identification. Lecture Notes on Computer Science, LNCS. Springer, Ascona, Italy, July 1999.

[WW81] T. Wonnacott and R. Wonnacott. Regression: a second course in statistics. John Wiley & Sons, New York, Chichester/Brisbane/Toronto, 1981.